



The ICT4me Curriculum

About ICT4me

ICT4me is an after school and summer curriculum for middle school youth to develop ICT fluency, interest in mathematics, and knowledge of information, communication, and technology (ICT) careers. This problem-based curriculum capitalizes on youth interest in design and communication technologies. ICT4me provides structured interactions with ICT professionals, including having youth participate in engineering design and development teams. ICT4me's promotes a train-the-trainer approach to building capacity in informal ICT learning.

Build IT vs. ICT4me

ICT4me is a derivative of the Build IT curriculum co-developed between SRI International and Girls Inc. of Alameda County. Questions about the Girls Inc. implementation of Build IT can be directed to them at <http://www.girlsinc-alameda.org/about/contact>.

SRI is no longer supporting the development of ICT4me, so the curriculum materials are offered as is.

Copyright

Copyright © 2016 by SRI International. All rights reserved.

Attribution

This material is based upon work supported by the National Science Foundation under Grant Nos. 1339181, 1232461, and 0524762. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Electronic Versions of Materials

Electronic versions of all materials in this unit are available for download from the website at <http://ict4me.sri.com/>.

Contact Information

Please contact the SRI International Inquiry line for questions about ICT4me.
<https://www.sri.com/contact/form>

Unit 5: Designing and Programming a Game

Overview

Youth design, program, and assemble one big game made up of several stages. Youth work in pairs to design and develop different stages, and then all pairs collaborate to put the stages together into the one big game. The overarching idea of the unit is that small pieces can be put together to make a larger product (e.g., each student makes one stage of a larger game; each stage is composed of smaller elements; each character is manipulated by a number of programmed behaviors; each character sprite is made up of pixels).

Unit 5 was designed using Stagecast Creator (\$30 software); however, any object-oriented programming software, like Scratch, will work in similar ways. Please see the “Alternate Technology Options” section at the end of this unit for an in-depth comparison.

Enduring Understandings

- Collaboration involves a strategy for dividing tasks associated with a solution into pieces that can be worked on individually and reassembling the work products into a cohesive whole to form the solution (NRC, SCANS).
- Leadership involves teaching others new skills, communicating ideas to justify a position and convince others, and supporting a vision that may challenge the status quo (SCANS).
- To troubleshoot a problem in an information technology system, application, or operation, it is essential to have some expectation of what the proper behavior should be and how it might fail to be realized (NRC).
- Algebra: represent patterns in tables, with graphs and with symbolic expressions.

Essential Questions

- How do you decide what to build?
- What is programming?

Unit Layout

Unit 5 is designed for two 2 hours and 10 minutes over 15 weeks. This document includes weekly leader preparation and curriculum sections.

The Summary and Getting Ready sections are to help leaders prepare for a week’s activities. The Summary section includes the Schedule and goals, Essential Questions, Design Process concepts, Glossary definitions, and a list of all the Materials needed that

week. The Getting Ready section includes an overview of the week's activities and Background information for the leader.

Every week is broken down into Warm-up, Challenge, Main Activity, and Discussion/Reflection curricular sections. Activity Pages include the handouts needed for the week.

Thoughts on Gender

Design is all around us, done by adults and youth. ICT4me units are designed to engage all youth in learning about design and Computer Science. It was especially designed for getting girls, African-American and Latino/a youth hands on opportunities to learn and develop expertise in these fields.

All youth should have an opportunity to explore the materials without being deterred by their own or others preconceptions about gender and race, in safe environments that promote collaboration, learning, and self-expression. All youth should have the same opportunity to see themselves reflected in the ICT professionals with whom they interact.

Gender Tips appear in orange boxes throughout the curriculum, with ideas on how to address particularly sticky topics.



Gender Tips

Connecting gender to ability (or lack of) or the way someone or something looks or behaves is a slur. Just like a racial slur. Explain to youth that slurs (racial, gender, sexual orientation, age, etc.) are not cool and not welcome.

Table of Contents

Unit 5: Designing and Programming a Game.....	1
Week 1: Programming vs. Playing Games	6
Summary.....	6
Getting Ready	7
Warm-Up.....	12
Challenge	14
Main Activity	15
Discussion/Reflection	17
Week 2: Learning Stagecast with PondLife.....	18
Summary.....	18
Getting Ready	19
Warm-Up.....	22
Challenge	24
Main Activity: Math Activity (option 1)	27
Main Activity: Math Activity (option 2)	32
Discussion/Reflection	33
Week 3: Visiting Electronic Arts & Selecting Your Team	34
Summary.....	34
Getting Ready	35
Warm-Up.....	36
Challenge	37
Discussion/Reflection	39
Main Activity	40
Week 4: Principles of Game Design & Stagecast Tutorial	41
Summary.....	41
Getting Ready	42
Warm-Up.....	44
Challenge	45
Main Activity	49
Discussion/Reflection	50
Week 5: Designing the Big Game	51
Summary.....	51
Getting Ready	52

Warm-Up	54
Challenge	56
Main Activity	58
Discussion/Reflection	60
Week 6: Prototyping & Storyboarding Stages	61
Summary	61
Getting Ready	62
Warm-Up	64
Challenge	65
Main Activity	67
Discussion/Reflection	68
Week 7: First Week of Programming	69
Summary	69
Getting Ready	70
Warm-Up	72
Challenge	74
Main Activity	76
Discussion/Reflection	77
Week 8: Second Week of Programming	78
Summary	78
Getting Ready	79
Warm-Up	80
Main Activity	81
Discussion/Reflection	82
Week 9: Third Week of Programming	83
Summary	83
Getting Ready	84
Warm-Up	85
Main Activity	86
Discussion/Reflection	87
Week 10: Testing Stages & Fourth Week of Programming	88
Summary	88
Getting Ready	89
Warm-Up	91
Challenge	92

Main Activity	93
Discussion/Reflection	94
Week 11: Fifth Week of Programming	95
Summary	95
Getting Ready	96
Main Activity	97
Discussion/Reflection	98
Week 12: Sixth Week of Programming	99
Summary	99
Getting Ready	100
Main Activity	101
Discussion/Reflection	102
Week 13: Putting Humpty Dumpty Back Together Again.....	103
Summary	103
Getting Ready	104
Challenge	106
Main Activity	108
Weeks 14 and 15: Family Tech Night.....	109
Summary	109
Getting Ready	110
Warm-Up: Brainstorming a Plan for FTN	112
Main Activity: Creating a Plan for FTN	113
Discussion/Reflection	114
FTN Presentations.....	115
Activity Pages	116
Using Scratch Instead of Stagecast	Error! Bookmark not defined.

Week 1: Programming vs. Playing Games

Summary

Schedule

Warm-Up	Discuss goals and objectives for the next semester. Brainstorm topics for the game.	30 min
Challenge	Introduce Stagecast Creator software.	40 min
Main Activity	Explore the difference between programming and playing.	60 min
Discussion/Reflection	Introduce ICT4me	10 min
Total Time		2 hr 20 min

Essential Questions

- Deciding what to make?
- What is programming?

Design Process Concepts Involved

- Define the problem.
- Research it.



Materials

- Computers with Internet access
- Computers with Stagecast Creator
- LCD projector (for showing PowerPoint presentation and Stagecast)
- 5x7 Post-its
- Introduction PowerPoint (G-G D-zine)
- Handouts

Getting Ready

Overview

During this unit, ICT4me youth will create *one* game with many levels or challenges. The whole group will brainstorm the game together and will split up into pairs for development. Each pair will design and develop one level or challenge. They will put the game together before testing the game with younger users.

The overarching idea of the unit is that small pieces can be put together to make a larger product (e.g., each student makes one stage of a larger game; each **stage** is composed of smaller elements; each **character** is manipulated by a number of programmed behaviors; each character is made up of many **pixels**). In Unit 5, youth will become familiar with Stagecast Creator as a programming environment and program their own game.

During The Warm-Up and Challenge are, youth come up with the topics they would like to teach younger students as they enter middle school. Then they play a few Stagecast games to get the feel for what is available. In the Main Activity and Discussion/Reflection, youth play and change aspects of MontyMole to discover the difference between playing and programming. Both are fun and involve interactions with Stagecast. The youth will learn that they can change the game by changing the rules of the game, not by playing the game.

Glossary

We hope to familiarize you, the facilitator, with the terms and concepts that will be used in the activity. Youth will be working with these concepts and ideas, but using the terms could be distracting. You can use the proper names for some objects at your discretion, if the youth will not be turned off from the activities.

- **Iteration.** A method for revisiting and improving upon the results of one step of a process. Sometimes games require several iterations: you may have an idea at first, then as you learn about Stagecast you'll refine that idea, and in the process of creating it and putting it together, you may end up refining it more. The end result may be different from what you started with, but it is indeed based on the original idea.
- **Stagecast Creator.** A highly graphical, intuitive software environment in which anyone can learn computer science concepts while programming a game or activity of their own.
- **Computer Games.** There are many games on computers these days. The youth will be using Stagecast Creator to program a game of their own design.
- **Sims.** The files created by Stagecast Creator. Sim is an abbreviation for "simulations." "A sim can be a game, a story, an animation, or anything you create in the sim window. Sims are made up of characters that you draw, then animate by giving them rules of behavior. Characters can interact with other characters on a stage and be controlled by

you through the mouse or keyboard. Together, the stage, the characters, and their actions make up a sim.”¹

- **Characters.** “Any object you create in Stagecast Creator. Each of the objects on the stage is a character. You create rules for characters to tell them how to perform on the stage.”
- **Drawers.** Located on the right side of the sim window and provide access to characters, stages, and all programming options.
- **Rules.** The “instructions you give a character to make it perform on stage.” These are basic programming units of Stagecast.
- **Tools.** (Play and Stop buttons, Paint, Delete, Copy).
- **Object-oriented programming.** A style of programming that gives each object (say a character in a game) a set of rules to define actions. For example, if you want an animal to move across the screen, you have to tell it to move up, down, left or right, otherwise it just stays where it is. From that moment, the animal will move until the environment does not match the rule you specified. Each object in the game has to be “programmed” to do anything. The other more typical kind of programming usually entails a long sequence of instructions that enable the computer or computer software to do something.

Stagecast Learning Goals

- Learn about Play and Stop buttons.
- Find and use the Copy, Delete and Paint tools.
- Learn how to drag out characters onto the stage from the Character Drawer.

Background

What are the challenges that middle school youth face that you would like to teach younger students how to deal with. Examples could be coping with school, afterschool, home, and homework. Obstacles in the youth’s social lives include copycats, bullies, cliques, blaming others, outcasts, peer-pressure, and smoking. An online resource on how to approach youth about these topics is: <http://www.4girls.gov/>.

Stagecast Creator is a great way to introduce concepts of object-oriented programming and programming in general. The focus in Stagecast Creator is on the creation of a game or activity, not the programming. Yet, the youth will learn how to program. One leader described the learning gains for ICT4me teens after finishing Unit 5: “They learned how to program, for sure. They learned how to troubleshoot. They learned how to work together.” So, youth will get to use the design process skills they are familiar with while learning programming techniques and computer science concepts.

¹ Source: Quotations about Stagecast Creator software features are taken from *Stagecast Creators Guide*, found in the Stagecast Creator 2 application folder.

To get started, you'll want to spend some time getting familiar with Stagecast Creator. ***Do all of the Stagecast activities that the youth will be doing. Create your own game (really!). And, do the tutorial that comes with the CD.***

You will also need to download and review all the rules and characters for the four model games (see below). You can review the entire makeup of the game by opening a game, clicking Make Report, and viewing the HTML file in any browser.

Youth will play with four model games: Sokoban, Jessi's Winter World, Eliot's Duck, and the GIAC game. You'll find these games at:

- c:\program files\stagecast\sims\sample sims\
- c:\program files\stagecast\sims\guide\
- <http://www.stagecast.com/cgi-bin/templator.cgi?PAGE=Cool/worlds/ALPHAWORLDS>
(Click on "download" not "play")

Put the model games into one easily accessible folder before you start this lesson, so the youth can find them quickly and are not wasting time navigating to different folders in Stagecast or downloading the game Eliot's Duck from the Internet.

Also, before showing the Introduction PowerPoint (for G-G D-zine), you'll need to add your name for the lead engineer liaison position. This will establish your role as the person who is in charge of seeing the project's completion during the 15 week unit.

Make sure you have the following materials to start:

- Introduction PowerPoint (G-G D-zine)
- Stagecast Resource for Students (separate document)
- ICT4me Sims zipped file (includes Unit 5 Starter Kit and games)

There are some charts or handouts you'll refer to often, and you may want to leave them up on the walls:

- Design Process chart
- Design Requirements handout
- Job Description handout
- Brainstorm of middle school topic chart (from week 1)
- Designer or programmer Hat chart (from week 4)
- Brainstorm of the Big Game chart (from week 5)
- Map of the Big Game chart (from week 5)
- Game reports (produced by Stagecast)

Purchasing Stagecast

Stagecast is only \$30 for a full license. Read more about the Stagecast Creator software and how others have used it at <http://www.stagecast.com>. Purchases should be made through the Stagecast website or email to orders@stagecast.com.

Stagecast Resources

- Stagecast Creators Guide (on CD-ROM)
- Stagecast Creator Teacher's Guide (on CD-ROM)
- Stagecast Creator Tutorial (on CD-ROM)
- Stagecast Creator Tips and Tricks (http://www.stagecast.com/cgi-bin/templator.cgi?PAGE=Shared/support/TIPS_TRICKS)
- TurboBetty Kit (on CD-ROM)
- Reignhead Club Lessons (<http://www.gamelearning.pwp.blueyonder.co.uk/index.HTML?clubs/resources/resources.htm>)
- The GameMaker's Apprentice (order book from <http://book.gamemaker.nl/frames.htm>)
- Direct your questions about Stagecast to Stagecast at support@stagecast.com.

Scratch Software

Scratch is free and just as easy to use as Stagecast. For more information about Scratch, visit <http://scratch.mit.edu/> and <http://scratched.media.mit.edu/>.

Scratch Resources

- See Scratch Help at <http://scratch.mit.edu/help/> for links to Getting Started resources, including guides, videos, starter projects, cards, and tutorials,
- Scratch Curriculum, <http://scratched.media.mit.edu/resources/scratch-curriculum-guide-draft>
- Design Studio (very useful if substituting Scratch in this unit), <http://scratched.media.mit.edu/sites/default/files/DesignStudio.pdf>
- Debugging challenges, replacement for debugging tutorial for Stagecast, <http://scratched.media.mit.edu/resources/debug-it>
- For putting multiple programs/stages together, see Remix <http://wiki.scratch.mit.edu/wiki/Remix> and Backpack <http://wiki.scratch.mit.edu/wiki/Backpack>

Scratch Games

- Puzzle: <http://scratch.mit.edu/projects/3124143/>
- Arcade: <http://scratch.mit.edu/projects/909079/>

- 3D Maze: <http://scratch.mit.edu/projects/1652740/>
- 2D Maze: <http://scratch.mit.edu/projects/507/>

Warm-Up

Time:	30 minutes
Purpose:	Understand what design is. Ask what games (if any) the youth like to play. Discuss what makes a game fun. Explain goals of unit. Brainstorm topics for game.
Materials	<ul style="list-style-type: none">• 5 x 7 Post-it notes• Job Description handout• Introduction PowerPoint (G-G D-zine)

To Do

1. Ask youth to describe their favorite computer games. Make a list of the games.
 - a. Do you play these games alone, with someone on the same computer, or over the Internet?
 - b. Why do you like playing these games (e.g., graphics are cool, game is engaging, for competitive reasons)?
 - c. What do you think is required to make a computer games (design, graphing, storyboarding, planning).
2. Explain the goals of the unit:
 - a. Show the Design Process chart and tell the youth they will be working on different steps of the design process each week.
 - b. Explain they will be creating *one* game together. Each pair will work on one step or level of the game. The steps or levels will be put together at the end.
 - c. Tell them that they will be designing a game for younger users about to enter middle school.
3. Show the Introduction PowerPoint (G-G D-zine) to the unit. Add narrative if you think it is necessary, but you can just have student read it.
 - a. Hand out or post in a prominent place (during the entire unit) the Job Description. Explain that they will be learning about each of the materials and tools for design in the following weeks.
 - b. Handout the Design Requirements and review them with the youth. They should keep these as the beginning of a binder or folder for the unit.

Brainstorming

4. Ask youth what the first steps in the Design Process are? (Define the problem and brainstorm. You can point to these step in the Design Process poster). What is the problem that you want to solve? (They just heard from G-G D-zine what the topic is).
 - a. Explain that they are going to brainstorm topics that would be important for a younger student to learn about middle school. This iteration will be the first on the topic, and it serves to set the stage for the entire unit.
 - b. Give youth a few minutes to think about the experiences and issues they have had in middle school.
 - c. Ask them to write down on Post-it notes some of these issues.
 - d. Have them put the Post-it notes in a central place so all youth can read them.
 - e. Do a bit of rearranging out loud with the youth. Put similar issues or experiences together and if necessary, help them make new categories. For example, if a few Post-it notes mention homework or classroom work, put all of those together and label them “school work” or “assignments.”

Teaching Tips

Save the brainstorm chart

5. Lead a discussion about which of these issues seem most important. Here are suggestions for facilitating the discussion:
 - a. Which of these issues and experiences are central to middle school?
 - b. If you had to share your experiences with a younger student, what would you tell them is the most important thing to learn based on these Post-it notes?
 - c. Depending on how many pairs there are, ask: “Can we make a list of the top (x) issues to include in our game?”
 - d. Tell youth that they will be referring to this brainstorm later on, to guide their design process.
 - e. Youth will iterate on the ideas from the brainstorming later on in the unit. This exercise will allow them to focus their efforts, but will not constrain them if new or refined ideas come up along the way.



Challenge

Time:	40 minutes
Purpose:	Set the stage for discussing the difference between playing and programming a game. Play with the game models (mazes, adventures, puzzle). Explore the Stagecast environment.
Materials	<ul style="list-style-type: none">• Chart paper• Computers with Stagecast Creator software• Access to model games: Sokoban, Jessi’s Winter World, Eliot’s Duck, and the GIAC game

To Do

1. Introduce Stagecast Creator software. Make this brief, since the youth will dive into the program in the next few weeks.
 - a. Explain that the youth are going to learn about Stagecast Creator, software that allows them to make games.
 - b. Ask youth to open Stagecast Creator, and to “Create new.”
 - c. Show them how to put a character on the stage and ask them to change the drawing of the initial green star to something else (e.g., a face, a stick figure).
 - d. Show the youth how to program their characters to move forward. Show them how to make a rule and use the Play and Stop buttons
 - e. Explain that they will be learning how to make graphics and program games using Stagecast during this unit.
2. Try out the Stagecast games.
 - a. Ask youth to play all the four model games: Sokoban, Jessi’s Winter World, Eliot’s Duck, and GIAC. Remind them to hit the “Play” button to start each game.
 - b. As they play the games, have the youth fill out the Game Comparison Table.
 - c. About 5-10 minutes before time is up, discuss each of the games. The youth should share their notes about what they liked about each game, and why it would or wouldn’t be fun for younger students.
 - i. Ask about differences and similarities between the games.
 - ii. Probe for what makes a game fun (e.g., it’s a puzzle, there are many ways of winning, the goals is attainable, the rules are easy to follow, there are many steps, it gets harder or easier, the colors are cool).

Main Activity

Time: 60 minutes

Purpose: Show youth what is meant by “programming” in the Stagecast environment. Help youth understand the difference between playing and programming a computer game (you must stop the game before you can program it and start it again to see the results).

Show how to create, move, delete, and copy characters.

Help youth become familiar with the terms used for describing parts of the Stagecast screen including the Play and Stop buttons, the Delete tool, the Copy tool, the Stage, the Sidelines, the Character Drawer, the Time Bar and the Speed buttons.

Materials

- Computers with Stagecast Creator
- Computer connected to overhead
- MontyMole.sim loaded on computers (in ICT4me sims folder)

To Do

1. Tell youth that they are going to start learning how to program by making changes to the MontyMole game.
2. Ask youth to open MontyMole.sim. Monty, the mole, is looking for worms to eat. When he gets full, he needs to find his way home to bed.
3. Show the MontyMole game on the overhead projector. Ask the youth to locate the worm and the bed. Now, hit the Play button and wait to see happens with the mole. Ask youth what they think is happening. Monty will not go anywhere because there is a rock in his way.



4. Ask the group for possible solutions (e.g., delete the rocks, move Monty).

5. Demonstrate one of those solutions to the group (e.g., move Monty to the other side of the rock). Then hit the Play button again to see what happens. Explain that in order to fix the game, they (and you) have to hit the Stop button – they can't edit while the game is running.
6. Using the Play button is often the best way to check whether something they changed in the game worked or not.
7. Let the youth try out fixing Monty's problem on their own computers.
8. When everyone has had a chance to free Monty from the rockslides, ask for suggestions on making worms accessible to Monty (by moving a rock or by copying a worm onto Monty's path).
9. Demonstrate using the Copy tool on the worms.
10. Tell the youth to try the Copy tool on their own computers and when they are ready, ask them to press the Play button.
11. When Monty has finished his worms, show that there are more objects that Monty can encounter. Demonstrate opening the Character Drawer and dragging another worm to the tunnels.



Tech Tips

We found that youth will have lots of fun with Monty's worm-eating laughter and other funny noises in the game, especially when they start cloning Monty and he bumps into himself in the bedroom.

12. Ask the youth to drag whatever items they want (*except the key*) to the tunnels and to press Play when they want to see what happens with Monty.
13. After a few minutes, ask the youth to drag the key to the tunnels to see what happens.
14. Now, ask the youth to open one of the games they played with earlier to see whether they can change the game by moving things, adding characters, or deleting others.
15. Walk around to assess whether the youth are using all the tools introduced in this activity.

 **Discussion/Reflection**

Time:	10 minutes
Purpose:	Reinforce Glossary and Stagecast tools.
Materials	Middle school topic chart (made In the Warm Up and Challenge)

To Do

1. Ask the youth to tell you what steps in the design process they did. (Define the problem, brainstorm, and research it). This is a good way to start the discussion every week.
2. Ask them what the difference between programming and playing a game is.
3. Have the youth share new things they learned about Stagecast (e.g., when you hold down the Copy or the Delete tools after you have selected a character on the stage, you can make or delete many copies).
4. Ask them what they think about their game topics now that they have played a few games. Do they want to add, delete, or modify any topics? They will revisit this question more formally in a few weeks, but it will be good to start modeling the iterations between topic and games and between games and topics.

Week 2: Learning Stagecast with PondLife

Summary

Schedule

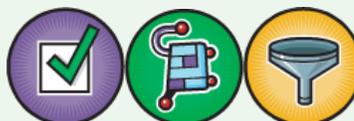
Warm-Up	How did Monty know where to go?	20 min
Challenge	Demonstrate rule creation. Practice making rules for the PondLife game. Share some of the buttons and tools the youth used.	40 min
Main Activity	Explore coordinate planes and rates of change.	60 min
Discussion/Reflection	Discuss math activity	10 min
Total Time		2 hr 10 min

Essential Questions

- What is programming?

Design Process Concepts Involved

- Research it
- Build it
- Test it



Materials

- Computers with Stagecast Creator software
- LCD projector
- PondLife no rules.sim on all computers
- PondLife Rules, from the *Unit 5 Rules Reference*
- Grid paper
- Handout

Getting Ready

Overview

These activities involve working with the PondLife.sim.

In the Warm Up and Challenge, youth will learn how to make rules for movement and interaction. The main idea here is to work on the “if-then” rule sequence, and to emphasize that “if you don’t program the character to behave in certain ways, then it doesn’t know what to do.” To prepare, do Stagecast activities yourself before the activity. Create all the PondLife rules yourself using these materials and the *Unit 5 Rules Reference*.

In the Main Activity and Discussion/Reflection, youth will start thinking about the mathematical structures (screen layout and rules of motion) of making a game in Stagecast.

Glossary

- **Congruence.** Two figures are congruent if they are the same shape and size—even if they are in different positions.
- **Cartesian.** Related to René Descartes, French philosopher and mathematician.
- **Cartesian coordinate system.** Two perpendicular axes in the plane define a Cartesian coordinate system. The place where these two axes intersect is called “origin” for both of them. Usually, but not always, one of the two axes is horizontal, other vertical; their positive directions to the right and upwards. Usually, but again not always, the horizontal axis is called the x-axis, the vertical one is called the y-axis.²
- **Coordinate point.** A specific place on the coordinate system that can be found by finding the intersection of an x value and a y value. For example (1, 2) or (-4, -3).
- **Equation.** An algebraic representation of a function (something that changes). In the diagram above, the equation of the red line is $y = x + 1$. If you enter the x values and do the operation you will find the y values that make up the red line.

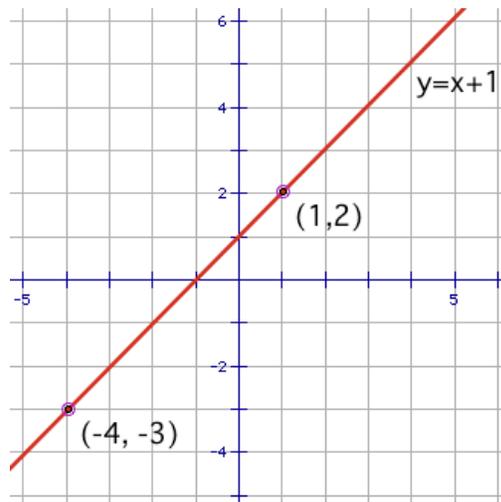


Figure 1. Cartesian Coordinate System and graph of $y = x + 1$

the
the
are

² Definition taken from <http://www.cut-the-knot.org/Curriculum/Calculus/Coordinates.shtml>. If link doesn't work from within MS Word, copy and paste into your browser.

- **Graph.** A graphical representation of a function (something that changes). In the diagram above, the graph of $y = x + 1$ is the red line.
- **Slope.** The ratio of y/x or the amount of change in y per every x . In the example above, the slope of the line $y = x + 1$ is 1, because for every x , the graph goes up one y . If you start at $(-4, -3)$ and move up one x value to -3 , then the y value will be -2 .
- **Cause-and-effect or causal relationships.** When one thing is true or happens, then there is a certain effect. The if-then statements in Stagecast programming are of this type: If there is an empty space, then move object to the left.

Math Concepts

- Coordinate geometry
- Locating points on a grid
- Plotting functions (describing paths)

Determining distance between points

Stagecast Learning Rules in this Activity

Movement

5. Swim left (tadpole)
6. Swim down (tadpole)
7. Drift right (floating weed)
8. Swim up (froglet)
9. Fly left (insect)
10. Fly right (insect)
11. Frog jump (frog)
12. Frog land (frog)
13. Pull tadpole down (frog spawn)

Appearance

14. Turn into froglet (tadpole)
15. Turn into frog (froglet)
16. Turn around (insect)

Interaction

17. Frog vanish (frog)
18. Click bottom left (frog spawn)

Stagecast Tools

19. Don't care square (p. 87, *Stagecast Creators Guide*)
20. On mouse click (p. 87, *Stagecast Creators Guide*)
21. PondLife report (from PondLife.sim, click on Create report)
22. PondLife rules

Stagecast Tutorial Reference

23. Getting Started
24. Making Your Own Rules
25. Making More Rules

Background

There are three math concepts in this week's activities: cause-and-effect relationships (aka causal relationships), the Cartesian coordinate system, and motion and range of change.

In the Warm Up and Challenge, youth explore the logic structure of causal relationships using **if-then statements**. Object-oriented programming, like the one we

use in Stagecast, uses this structure to follow through a set of rules for each character. If a set of conditions is met, then Stagecast performs some action(s) for a character. If the set of conditions is not met, the software does not apply that rule and moves on to the next rule in the list (when there is another rule). Youth will learn to interpret and write if-then sentences for each rule they make throughout Unit 5. In the Warm-Up and Challenge of week 2 is the first time they will experience the rules.

Prepare for Stagecast rule reading, creating, and testing by looking over the Stagecast Creator 2, *Creators Guide*, “Using the Rule Maker,” pp. 80-91.

In the Main Activity and Discussion/Reflection, youth invent their own ways of describing the position of objects and then learn about the **Cartesian coordinate system**. Help youth understand how the Cartesian plane helps describe the location and size of objects with precision. The location of the origin is arbitrary but important—arbitrary because you can decide to put it anywhere; important because you need to make sure other people can figure out what the starting point is. Many people working on the computer, whether creating web pages, using software, or creating games, use the structure of the coordinate system all the time to describe positions on the screen. The youth need to be familiar with this system when they determine the size of their stages, when they are working with pre-made stages, and when they try to figure out where to place their characters for their games.

The third major concept, also In the Main Activity and Discussion/Reflection, relates to **motion and rates of change**. In the math activity we ask youth to calculate the fastest route the tadpole can take to get to the edible weeds. The goal is to have them figure out what the route is, and then to program the rule to see if it is true. The rule for the tadpole in this case is the equivalent of the rate of change or the slope in a linear equation. There is no need at this stage to emphasize the formal names of any of these concepts. You want youth to learn how to generalize their understanding of rate of change from the PondLife stage to larger stages.

Resources on Descartes and the fly:

- <http://mathforum.org/cgraph/history/fly.HTML>
- *A Fly on the Ceiling* (Step-Into-Reading, Step 4) (Paperback) by Julie Glass (Author), Richard Walz (Illustrator)

Warm-Up

Time: 20 minutes

Purpose: Learn if-then rules

Materials

- Overhead examples of if-then sequences
- MontyMole.sim
- Computers with Stagecast Creator software

To Do

1. Open MontyMole and ask youth. How did Monty know what to do? How did he know where to move?

(Answer: Monty was programmed).

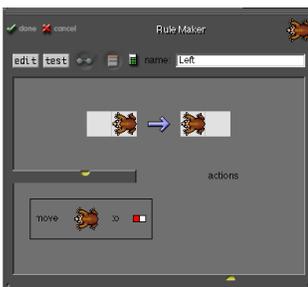
2. Open up MontyMole.sim and double click on Monty, then open the LEFT rule. Show the tab for the IF portion of the statement, and ask them to tell you what this part of the rule is saying.

(Answer: IF there is an empty square to left of MontyMole, and IF his appearance is the left-looking mole)



Show them the THEN portion of the rule.

(Answer: THEN move Monty one square to the left).



3. Ask youth to click on Monty's up, down, left, and right rules.

Tech Tips

These rules (up, down, left, right) appear towards the bottom of the list of rules. Why would that be? Think about the order of firing. If these were at the top, Monty would never do anything else first.

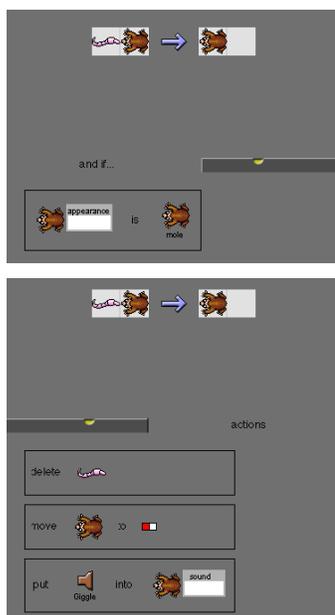
Model a few more if-then sentences. Help youth focus by asking what they notice, how many squares there are, and what the difference is between the before and after pictures.

- Using the If...Then handout, ask the youth to make sentences that combine the two images. Note that there are many types of actions that a character may perform: appearance may change (if the worm is in X position, then he'll turn on his side); movement can occur (how characters move up, down, left, right); and interaction can take place (when faced with a wall, character turns around).

Repeating the formal if-then wording is important. It will formalize the format of the rule: If such and such, then something will happen. You can help youth internalize the format in this exercise and throughout the unit.

- Show the MontyMole rules that combine several conditions (IFs, e.g., Enter Door Up), or several actions (THENS, Get Key Right, Eat Worm Left).
- Together write the “IF...and IF...THEN...and” statements

Example: Eat Worm Left



Tech Tips

You should repeatedly ask the youth to look at rules to tell you how they would write them out in the if-then form. You want them to learn that the software notices all the conditionals (IFs) before it applies a rule. When something in the situation does not fit the conditions, then the software proceeds to the next rule. You will also have an opportunity to start talking about the order of rules, if only briefly. First rules fire first. If a rule is important or has precedence, it should occur before other rules with less importance.

 **Challenge**

Time: 40 minutes

Purpose: Practice programming.
Make if-then rules for characters.

Materials

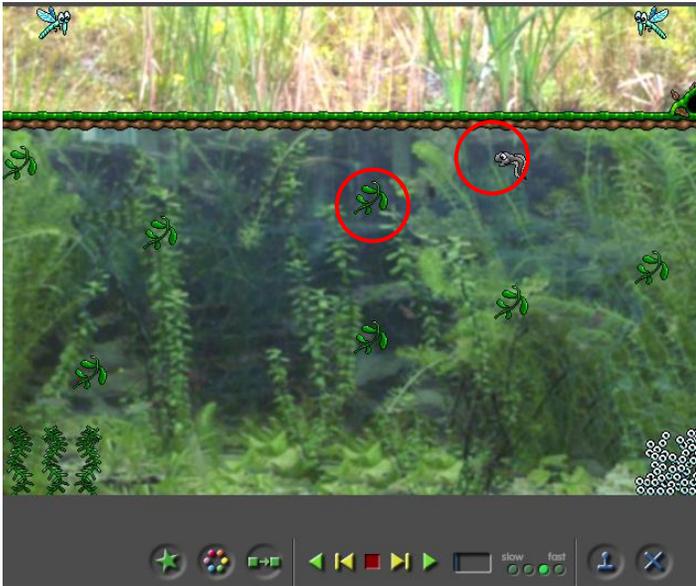
- Computer with Stagecast Creator software
- LCD projector
- PondLife.sim (**without** rules) (in ICT4me sims folder)

To Do

Modeling how to create and test rules

1. Open PondLife no rules.sim so everyone can see it. Press Play. Ask youth to tell you what happened. (Answer: Nothing happened, because there are no rules).
2. Say: You will learn how to create rules so that the tadpole swims down to find the edible seaweeds, then swims up to become a frog, and finally hops to its house.
3. Press Stop and ask youth to describe how to make a **swim left** rule for the tadpole. (Hint: Nothing should be to the left of the tadpole). **MAKE SURE YOU WRITE THE NAME OF THE RULE.** This is the most often forgotten action, and it leads to a mess of unnamed rules.
4. Ask youth to describe what is going to happen. Then, press Play. Tadpole will move to left until there is a weed. Making a rule, then pressing Play is one way to test the rule and find out where it breaks down.
5. Press Stop and ask youth to explain what happened. This is your opportunity to teach them to **create** a rule, **test** the rule, and **explain** what happens when the rule stops working.
6. Ask the youth to describe how to make a **swim down** rule for the tadpole each time it encounters a weed. (Hint: Weed should be immediately to the left of the tadpole).
7. Press Play and see whether the rule works. Then, place the tadpole back at the starting place. Continue to state what you are doing: I'm going to start the game to check whether my rule works. It's not enough to model, you also must make the connection between what you are doing and what you want them to notice explicitly.
8. Now, ask youth to describe how to make the weeds **drift right**. (Hint: Nothing should be to right of the weed). Follow their instructions closely. If the youth make a mistake, this will be a good opportunity to demonstrate publicly how to troubleshoot a rule. (Read how to do this in the Stagecast Creator™ 2, *Creators Guide* "Testing Rules," p. 83).

9. Press Play. See how the tadpole and all the weeds move.
10. Have youth make the rules for the tadpole and the weed on their own computers. Encourage them to take notes on the rules they have to make.



11. Gather the whole group again, and collectively create new rules. Run the sim to see the tadpole move across the screen and hit the edible weeds. Ask the youth how they think the frog made it the last level down? (The frogspawn has a rule to pull the tadpole down). Explain that you can create a rule that affects the main character (aka tadpole), by making a rule for another character in the game (aka frog spawn) to interact with the main character. Show the **pull tadpole down** rule, ask the youth to narrate what it says, and talk about the Don't Care Square. This is the first time that we see the Don't Care Square, which tells the program to ignore anything that might be in the square.
12. Stop the game and ask the youth to show how to turn the tadpole into a froglet (by changing the appearance). Emphasize that the weed and the tadpole both have to be in the rule (and in this orientation). This is the first time we change the appearance of a character—or change the character completely.
13. Ask the youth to create a **swim up** rule for the froglet. Press Play and watch the froglet swim to top.



14. Have youth practice changing the froglet's appearance to turn the froglet into a frog when the froglet arrives at the bank. Repeat that the turning into a frog requires the bank square, the square above the bank, and the froglet (nothing

should be immediately above the froglet). Press Play and watch the frog jump home (notice that the frog still has its own rules).

15. Show the **On Mouse Click** rule for creating new tadpoles. This is the first time youth see a user interaction, but it won't be the last. If you think the youth are overwhelmed, just ask them to click on the frogspawn when they are playing the game. If you think they can learn a bit more, then ask them to look at the rule and analyze it.
16. Once all the youth have all the rules, challenge them to make flying rules for the dragonflies:
 - a. **fly left**
 - b. **fly right**
 - c. **turn around** (when dragonflies bump into each other; this requires a new step—changing appearance).



17. When everyone is done with their simulation, ask youth to point out rules that they didn't make that appear in the simulation.
 - a. **frog jump** and **frog land** (Frog leaps across the bank).
 - b. **frog vanish** (Frog enters his house).
 - c. **pull tadpole down** (Tadpole goes one level down when he's above the eggs).
 - d. **click bottom left**, **click bottom right**, **click top left**, **click top right** (Eggs float to surface and turn into tadpoles).

(Whether youth discovered this on their own, or whether they didn't see it, show them the rules created for the eggs. See whether they can tell the story of each of the rules to predict what happens when you hit Play).

18. Ask the youth what other rules they can think would make PondLife.sim more like a pond. Some possible answers could be:

Make the edible weeds flow to simulate water.

Add more characters.

Main Activity: Math Activity (option 1)

Time: 60 minutes

Purpose: Use the coordinate grid as a way to plan movement, actions, and location of characters.

Explore the rate of change (slope, speed) of the tadpole and generalize for larger screens.

Determine the paths of characters as they move across the screen.

Materials

- Computers with Stagecast Creator software
- Grid paper the size of the screen (14 x 20)
- Whiteboard
- PondLife on a grid
- LCD projector
- Grid paper overhead
- PondLife.sim

To Do

1. Ask youth to sit in pairs with blank sheets of paper, in such a way to prevent their partners from seeing what they are drawing (5 minutes). Youth 1 (in the pair) should draw a *diamond* on their paper. Ask Youth 1 to describe the location and size of their *diamond*. Youth 2 will try to draw the *diamond* by listening to Youth 1 give instructions. During this stage, the youth are *not* allowed to use conventional measurements (inches, centimeters) to describe the location of their *diamond*. When partners have reproduced the original diamonds, have the youth compare their drawings to see how they did.
 - a. Notice how youth decide that their drawings are similar. Are they lining up the pages to see if the *diamonds* are congruent?
 - b. Do this stage only once. For the second part, the youth will switch roles for



who draws and who describes.

When asking questions in the math activity, make an effort to ask those students whom you think are not so good in math. When they respond, really pay attention and try to understand what they say, engage with them, even if it is not an answer you expect. Typically, there is logic to youth's wrong answers. See if you can figure it out. This effort will show them that you believe in them, even when they doubt their own abilities.

After each prediction, or question, do not give away the solution or reject incorrect answers. Probe answers and ask for justifications for each response.

2. As a whole group, ask some pairs of students to tell you how they explained location to each other (5 minutes).
 - a. Probe for rich descriptions. Write on the board any words that were used for describing position, length, or orientation.
 - b. Length/size: Highlight ways in which the youth recreated a Cartesian system by using units like "one finger's width."
 - c. Origin: Ask youth to explain how they described their starting place (e.g., "Fold your paper like a hotdog/hamburger, then follow the middle groove to the center of the paper.") The location of the origin is arbitrary, but if agreed upon, it gives partners a point of reference.
 - d. Sources of knowledge: Ask the youth how they know that their *diamonds* were similar or not. Always probe them on how they know. This helps them reflect on their process and gain insight into some of the deep math concepts in this activity.
 - e. Discuss the descriptors on the board. Point out some similarities (all the words that stand in for conventional units, for example).
3. Hand the youth grid paper and ask Youth 2 (in the pair) to do repeat the process. This time they should draw a line. (Draw a line and describe the location of the line to your partner).
4. Ask the youth to compare their drawings to see how well they did this time around.
 - a. Write down on the board some of the words they used to describe the size and location of their lines. You should expect them to refer to the number of squares from a certain point. Probe for starting point (origin). Where did they decide to start counting? How did they communicate this to their partners?
 - b. Highlight some of the similarities and differences from the previous drawings. Point out that in the first activity they invented their own coordinate system, without using conventional math terms. Then, explain that the confusion of having to make up new terms and ways of explaining location is why we all still use the Cartesian coordinate system.

5. Tell the story of Descartes lying in bed sick, staring at flies on the ceiling. Explain how he developed the idea of the coordinate plane. Demonstrate various examples of objects on the grid paper overhead, naming their coordinate points. Youth will probably remember naming conventions for coordinates; if not, review (x, y) . (You can omit this story if you are pressed for time).
6. Ask youth to explain how the coordinate system is relevant to programming and ICT4me.
7. Add your explanation by starting the next activity. Show a screenshot of the PondLife.sim, with the tadpole in the right hand corner position and a few of the floating weeds at regular intervals.
8. Ask the youth to describe where the tadpole is on the screen. Do not provide the right answers and try not to give hints. Ask clarifying and probing questions to make sure everyone understands and is on the same page.
 - a. Depending on where the youth put the origin, the answers could be:
 - i. $(20, 10)$ - if origin is bottom left
 - ii. $(20, -5)$ - if origin is top left
 - b. As the youth answer, ask them what reference point they used as the origin? Youth will give different answers. If they don't, suggest different corners as the origin to see whether they can determine where the tadpole is.
 - c. Explain that programmers use any of the four corners of the screen as the origin, as long as they agree on which one. Most programmers use the top left or the bottom left corners as $(0,0)$.
9. Have the youth open up their PondLife games. Tell the youth that one of the points of the game is for the tadpole to spend as little energy as possible getting to the food. The longer the tadpole takes, the more tired it gets. Now, ask the youth to tell you:
 - a. How many turns will it take the tadpole to get to the edible weeds? (Answer: 55 turns. To figure it out, youth can click on the yellow right-arrow to see how long it takes. With each click, they can see where all the characters will be).
 - b. Can they figure out where to put the floating weeds to get the tadpole to the edible weeds (bottom) in fewer steps? More quickly?
 - i. They could also disable the floating weed rule to simplify the task (refer to p. 69 of the *Stagecast Creators Guide*).
 - c. What is the fastest route to the edible weeds from the current position?
 - i. Use the PondLife with lines handout to have them practice. Some solutions are provided at end of this document.

10. Now, have youth delete all the floating weeds.³ Can the youth create ONE rule to make the tadpole arrive at the edible weeds the fastest?
- We are exploring the UNIT RATE of the tadpole's movement (without calling it that). This is the core of the activity. This ONE rule is equivalent to a mathematical function, such as $y = ax + c$, in which you set the rate at which something changes. Instead of having to tell the tadpole where to be at each second, we are going to tell it how fast or slow it needs to move.
 - Let the youth figure out the answers whatever way they can. If they use some form of mathematics that's fine. You want them to develop their own system and then you can introduce the formality after they have tried making a rule on their own.
11. Have the youth share their solutions with the whole group.
12. Having the youth generalize their understanding, their findings, is an important part of what makes math a powerful tool. To support this process, ask youth how they would figure out the shortest path to the food if the screen was twice as long (You have a PondLife stage that is twice as long as the original in PondLife no rules.sim). Hopefully youth will double their original rates to make it 4 over and 1 down.
- What if PondLife was the size of the whiteboard or a movie screen? Youth might figure out the shortest path by dividing the height of the board by the width of the board. The point of suggesting the whiteboard, or a movie screen, is to show that calculating every step of the tadpole would not be an efficient way of calculating the ending position. It's much easier to figure out the path with a line, and then figure what each step will be. (For a whiteboard, keeping the tadpole in the top right and the food on the bottom left, the rule could be 15 over, and 1 down; essentially, the tadpole will have to take "wider" steps).
 - What if you move the food? How would you figure out how to get the tadpole there the fastest?
 - Depending on the youth's abilities and interest, you may want to start using coordinates for positions. What if the food is at (0,0) and the tadpole is at (10, 10), (5, 25), or (100, 10), how would you move the tadpole to get to the food quickly?
 - You are hinting at slope here—the inclination of the line connecting the tadpole to the food—and therefore the width or height of each step indicates the slope of the line. Unless the youth bring up the word slope,

³ Since the weeds are floating across the screen, you must take into account how they move and where they will be at each step of the way. It's best to eliminate this complexity from this stage of the activity. If the youth are more advanced, you can always bring the weeds back afterwards as a challenge.

there is no need to mention it, as it will raise more complexity for some youth.

 **Main Activity: Math Activity (option 2)**

Time:	20 minutes
Purpose:	Work with graphing some functions
Materials	<ul style="list-style-type: none">• Computers with Internet access• Graph paper• Activities (graph paper with “treasures” to hunt that you create)

To Do

1. Have youth try out the following game online at <http://funbasedlearning.com/algebra/graphing/lines/default.htm>.⁴
2. Put an example on the overhead where there are three gems on (0, 3) (3, 3) and (7, 3) for explicit teaching about equations. Ask youth how they would program a character to move to get all three gems. (They should say either move left or move right, depending on where the character starts). Now, show the youth that they can write an equation to represent the same path ($y = 3$).
3. Introduce another example, where the gems are lying on a linear graph ($y = x$, $y = 2x$, $y = 3x$) and repeat the activity. When they are finished suggesting a rule, bring back the idea of an equation to represent the same path. The equation will allow them to predict where the character would be after 10 steps, or 15, and so on. Explain how they can put the information into a table to show that all three points happen to be on the same graph.
4. Give each youth a graph paper with several “gems”.
5. Ask the youth to work in pairs to figure out which is the best way for a character to move in order to get all the gems.
6. When they are finished, give the youth a second sheet, which includes many more gems than can fit on one graph. With this sheet, they can try out several paths until they can get all the gems.
7. Have the youth share their strategies for the first two sheets. They are likely to come up with several combinations of lines.

⁴ This game is based on the original Green Globbs game for Macs.

 **Discussion/Reflection**

Time:	5 minutes
Purpose:	Review what rate and coordinate planes have to do with programming.
Materials	PondLife.sim

To Do

1. Ask the youth to tell you what steps in the design process they performed? (Research it, Build It, Test it).
2. Ask the youth to share what they learned this week that will help them create their games? (Answers will vary: math, role of coordinate grid, creation of rules; screen size and scale; xs and ys useful when planning things on a screen and figuring out where your characters are; if-then statements format; programming a small animation!)
3. If the youth didn't come up with everything, here are some of the key learning concepts that programmers use when creating computer games:
 - a. Location via coordinates
 - b. Motion, paths, rates, trajectories
 - c. Scale (when you asked them to predict the path for a large screen)
 - d. If-then statements (for everything that happens in a game)
 - e. Creating, testing, and changing rules

Week 3: Field Trip & Selecting Your Team

Summary

Schedule

Warm-Up	Review Design Process chart for game design.	10 min
Challenge	Choose a design team.	50 min
Discussion/Reflection	Youth present their storyboards and design teams.	10 min
Main Activity	Go on a site visit. Choose a design team.	70 min (or more)
Discussion/Reflection	Discuss math activity	10 min
Total Time		2 hr 20 min

★ Essential Questions

- What is programming?

Design Process Concepts Involved

- Research it



Materials

- Computers with Internet
- Projector
- Walk Through Careers in Game Design handout
- Game Design: Who is in charge? handout
- My Design Dream Team handout

Getting Ready

Overview

Youth explore the roles of game designers and visit Electronic Arts. The following weeks, youth will wear their Designer hats, they will be graphic artists, and they will program the game. They also will do a quality assurance process towards the end, to see if the games have any bugs that they have to fix.

Glossary

- **User testing or usability testing.** An early stage of game development that helps developers understand how well people can use some human-made object (such as a web page, a computer interface, a TV, a cell) for its intended purpose. Usability testing helps developers improve on previous designs. Imagine what wealth of information Apple had to design its new iPhone, based on feedback from testers using older cell phones to navigate the Web.
- **Quality assurance or software testing.** A later stage in game development, software testing is a process used to identify whether the thing that was designed and built actually does what it is supposed to do. Testers use the software in a variety of ways to make sure it is functioning as designed. Think about all those beta software packages: they are put out so that the users report inaccuracies and bugs.

Background

The Warm-Up and Challenge are designed to prepare the youth to visit a game company. Youth will research job titles and roles in a game company. We have also provided links to Electronic Arts, because it happens to offer good information online. You may choose to look for information for the site you visit, if different from that of Electronic Arts, but you can also use Electronic Arts's website to prepare youth to think about the types of careers available in gaming.



Gender Tips

Be sure to prep ICT professional around gendered language, stereotypes, etc. so that they share examples of both genders and a variety of races, working in ICT.

 **Warm-Up**

Time:	10 minutes
Purpose:	Stimulate youth thinking about how the design process relates to game creation.
Materials	<ul style="list-style-type: none">• Design Process chart

To Do

1. Ask youth:
 - a. Do you remember the design process?
 - b. What's the first step in the design process?
 - c. What's your favorite step?
2. Additional prompts to get youth to think about design at two levels (form and function):
 - a. What needs to be designed in the game?
 - b. How would you go about designing a game?

 **Challenge**

Time:	50 minutes
Purpose:	Choose a design team.
Materials	<ul style="list-style-type: none">• Internet access• My Design Dream Team handout• Walk Through Careers in Game Design handout

To Do

1. Set the stage: We are going to look at Electronic Arts website to learn about how gaming companies organize teams and develop games.
2. Ask youth about the types of professionals involved in game development: Many of you have elaborate plans for your games. Who has an idea of the types of professionals they'll need to work with to create their games?
3. Prompt youth to name a couple of job titles they know from IT professional visits: web developers software developers/programmers, graphic designers, etc.
4. Have youth use the Walk-Through Careers handout.
5. Have youth poke around under each main category (audio, design, etc). to find out the types of jobs in these categories, called Subtypes (Note: youth only need to go to the subtype page for each category to answer the questions below, not individual interviews).

Sample answers for Walk Through Careers in Game Design handout

Q. What department or person is in charge of user testing?

A. Designer

Q. Which department or person is in charge of quality assurance testing?

A. Production department or producer

Q. Do you think you need both user testing and quality assurance testing for your game design?

A. You want them to say yes! The reason they need both is that user testing shows how users will respond to the product (e.g., will they understand how to use it, will they like it). User testing is done early in the development process to determine what to build. Quality assurance tests the product to make sure there are no major bugs or broken parts of the game. This testing is done near the end

of development when the product is nearly finished.

Q. Which category pays the highest potential salary?

A. Programming, \$300K

Q. Which category pays the next highest salary?

A. There are two design and visual arts, \$200K

Q. What does a sound engineer do?

A. A sound engineer creates all the audible material in the game, except music. They generate the game's sound effects, both for environmental ambiance like wind, water, or dogs barking, and for events that happen in the world, like footsteps or car crashes.

6. Ask youth to do the following (if necessary, demonstrate):
 - a. List the tasks that need to be done. You can use the Game Design handout to help you think of the specific tasks that need to be done to create your game.
 - b. Now that you know what the audio, design, visual arts, production, and programming people do, use the My Design Dream Team sheet to put in pictures, names, titles, and tasks for your teammates. (Demonstrate filling out the sheet on the computer).
7. Youth should recognize need for programmers, designers, producers, audio engineers, and visual artists. They may have others as well, or specifics such as interaction designer or web designer.
8. Make sure that the youth place themselves in their design team. To guide them, ask them what type of work they see themselves doing best (programming, designing, marketing, sound, etc).
9. Their dream teams could be all women, all younger people, cheaper or more expensive personnel, or Electronic Arts employees who have the best experience in designing the kind of game that the youth might choose.

 **Discussion/Reflection**

Time:	10 minutes
Purpose:	Youth present their design teams.
Materials	Projector to connect to each machine

To Do

1. Use project to enable youth to share their My Design Dream Team lists.
2. Have the youth explain why they chose this particular group of people. What are the strengths of the team? Where do they (the youth) fit in and why? Expect the youth to be the designers, the programmers, and the graphic artists, at minimum.

 **Main Activity**

Time:	70 minutes
Purpose:	Make a site visit. Choose design team.
Materials	<ul style="list-style-type: none">• Internet access• Walk Through of Careers• My Design Dream Team handout• Design Tasks sheet

To Do

1. Go on Field Trip. See the IT Professional Field Trip Packet for details.
2. Ask your guide to meet IT professionals from all the categories in the Walk Through Careers website, as referenced in the handout. Also ask if there will be opportunities for the youth to experience what the IT professionals do (e.g., participate in user testing) rather than just listening to them talk about what they do.
3. Help the hosts at your Field Trip site make the visit interactive and hands-on for the youth.

Week 4: Principles of Game Design

Summary

Schedule

Warm-Up	Review design requirements and plan.	5 min
Challenge	Discuss what makes a good game.	1 hr 15 min
Main Activity	Do Stagecast Tutorial.	50 min
Discussion/Reflection	Review design requirements to help you design a game.	10 min
Total Time		2 hr 20 min

★ Essential Questions

- How do you decide what to make?
- What is programming?

Design Process Concepts Involved

- Research it



Materials

- Computer with Stagecast Creator Software
- Internet access
- Keeping Track & Rules You Learn handout
- Access to Pac-Man at http://www.thepcmanwebsite.com/media/pacman_flash/
- Fun & Simple handout
- ICT4me Unit 5 Starting Kit.sim
- Designer or Programmer that poster handout

Getting Ready

Overview

For the next few weeks, youth will wear Designer hats In the Warm Up and Challenge, and Designer and Programmer hats In the Main Activity and Discussion/Reflection. In the Warm Up and Challenge, youth will learn what the principles of game design that they can apply to designing their games. In the Main Activity and Discussion/Reflection, youth start learning Stagecast formally, through the tutorial.

Glossary

- **Maze.** A type of game that has a labyrinth-style structure, with many paths and dead-ends. User choices are restricted by the maze. Often the goal is to make it to the exit, overcoming challenges along the way (such as collecting coins or avoiding pursuers).
- **Adventures.** Some of the earliest computer games take the user on a quest or adventure. Along the way, they can pick up clues or treasures, and often has to accomplish additional tasks. User choices are restricted by the previous actions of the user.

Background

Do the Stagecast Tutorial.

Before this week starts, make sure you have already selected pairs of students that will work well together. It is up to you how you want to assign pairs (whether you choose them, or whether you listen to the youth's preferences). This will be the first week of paired collaboration. Pairs must have folders in their computers (or flash drives) to keep their sims and notes they will want to have for each stage and for the whole game.

Gender Tips

Consider gender dynamics when selecting pairs in the mixed gender setting. If you select a mixed gender pair, make sure that both students use the computer and other resources equally.

Make a poster or copy on chart paper the Designer or Programmer Hat handout. You'll need a Post-it note or some other way of signaling the task the youth will be engaged in. When the youth are working as designers, put the Post-it note on the Designer half of the poster. When the youth are working as programmers, make sure you switch the Post-it note to the programmer half.

Make sure that the Pac-Man game works properly on your computers. Review the following article about the game design principles involved in Pac-Man and why it's such a brilliant design:

- http://dukenkem.typepad.com/game_matters/2004/01/the_genius_of_p.HTML

For information on adventures review a few sites:

- Wikipedia http://en.wikipedia.org/wiki/Adventure_games
- Hitchhiker's Guide: <http://www.douglasadams.com/creations/infocom.php>

Warm-Up

Time:	5 minutes
Purpose:	Review the Design Requirements (from Week 1).
Materials	<ul style="list-style-type: none">• Designer or Programmer Hat handout

To Do

1. If you haven't already, assign partners so that youth can work in pairs.

Tech Tips

Pair programming: Give the first youth 10 minutes to be the “driver” of the computer (moving the mouse/typing on the keyboard), while his/her partner is the “navigator” (telling the driver what to do.) Then switch roles for the next 10 minutes, so that both youth get at least 10 minutes to be drivers and 10 minutes to be navigators.

Gender Tips

Youth will be in pairs for the duration of the unit, so select pairs carefully. Consider pairs of youth with compatible and complementary abilities, interests, strengths. To support mixed gender groups, emphasize pair programming to give both youth equal opportunities to program and design.

2. Ask youth whether they remember the kinds of jobs that people can have in a game design company.
3. Explain that they are working on a small budget for G-G D-zine, so they only have two jobs to accomplish in the next few weeks: Designer and Programmer. In the following two weeks they are going to wear their Designer Hats.
4. Review the Design Requirements with them. Make sure they are clear on the two components they will be building together: the whole game, and the pair-programmed stages.

 **Challenge**

Time: 75 minutes

Purpose: Analyze the design principles in Pac-Man.

Materials

- Computer with Stagecast Creator software
- Internet access
- Access to Eliot's Duck (a Stagecast game)
- Two Fun & Simple handouts (one for Pac-Man and one for Adventure games)
- Access to Pac-Man at:
http://www.thepacmanwebsite.com/media/pacman_flash/

To Do

1. Tell youth that they are going to analyze two games to see how some of the Principles of Game Design look in action. The reason for this activity is to empower them to create their own game with these principles in mind.
2. Give youth 5 minutes to play Pac-Man unrestricted.
3. Pass out the Fun & Simple handout to analyze Pac-Man. Youth work in pairs or as a whole group.
4. Discuss with the whole class the design principles relevant in Pac-Man. Ask them to tell you first what they think goes in each category, then offer some help.
 - a. Few characters and a main character that users can connect with
 - b. Fun, simple layout so that the player always knows where they are and where they're going
 - c. Clear goal, with obvious challenges and rewards
 - d. Simple way for user to play game: Rules are simple: move left-right-up-down, eat dots→gain points, eat power dots→become invincible, eat ghosts→gain points, meet ghosts without power dot→die, eat fruit→gain points.
 - e. Trade-offs: Every key decision the player makes has both a positive and negative effect.
 - f. What are the trade-offs for Pac-Man? Eating dots, the main goal of the game, doesn't give him too many points and slows him down (imperceptibly). Eating ghosts earns Pac-Man more points but distracts Pac-Man from main goal of eating dots. The more ghosts Pac-Man eats, the more points he earns, but pursuing ghosts takes time and leaves Pac-Man

vulnerable to resurrected ghosts. Eating fruit earns more points, but distracts from main goal.

g. Easy way for user to tell whether they are winning or losing.

10. Have youth analyze the adventure game Eliot's Duck in a similar way.

Answers to Fun & Simple chart for Pac-Man:

Principle of Game design	Aspect	Description
Main character that your users can connect with	Characters	<ul style="list-style-type: none"> Pac-Man, ghosts, dots, power dots, fruit.
Fun, simple layout so that the player always knows where they are and where they're going	Layout	<ul style="list-style-type: none"> Maze structure (user can go to some places, but not others). Start-up screen is different.
Clear goal	Goal	<ul style="list-style-type: none"> Clear screen of all dots (to go to next stage).
Simple way for user to interact with game	Controls	<ul style="list-style-type: none"> User moves Pac-Man with arrow keys.
Clear decision making points	What choices are available?	<ul style="list-style-type: none"> Maze limits on where the user can go. User chooses the direction to move.
Trade-offs: Every key decision the player makes has both a positive and negative side.	Distractions	<ul style="list-style-type: none"> Shiny objects (make Pac-Man invincible temporarily). Fruit (earns more points than dots).
	Rewards	<ul style="list-style-type: none"> You gain points for eating dots, power dots, ghosts, fruit. If you win several stages, you get another life.
	Challenges	<ul style="list-style-type: none"> Eating dots makes Pac-Man a little slower (than when he runs in a clear passageway). Ghosts are faster than Pac-Man.
	Punishments	<ul style="list-style-type: none"> Lose a life if you get eaten by a ghost.
Easy way for user to tell whether they are winning or losing.	Feedback	<ul style="list-style-type: none"> User can see the dots disappear. Score is displayed prominently. New screen appears when Pac-Man dies or moves to next stage.

Answers to Fun & Simple chart for Eliot's Duck:

Principle of Game Design	Aspect	Description
Main character that your users can connect with	Characters	<ul style="list-style-type: none"> Eliot, Drowning Girl, Person, Tree, Lake, Cat, Duck, Turtle, Signpost, Bridge, Houses, furniture, Inv Siding, Black, Wall, Mushroom.
Fun, simple layout so that the player always knows where they are and where they're going	Layout	<ul style="list-style-type: none"> Several screens. Some screens are forests or gardens. Two labyrinths.
Clear goal	Goal	<ul style="list-style-type: none"> Eliot has to find his duck. There are smaller goals: help the old man, help the drowning girl.
Simple way for user to interact with game	Controls	<ul style="list-style-type: none"> User moves Eliot with arrow keys.
Clear decision making points	What choices	<ul style="list-style-type: none"> User can go anywhere and is not restricted in

	are available?	movement. <ul style="list-style-type: none"> • User chooses to do or not to do the small goals (find the cats, rescue the drowning girl). • Game offers new choices if user makes the right choices.
Trade-offs: Every key decision the player makes has both a positive and negative side.	Distractions	<ul style="list-style-type: none"> • None?
	Rewards	<ul style="list-style-type: none"> • If Eliot helps his neighbors by doing the tasks they ask, he receives more clues that will eventually lead him to finding the duck.
	Challenges	<ul style="list-style-type: none"> • Each screen has a new task.
	Punishments	<ul style="list-style-type: none"> • None?
Easy way for user to tell whether they are winning or losing.	Feedback	<ul style="list-style-type: none"> • When Eliot completes a task, something new appears in the game (cats, doors, paths). • People in the game give Eliot clues and praise when he does the right thing.

11. Ask the youth why they analyzed these two games? (To get a sense of what makes a good game, and to learn how the design principles apply to many kinds of games).

12. Ask the youth to tell you what things are similar and what things are different between the two games. (You want them to see some of the elements described below about how mazes and adventure games are different and similar).

13. Create a chart that has the main design characteristics of these games. A few things that are the same: story or narrative can be part of both and the principles of game design apply to both games. The *main difference* between a maze (Pac-Man) and an adventure (Eliot's Duck) is whether user choices affect the game or not.

a. **mazes** (Pac-Man)

- i. User can always go back to where they were before, because their choices do not alter the game world.
- ii. Places a user can go are determined by the walls or layout of the game.
- iii. Mazes contain places where the user can go and places that are dead-ends. Some mazes are labyrinths, but not all. Some mazes have one pathway; some have multiple paths.
- iv. User may have to collect or manipulate objects.

b. **adventure** (Eliot's Duck)

- i. User choices affect the world and future choices they can make. For example, if they go to a door but haven't picked up a key, they can't open the door. But once they have the key, they can open the door.
- ii. Focus is on investigation and exploration of game environment.

- iii. There are clues along the way. User may interact with other people in game to get clues.
- iv. User may need to solve puzzles or go through mazes to get next clue.
- v. Player takes on role of main character.

14. (Optional) Review the chart of games and fun factors from week 1. Can youth see how these games also fit the design principles (some better than others)?

 **Main Activity**

Time:	50 minutes
Purpose:	Start learning Stagecast using the tutorial Review artwork created by the G-G D-zine art department
Materials	<ul style="list-style-type: none">• Computers with Stagecast Creator software• Access to the Stagecast Creator Tutorial (in Stagecast)• Rule notebooks for the students, based on Keeping Track of Rules You Learn handout

To Do

1. Have youth go through Tutorial sessions 1-4.
2. If there are youth who have trouble reading and get distracted easily, encourage them to read the blue text only or pair them with youth who have complementary abilities.
3. Have the youth *write down any new rules* that they learn in the tutorial. At a minimum, they should have a list of the rules they learned. Preferably, they should write down more information about the rules so they will remember the process of creating them.
4. Remind them to pay attention to the “if-then” rules, especially the ones that are highlighted in the tutorial. (They learned about if-then rules in week 2, with PondLife).
5. Naming of rules is important, so discuss naming with them after Tutorial 3. Ask the youth why it’s important to name rules (so that they can easily find the rules, or other programmers can find the rules easily). Ask the youth what names they can give to their rules. The best names are not the most creative ones but those that describe the action.
6. Tutorial 4 on the order of rules is *very* important, so make sure you stop and discuss what they have learned in Tutorial 4. If they don’t get this step, programming will be a huge challenge for them and for you. If necessary, they can replay Tutorial 4 later on.
7. In the last 5 minutes, have youth review the artwork that G-G D-Zine has already created for them to design and program with. Introduce the main character. (Have youth open ICT4me Unit 5 Starting Kit.sim. Ask them to explore the database of characters by hitting play and clicking start.

 **Discussion/Reflection**

Time: 10 minutes

Purpose: Discuss how rules will help you design a game.

Materials

- Job Description handout (from week 1)

- Charts created in this unit for Pac-Man and Eliot's Duck

To Do

1. What part of the Design Process did you work on this week? (Research it)
2. What hats did you wear this week? (Designer and programmer hats)
3. What did you learn this week that will help you design our games? (Principles of game design and Stagecast)
4. What are some of the principles of game design?
5. What rules did you learn to create in Stagecast?

Week 5: Designing the Big Game

Summary

Schedule

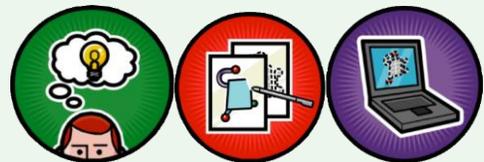
Warm-Up	Create chain stories to prepare for game design.	30 min
Challenge	Develop a design for the Big Game.	40 min
Main Activity	Map out the Big Game. Do Stagecast Tutorials 5-8.	60 min
Discussion/Reflection	Review pieces of the game youth accomplished.	10 min
Total Time		2 hr 20 min

Essential Questions

- What is programming?

Design Process Concepts Involved

- Brainstorm
- Sketch it
- Develop designs



Materials

- Computers with Stagecast Creator software
- Chart paper
- 5 x 7 large Post-its
- Tape
- Writing tools
- Rules notebook based on Keeping Track of Rules You Learn handout
- Charts from previous weeks
- Brainstorming the Big Game handout
- Map of the Big Game handout
- Paper
- ICT4me Unit 5 Starting Kit.sim

Getting Ready

Overview

Youth will wear their “Designer” hats to brainstorm and map out the big game. In the Main Activity and Discussion/Reflection, they do four more Stagecast tutorials.

Mapping out the big game will help them see the whole puzzle and determine the pieces that they need to build in the following weeks.

 **Tech Tips**

Keep this map out for the duration of the unit.

Glossary

- **Chain story.** A story written by several authors sequentially. When the first author is done, they pass the story to the second author, and so on. This process leads to unexpected, and often funny results.
- **Brainstorm.** Method by which a group generates many ideas to solve a problem. Groups aim to have an open and safe environment during brainstorming, avoiding making negative remarks about the ideas been generated. After the brainstorming stage, groups organize the ideas and may discard some depending on whether or not they are feasible within the constraints of the design requirements.

Background

Students need to develop a coherent story about the Big Game, whether it’s about a young student starting his/her first day of middle school, or whether it is Troya, the defender of the meek, in the land of the fruits, teaching users about smart behavior in middle school. *The important part is that youth agree on the major brush strokes of the story, because they will be creating stages that must fit into the story arc.* They must also agree on what constitutes winning the game—maybe passing through all the stages. Or maybe going to 5 out of 8 stages. Or maybe they have to collect all the signatures of teachers, or classmates in each of the stages. Steer the youth away from war and fighting games, *emphatically* if you must.

Resources on Game Design

- To learn about game design, here is a site to help you prepare: <http://www.etc.cmu.edu/curriculum/gamedesign/index.HTML>
- You will have a database of characters from ICT4me, but you may want to get more images.

- Look especially at Dungeons and Dragons: Eye of the Beholder and Pokemon. You may find other images that you might like to offer the youth as backgrounds for their stages. For more background stages, go to:
<http://www.vgmaps.com/Atlas/GBA/index.htm>

There are several examples from which you can learn about storyboarding:

- <http://www.uncc.edu/webcourse/sb/storyboard.htm>.
- <http://multimedia.journalism.berkeley.edu/tutorials/reporting/starttofinish/storyboarding/>

 Warm-Up

Time:	30 minutes
Purpose:	Create chain stories to prepare for brainstorming.
Materials	<ul style="list-style-type: none">• Story Starters handouts (Put one story starter at the top of a sheet of paper and make 1-2 copies of each.)• Paper and writing tools

To Do

1. Ask youth to review what they did the previous week, and how that helps them in their job of designing and creating a game. (They analyzed the two game structures they are going to use to come up with a game, and they learned how to make rules in Stagecast).
2. Explain the goals for the week. They are going to wear their designer hats to brainstorm the Big Game. What is the Big Game? It's the game that they have to put together. Each of the stages that they program will have to fit into this Big Game. Because their game will have to tell a story (about surviving middle school), they are going to try to tell a story together.
3. There are several ways to do group storytelling. You are going to help the youth think about how storytelling as a group can be done by agreeing on some rules and sharing information. But first, you will prepare the youth with a bit of chain story writing. You can do it with the whole class or in small groups.
4. Pick one of the starter sentences (or fragments) below or use your own. Write the sentence on a piece of paper, read it to the class, and then fold the paper down. The next person will write another sentence, fold the paper over, and then pass it along to the person on their left. The paper will go around the room until it returns to the first person. The last person is in charge of writing the closing sentence.

Story Starters

After I spilled chocolate all over my shirt, _____

Because they loved jumping, _____

Ever since Jonah started playing drums, _____

If you get an A in math, _____

One day I woke up and discovered I was invisible.

One day the President accidentally dialed my home number.

One day I ran into (*famous person*) in the supermarket.

5. When you are finished, open the paper and read the story.
6. Ask the youth: Is there a way of making this story better? (You are looking for a way to make the process of telling stories as a group better, not for a way to improve their particular story. Ask for suggestions. You can try the game again with a few of their suggestions, but *don't spend too much time* doing the Warm-Up.)
7. Explain that making a game together will be similar to telling a story together. One way of telling a good story is for everyone to agree on the main elements of the story, such as the main character and locations.

 **Challenge**

Time: 40 minutes

Purpose: Brainstorm what the whole game is going to be like.
Agree on the main character, place, and story of the game.

Materials

- Design Process chart
- Chart paper
- 5 x 7 Post-it notes
- Tape
- Brainstorming handout
- Chart about Middle School Issues from Week 1.

To Do

BRAINSTORMING 1

1. Remind the youth they are wearing their “Designer” hats this week.
2. Explain the rules for brainstorming:
 - a. No idea is a bad idea.
 - b. Don’t criticize other’s ideas.
 - c. Be creative.
3. Tell the youth: The process for brainstorming will be like the telling of a story—except they are all going to participate in throwing out ideas. Write down the youth’s ideas on chart paper.
4. In the same way that you gave them story starters in the Warm-Up, they will need a few tools to help them brainstorm ideas for the Big Game. Ask the youth to tell you what these tools might be, what they already know about the game:
 - a. The main character is a middle school student.
 - b. The student is dealing with issues common to youth in middle school (point to the chart about middle school Issues from week 1).
 - c. The game takes place in a middle school.

Teaching Tips

Save the chart with the story.

5. Use the Brainstorming the Big Game handout. Students can work in teams of four or as a whole group. If they work in teams, you’ll have an opportunity to regroup during the Main Activity, next.

6. Have youth come up with ideas for the game. The brainstorm is not completely open-ended because there are some requirements, but otherwise, help them be creative. You will have time to guide them to a more concrete design later.
7. You can write down the story as the youth start to agree on the main elements. Youth will be making more specific decisions in the Main Activity.
8. Now that they have a story that they agree on—and one which they will revisit later when they are putting everything together—point to the Middle School Issues chart from week 1.
9. Ask the youth to tell you how these issues fit in with the larger story. You are looking for broad strokes. Add whatever they say to the brainstorm chart.
10. Alternatively ask the youth to think about *which* of these issues would work well in their game. They may have to revise their story so that they are addressing these issues. Ask them to pick X issues to tackle in this game (X should be equal to or less than the number of pairs).

 **Tech Tips**

Facilitators have found that even though they had eight issues at the beginning, the stories converged. They recommend giving youth options and narrowing it down to two to three issues, with two pairs working on the same issue.



Main Activity

Time: 60 minutes

Purpose: Create a map of the big game.
Learning more about Stagecast.

Materials

- Design Process chart
- 5 x 7 Post-it notes
- Chart paper about Middle School Issues (from Week 1)
- Map of the Big Game handout
- Chart paper
- Tape
- Brainstorm story created during Challenge

To Do

BRAINSTORMING 2

1. Explain that for Stagecast, youth have to think about how to organize the game, so they have to make a map of the big game and the stages that they are going to program.
2. Give youth the map of the big game handout. In teams or pairs, give them 5 minutes to review the three maps and answer the questions.
3. Together, decide which of these maps makes sense for their game.
4. Draw a map of their big game on chart paper. Make sure there are an equal number of rooms/stages as pairs, and an equal number of pairs as issues.

Teaching Tips

Save the map of the big game.

5. Post the “issues” in the rooms or stages on the map.
6. Ask the youth to start telling a story about the game. This is similar to the chain story idea in the Warm-Up, but the process will be transparent. Interject comments and questions that will help the youth clarify their ideas. Write down the story on which they agree on chart paper.

Teaching Tips

Youth do not pick a name at this stage—that can take many moons and may be distracting from the design process. However, if the youth want to come up with a “temporary” name—what you will call the game within the group, a private name—you can give them 2 to 5 minutes.

LEARNING STAGECAST

1. In the remaining time, ask youth to do Stagecast Creator Tutorials 5-8. (If any youth forgot what they learned, they should go back and do Tutorials 1-4, or whatever portion of those that seem necessary).
2. Make sure the youth write down the rules in their RULES notebook. Tutorial 8 is VERY important. It helps the students learn how to troubleshoot their rules. If necessary, stop and work on the rules together as a whole group.

 **Discussion/Reflection**

Time:	10 minutes
Purpose:	Youth present their stage designs.
Materials	<ul style="list-style-type: none">• Design Process chart

To Do

1. What part of your job did you accomplish this week?
2. Which of the steps in the Design Process did we use?
3. What new Stagecast rules did you learn? Can you show us?

Week 6: Prototyping & Storyboarding

Summary

Schedule

Warm-Up	Select stages in pairs.	10 min
Challenge	Create prototypes for the individual stages.	60 min
Main Activity	Create storyboards and plan for programming.	60 min
Discussion/Reflection	Discuss what pieces of the game youth accomplished.	10 min
Total Time		2 hr 20 min

★ Essential Questions

- What is programming?

Design Process Concepts Involved

- Sketch it
- Develop designs



Materials

- How to Make Storyboards handout
- Brainstorm chart (from week 5)
- Map of the Big Game handout
- Job Description (from week 1)
- Chart paper
- 5 x 7 Post-it
- Writing tools
- Tape
- Rapid Prototype of Your Stage handout

Getting Ready

Overview

The Warm-Up and Challenge are *crucial* to the subsequent weeks. Youth should storyboard as much of their stages. Storyboarding is a rough sketch of the game. Storyboards are as sequential as comic strips: they show big changes between each scene. Each scene should contain as many technical details as possible and notes for programming, such as the following: main character starts here and the door to the next stage is over here; when the user eats a strawberry, they turn red; if they hit a wall, they bounce back; there will be five strawberry trees and two walls.

Glossary

- **Sketch.** A rough drawing representing the main features of an object or scene. This is not a carefully drawn image; sketches represent an idea broadly.
- **Rapid prototyping.** A prototype is a model of something that will be developed in the future. Rapid prototyping is a fast way to give the designer and client a visual idea of the final product. Rapid prototypes help design teams identify the major elements in a product without spending too much time or money.
- **Storyboard.** “Preliminary sketches of action in sequential order. The storyboard serves as the initial description of the user interface and is necessary in discussion and planning of the production.”⁵

Background

There are linear and nonlinear storyboards. The linear games have a sequence: first one thing happens, then another, and so on until the end of the game. The sequence is represented in a linear storyboard. Nonlinear games have more than one path: the start place may be the same, but then there are two or more places or things that happen next. The storyboard will require many more arrows for all the possible choices in the game.

By the end of week 6:

1. Youth have selected a topic for their stage (*from the chart in week 1*).
2. Remind them that they already have artwork and games styles to select from.
3. They need to tell as much of the story as possible in panels (e.g., where does the user start, what does the stage look like, what is the goal of this stage, and how does the user go about achieving it).

⁵ Resource for storyboards is based on Curtis, G. and Vertelney, L. (1990) *Storyboards and Sketch Prototypes for Rapid Interface Visualization, Tutorial 33, CHI '90*. Storyboard example available from <http://www.unnu.co.uk/blog/?p=74>.

4. Point out that they want to make games fun. If the user can achieve the goal of a stage really easily, it will not be a fun game. Encourage the youth to think about “rewards” and “punishments” for the user. If the point of the game is to find a key, then the user could be distracted by other rewarding objects (e.g., coins). If there are rewarding distractions, there might be interactions that make the user lose points. In Pac-Man, the goal of game is to eat all the dots. Ghosts take Pac-Man’s lives away. Shiny objects in the corners—distractions—allow Pac-Man to move faster and eat ghosts. They do not help the user win.
5. Make sure that by the end of this activity the youth create the Storyboards for their stages.

Tech Tips

Programmers have different ways of keeping track of their tasks. Youth should use their Design Notebooks to keep a detailed plan for programming, including the characters they need, alternate appearances, a rough idea of the stage layout, and a list of jobs that they will have to accomplish. Knowing what they need the character to do will help them focus when they are doing the Stagecast tutorial in the next activities.

 **Warm-Up**

Time:	10 minutes
Purpose:	Select stages in pairs.
Materials	<ul style="list-style-type: none">• Map of the Big Game chart with topics (from week 5)• Brainstorm chart• Job Description (from week 1)

To Do

1. Ask youth to remember where they are at in their job process, which pieces they have accomplished, and what they yet have to work on.
2. Explain that they will be wearing both the Designer and Programmer hats this week.
3. Find a way to distribute the stages among the pairs in an equitable, thoughtful way that is agreeable to all. (You can write the names of topics on folded sheets of paper and put them in a basket, having each pair pick one. Perhaps you can allow them to swap once if necessary).

 **Challenge**

Time:	60 minutes	
Purpose:	Rapid prototyping individual stages	
Materials	<ul style="list-style-type: none"> • Rapid Prototype of Your Stage handouts 	<ul style="list-style-type: none"> • About 10 copies of the grid sheets per team

To Do

RAPID PROTOTYPING

1. Rapid prototyping is a way of sketching the game. In rapid prototyping, you look at the brainstorm ideas and figure out how you can tell a story with different pieces. It's a way of analyzing the brainstorm and creating concrete proposals for a game.
2. Ask youth what tools they have for this stage of the design. (e.g., artwork, game structures, a story for the Big Game. All those aid in creating the rapid prototypes).
3. Explain the rules for rapid prototyping:
 - a. Don't invest too much time in the sketch.
 - b. You need to be willing to throw the sketch away.
 - c. Each rapid prototype sketch is one story.

Encourage youth to think of rapid prototyping as a type of brainstorm. *Rapid prototypes are not storyboards.* Those come after the story arc for the stages.

4. Ask the pairs to come up with an idea for what will happen on their stage. Give the youth several copies of the Rapid Prototype handout for drawing.
5. Give them 5 minutes to come up with a story and rapid prototype. Have them make two more rapid prototypes for their stage, giving them 5 minutes for each.
6. When they are finished, they should have three prototypes. Have them spend a few minutes reviewing which of the rapid prototypes they like the best. They can like different pieces from the three prototypes they created.
7. Ask them to create their final rapid prototype using whatever they liked from the previous three rapid prototypes, or to create something completely different.
8. Once youth have a rapid prototype, they should flesh out the story. Have them write down all the things they think they will need for the game. They will have more time to do this systematically, but catching their thoughts now will be important!
9. Ask each team to share their rapid prototype of the stage with the group.
 - a. Encourage constructive feedback between the pairs.

- b. Ask the youth to keep the big story in mind when reviewing each other's work. How does the stage design fit into the Big Game story?

 **Main Activity**

Time: 60 minutes

Purpose: Work through storyboarding.
Plan for programming.

Materials

- Rapid prototypes from Challenge
- How to Make Storyboards handout

To Do

1. Have each team write down the characters that it will need for its game.
2. Teams can add or remove characters from this list, especially as they work through the storyboarding.

STORYBOARDING

3. Give youth the How to Make Storyboards handout.
4. Explain to the youth that storyboarding is a way of organizing their thoughts about the stage.

 **Tech Tips**

Doing the storyboarding in their design notebook will also give students a plan to follow when they are programming, so they should write down important details that will help them program their game.

5. If youth need to add or remove characters from their list, they can do so anytime.

 **Tech Tips**

Tip: In gaming companies, there is a person who specializes in creating storyboards. In other fields, many people start their work with storyboards: film and TV directors, web designers, and visual artists.

6. Push youth for specificity on their storyboards. Help them write down what is important. (What is the user going to do? What is the user going to interact with? What challenges are present? What rules do you need to program)?

 **Discussion/Reflection**

Time: 10 minutes

Purpose: Discuss new pieces of the puzzle.

Materials

- Job Description (from week 1)
- Design Process chart

To Do

1. What part of your job did you accomplish this week?
2. What stages of the design process did they use this week? (Sketch it, and develop designs).

Week 7: First Week of Programming

Summary

Schedule

Warm-Up	Open a new Stagecast file and import all images.	25 min
Challenge	Organize rules and start programming.	45 min
Main Activity	Start programming the main character.	60 min
Discussion/Reflection	Share what you learned about organizing and programming.	10 min
Total Time		2 hr 20 min

Essential Questions

- How do you decide what to make?
- What is programming?

Design Process Concepts Involved

- Build it



Materials

- Computer with Stagecast Creator software
- ICT4me Unit 5 Starting Kit.sim

Getting Ready

Overview

Youth begin programming their games. They create a new project (sim), add stages and characters. Next, they write out the rules they want to program for their main character using a table that scaffolds and documents their work. They should keep this table of rules in their Design Notebooks and update it as they work on new characters or objects. During the Main Activity and Discussion/Reflection, youth begin programming the rules they wrote out in the table.

Glossary

- **User input.** Users can control characters in Stagecast with keystrokes or mouse clicks. In the Rule Maker window, youth can click on the Keyboard or the Mouse tools to add user input to their rules.
- **Rule grouping.** Stagecast rules can be put into groups to organize the order of firing. Grouping rules is also a way of keeping the rules organized and the Character window clean.

Background

The actual programming of the games has the potential of being messy, and the strategies offered in this week will help youth organize their work and their thinking.

Make sure you practice importing stages and characters into Stagecast. Read chapters 2, 3, and 6 of the *Creator's Guide* for full details. (You will find the *Creator's Guide* in the Stagecast folder once you install the software).

To add backgrounds, go to the Stages Drawer, double-click the desired stage, and look for the background variable. Click the small triangle to see the menu for Get Background. Stagecast Creator comes with some sample images, but you can also make your own or find images on the Internet. Background images can be in GIF or JPEG image format. You can set the Background Position variable to Tile, Center, or Scale your image. (See http://www.stagecast.com/cgi-bin/templator.cgi?PAGE=Shared/documentation/CREATOR_FAQ)

At least twice during the programming weeks, review the youth's stages. You will need to print a report of each game, and in it, provide suggestions to the youth regarding game design and rule making or rule fixing.

For every game, be sure to do the following:

1. Make and review the report.

2. Play the game and see which rules are troublesome. (Write down rules for youth to fix).
3. Give recommendations to youth on rule making and game design.

During programming, encourage youth to copy rules from previously designed games or each other. Also, help them to learn how to troubleshoot their rules. Have them go through tutorial 8 more than once if necessary. Make sure they know how to test a rule, and how to eliminate problems by looking at one aspect of the rule at a time. For example, does the stage look like the rule? If not, what can you do to change this? Do you need another rule? Maybe you need an i-don't-care box in the rule? Are the rules firing in order? Is there a rule that interferes with this one in the order? Can you reorganize your rules in a different way?

Warm-Up

Time:	25 minutes
Purpose:	Import stages and characters to new Stagecast file.
Materials	<ul style="list-style-type: none">• Computer with Stagecast Creator software• ICT4me Unit 5 Starting Kit.sim• List of characters for each pair

To Do

1. Ask youth to open Stagecast and select Create sim.
2. Have them save their project/sim as youth's name 1 + youth name 2 v01. Introduce the idea of version control. Start with version 01. When youth start the next week, they just change the name of the file to youth's name 1 + youth name 2 v02.
3. Have youth open ICT4me Unit 5 Starting Kit.sim as a *second* sim. You can have two sims open at the same time. From the starter kit, youth can drag over the stages and characters they need.
 - a. For stages, open the Stages tab, then click the Background tab. Youth will have the option to leave blank or to get one. If they click Get, they can use any of the background images in their database of images. Note that characters do *not* interact with stages—they are just a background image. Characters can only interact with other characters, so if they want “walls” for example, they need to create “wall” characters. If they want a stage from another sim, they can simply open second sim and then drag the stage from one sim to the other.
 - b. To import other characters, once they place a character on the screen (i.e., a green star), they click on it with the Image Editor (circle of colors tool). They can then “get” a new picture from their files. This will also be useful for sharing images in the future.
 - c. To have multiple images in one character, click New while editing one character, and then either draw the new one or “get” one, as in the step above.
 - d. Have the youth experiment with the edit features of the drawing tool—to create characters that look to the right, or look to the left, with “flip horizontal” or “rotate” features.
 - e. Throughout, ask youth to name their images so they can recognize them, both the stages and the characters. They should also name the different versions of their characters.

- f. Encourage the youth to share any other characters that they may want across the games.

Challenge

Time: 45 minutes

Purpose: Write down all the rules to be programmed.

Materials

- Computers with Stagecast Creator software
- Making Character Rules handout (Make 5 copies per team.)

To Do

- Using the Making Character Rules handout, ask youth to review all the rules they think they will use to program their main character. This is the planning phase for the Main Activity. What are all the rules they need?
 - Action rules without user input
 - Action rules with *user input* (key stroke/mouse click) for main character
 - Interaction with other characters
 - Interaction with objects
- Use the following table to walk the youth through their worksheet.

When youth are making their rules, they should be able to write out the rule in sentence form. It will help if you give them the if-then framework. Try out a few examples if necessary (although you may have already done this in week 2, it'll be a good refresher). Paying attention to the appearance and all the conditionals (ifs) will help youth determine what the problem is in case the rule doesn't work.

Rule Name	Move down
Describe what happens in the rule	When Jessi is holding the shovel, she can shovel snow.
Starting position and appearance	
Other characters involved in rule	Snowflake
User input	Right arrow
Ending position and appearance	
Write the rule (e.g., If...then...)	If there is a snowflake in the square to the right of Jessi, and if she is holding a shovel and when the users presses right arrow, then delete the snowflake and move Jessi one square to the right.
Rule grouping	Basic motion

3. If teams run out of ideas, have them compare with other teams to see whether they find more rules they were missing.
4. Remind the youth that programmers have to plan out their work before they start programming so that they can be more efficient. That's why they brainstorm, do a rapid prototype, gather images, organize them in their sims, and write down all the rules they can think of, before they start programming.
5. Once the youth have finished writing down the rules for the main character, have them start writing down the rules for secondary characters.

 **Main Activity**

Time:	60 minutes
Purpose:	Begin programming main character.
Materials	Computers with Stagecast Creator software

To Do

1. Ask the youth to use their Making Character Rules handouts to start programming the main character of their stages.
2. Walk around, make sure that the youth are writing down the names of their rules, and that they are keeping track of their rules on the rules table.
3. Copying is an important skill that they need to learn. If there are complicated things they would like their characters to do, and they have seen them in a game before, they can analyze the code from that game and copy the rules. Encourage them to use copying as a problem-solving strategy while programming. *See Page 70 in the Creator's Guide for more information on copying rules.*

 **Discussion/Reflection**

Time:	10 minutes
Purpose:	Share ideas about organizing rules.
Materials	None

To Do

1. Ask youth why they think organization is important to game design.

Many reasons are valid, among the ones you hope they will think about are: organizing rules allows you to see easily whether you've created all the rules you need; it helps you find rules easily; it allows you to find a programming bug easily; it allows you to understand your code after you finish a game. For game developers, organization helps them share their code easily with colleagues. Youth might want to share their code with friends; organized code makes sharing easier.

2. Ask the youth whether they can think of other times when organization may be important.

Examples: you might like a messy room, or at least you don't think about it. Perhaps it's your way of keeping your siblings or parents out. But imagine if one of your friends wanted to find your homework, or something you had asked them to find in your room. If it's messy, they will spend a long time trying to find it, and maybe they will give up. The same happens with programming. Sometimes, different programmers have to work on the same game or program. They need to be able to figure out how things are organized fast. Otherwise, lack of organization costs lots of money, and programmers can't understand the code or find the bug, they may even have to spend time rebuilding something that took a long time already.

3. Tell the youth that a programmer who uses has clear coding schemes and who writes organized code is a highly paid and coveted programmer. Being organized pays well in this field.

Week 8: Second Week of Programming

Summary

Schedule

Warm-Up	Do Stagecast Creator Tutorials 9-12.	30 min
Main Activity	Continue programming the game.	1 hr 40 min
Discussion/Reflection	Share insights in programming.	10 min
Total Time		2 hr 20 min

Essential Questions

- How do you decide what to make?
- What is programming?

Design Process Concepts Involved

- Build it



Materials

- Computer with Stagecast Creator software

Getting Ready

Overview

During the next few weeks, youth will be programming their games. Youth will find sounds they want to use for their game.

Glossary

- **Troubleshooting** is a form of problem solving. It is the systematic search for the source of a problem so that it can be solved. Troubleshooting is often a process of elimination—eliminating potential causes of a problem. Troubleshooting is used in many fields such as system administration and electronics.

Background

You can substitute any of the programming days with a site visit or a visit from an IT professional.

Youth will need a lot of support in organizing their rules and learning how to program things that are not included in the tutorials. You will need to keep reminding them that they only have 5 to 6 weeks to program, and that they can come back to a difficult rule if it's taking them too much time. In addition, you should encourage them to analyze and copy rules from other games to save on programming time. Copying is something that programmers and computer scientists do often: "If someone has already written code that I can use, I can learn from them."

Make sure you know where to find sounds and that you practice programming sounds in Stagecast. Review page 136 in chapter 7 of the *Creator's Guide* for an example. (You will find the *Creator's Guide* in the Stagecast folder once you install the software). Tutorial number 11 also explains how to program sounds.

You can find audio clips and sound effects in the Stagecast folder.

 **Warm-Up**

Time:	30 minutes
Purpose:	Do Stagecast Creator Tutorials 9-12.
Materials	<ul style="list-style-type: none">• Computer with Stagecast Creator software

To Do

1. Tutorial 9 will be important for youth, because it teaches them how to create rules for the user to control the main character. This tutorial also reminds the youth about stepping through the rule-making process in order.
2. Tutorial 10 walks the youth through creating new characters.
3. Tutorial 11 is to learn how to add sounds. It will be a good opportunity for some of the more advanced youth to add sounds or record their own voices for the games.
 - a. Ask youth whether they remember the job title of the person in a game design company that works with audio. (There are several: a sound engineer works with all sounds except music, a composer makes the music for the game). Explain that because they don't have a lot of time, they will be using sounds that other sound engineers have created. But, as designers of the games, they should be thinking about the kinds of sounds that might be useful for their games.
 - b. Ask youth to spend 2 to 3 minutes brainstorming the sounds they would like to have for their games
 - c. Have the youth explore the sounds in the Stagecast program, by opening up the Sounds folder in a second sim. They can import Stagecast sounds the same way they have imported images from other Stagecast sims.
 - d. Show them how to find sound files on their Macbooks, by clicking the Spotlight magnifying glass (top right of computer screen). They can type in the word sound or sounds and they will see many files (perhaps included in Microsoft Office installation, or some that the computer comes with).
4. Tutorial 12 is about putting stages and doors together. Since the youth are working with one stage at this point, it is not relevant, except that they will have to add a door to their stage at the end—and they should think about this design feature.

 **Main Activity**

Time:	1 hr 40 minutes
Purpose:	Continue programming the game.
Materials	<ul style="list-style-type: none">• Computers with Stagecast Creator software

To Do

1. As the youth continue programming, walk around to see how they are working together, monitor their progress, and note any challenges or solutions that you want to address with the whole group.
2. This task is difficult to plan, but there are a few things you should be looking for:
 - a. Help the youth focus on creating rules for one character at a time. As the youth create rules for each of their characters, remind them to check rule and character interactions. Sometimes, a rule they create for one character is in conflict with the rule they create for another character.
 - b. The games will naturally evolve as the youth develop their programming skills, especially when they encounter barriers to implementing a solution and develop new solutions. This is natural, and you should encourage it. Listen and be available to help if they need it.
 - c. When youth do change their original plans, encourage them to document the reasons for the changes and the additional rules they need to create.
 - d. Some youth will have an easier time writing rules and coming up with innovative solutions to problems. If a youth is not challenged enough, find ways in which she can help other teams from time to time. The same will be true for drawings or images: some youth will have more artistic talent—encourage them.
 - e. Youth should have something they can share with their peers by the end of this day.

 **Tech Tips**

Typical problems that occur with rules:

1. Order of rules. Have youth review Tutorial 8, which will help with most troubleshooting problems.
2. Too many spaces in rule. Rules ought to be minimally designed so, if youth want to do motion up, down, left, or right, they need to just make sure that the square immediately adjacent to the character is empty.

 **Discussion/Reflection**

Time:	10 minutes
Purpose:	Share ideas about programming.
Materials	None

To Do

1. As you walk around the room, take note of significant programming challenges or solutions. During the discussion/reflection time, ask the youth to share some of the challenges they discovered in the process of programming.
2. Ask youth where they are in the process of designing their stages. What part of the job description did they accomplish this week?
3. What parts of the Design Process did they do this week? (Build it).
4. How many more weeks do youth have before they should be done with their games?

Week 9: Third Week of Programming

Summary

Schedule

Warm-Up	Do Stagecast Creator Tutorials 13-15.	20 min
Challenge	Continue programming the game.	50 min
Main Activity	Bring in an IT professional.	60 min
Discussion/Reflection	Share insights in programming.	10 min
Total Time		2 hr 20 min

★ Essential Questions

- How do you decide what to make?
- What is programming?

Design Process Concepts Involved

- Build it



Materials

- Computer with Stagecast Creator software
- Internet access

Getting Ready

Overview

Youth are going to continue to program their stages. Youth should have a somewhat coherent stage by the end of this activity, in order to share with their peers next week. Youth give you their games so you can review them.

Glossary

- **Troubleshooting** is a form of problem solving. It is the systematic search for the source of a problem so that it can be solved. Troubleshooting is often a process of elimination—eliminating potential causes of a problem. Troubleshooting is used in many fields such as system administration and electronics.

Background

The actual programming of the games has the potential of being messy, and the strategies offered in this week will help youth organize their work and their thinking.

You can substitute any of the programming days with a site visit or visit from an IT professional.

Tutorials 13 to 15 are more advanced than the previous tutorials, so it will be up to you whether to let youth do the tutorials or continue programming. In general, being exposed to more rule possibilities may help the youth understand rule development better, and may even suggest solutions to some of their programming challenges.

Review the games. For every game be sure to do the following:

1. Make and review the report.
2. Play the game and see which rules are troublesome. (Write down rules for youth to fix).
3. Give recommendations to youth on rule making and game design.

 **Warm-Up**

Time:	20 minutes
Purpose:	Do Stagecast Creator Tutorials 13-15.
Materials	<ul style="list-style-type: none">• Computer with Stagecast Creator software

To Do

1. Tutorial 13 introduces randomness. Youth really like this particular rule.
2. Tutorial 14 is about working with two characters who are in the same place at the same time. This is an advanced skill. You can let some youth skip the tutorial.
3. Tutorial 15 is very useful, since the youth may create or use characters that are larger than one square.

 **Main Activity**

Time:	5 minutes
Purpose:	Continue programming the game.
Materials	<ul style="list-style-type: none">• Computers with Stagecast Creator software and youth games

To Do

1. Help youth copy rules and troubleshoot, and keep them on track with their designs.
2. Make sure the youth organize their rules and label them. Clearly identified rules make putting the games together much easier.
3. At the end of the day, tell youth that they should be about half-way finished with their programming. Explain that next week they'll be sharing the games with each other in order to receive feedback.

 **Discussion/Reflection**

Time:	10 minutes
Purpose:	Share ideas about programming.
Materials	None

To Do

1. As you walk around the room, take note of significant programming challenges or solutions. During the discussion/reflection time, ask the youth to share some of the challenges they discovered in the process of programming.
2. Ask youth where they are in the process of designing their stages. What part of the job description did they accomplish this week?
3. What parts of the Design Process did they do this week? (ICT4me).
4. How many more weeks do youth have before they should be done with their games?

Week 10: Testing Stages

Summary

Schedule

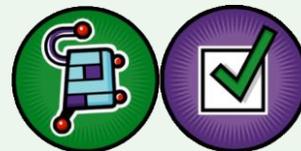
Warm-Up	Get stage ready for review.	20 min
Challenge	Review at least two other stages and provide feedback.	50 min
Main Activity	Debug games based on colleagues' feedback.	55 min
Discussion/Reflection	Share insights about programming.	15 min
Total Time		2 hr 20 min

★ Essential Questions

- How do you decide what to make?
- What is programming?

Design Process Concepts Involved

- Test it
- Build it



Materials

- Computer with Stagecast Creator software
- People Involved in Testing Games handout
- Zip/flash drives, access to server, or access to flickr.com
- Internet access
- Game Instructions handout
- Game Review handout

Getting Ready

Overview

Youth prepare to share their stages with other teams. They provide feedback to at least two other teams, and they revise their games based on peer feedback.

Enduring Understanding Assessed

Leadership involves teaching others new skills, communicating ideas to justify a position and convince others, and supporting a vision that may challenge the status quo (SCANS).

Glossary

- **Debug.** A method of systematically going through code to remove “bugs” or errors in a program. Debugging is broader than just testing rules to fix them. It is an organized and methodical way of checking every rule to find and fix the problem. Fortunately, Stagecast provides a Test button to help programmers debug their rules.
- **Test button.** To find out why rules are not working, you can test them. When you click on the Test button in the Rule Maker window, Stagecast will look at the situation on the stage and tell you which squares don’t match the rule. For more information, see the *Creators Guide* p. 83.
- **Feedback.** A process of sharing helpful suggestions and observations regarding a product or group performance. The goal is to improve the product or performance, and not attack a person. It is an opportunity for designers to get perspective on their designs; see if the design makes sense to a person not involved in the design itself.

Background

During this mini-performance task, youth write down instructions for playing their stages so that pairs of students can review their games. This is the first time they are formally asked to explain their game to others, other than you. Youth revise and refine the instructions after they debug their stages so they can be included in the larger game. You can review any Stagecast game to see the instructions for these games.

For the Challenge, youth review their stages and provide feedback to each other. Ideally, each computer has one game and a sheet of instructions on how to play the game (provided by the designer pair). The pairs walk around with the feedback forms to review other youth’s games. During the Main Activity, the youth review what others have said about their game and decide whether they are going to incorporate feedback or ignore comments. They also refine their games based on this experience.

You will have an opportunity to assess the youth's ability to communicate ideas clearly to each other through the task of writing out stage instructions. Clarity and organization of ideas is something that will be developing as they write and organize rules for their games.

 **Warm-Up**

Time: 20 minutes

Purpose: Prepare stages for peer review.

Materials

- Computer with Stagecast Creator software
- Game Instructions handout
- People Involved in Testing Games handout

To Do

1. Explain to youth that, because they have been working hard on their stages, it is time to step back and let new eyes look at the work, so that they can fix major problems before they complete their games.
2. Review the People Involved in Testing Games handout. Explain that professional game development companies usually have all of these people and processes involved in testing games.
3. Ask youth why testing is an important part of game development? (Possible answers: People pay a lot of money for games and expect them to be fun and worth the money. It makes the game better because you get feedback from testers who aren't biased.)
4. To prepare their work for others, youth need to do two things. First, they should write down instructions for their game.
 - a. Way to play the game? (How does the user interact with the stage (key strokes, mouse click)
 - b. Goal of the game. (What is the user trying to accomplish)?
 - c. Writing down the instructions is important. Youth will have to be concise without being confusing. A revised version of the instructions youth write out now will be used to convey to the users how to play the game, as in Jessi's Winter World. Game designers are not usually available to help the players play the game, so it is important to be clear.
5. Second, youth should write down one question they really want answered (e.g., Can someone look at the rule for the dog and help us figure out why it doesn't work? Is the game fun? How can we make our game more challenging?)

 **Challenge**

Time: 50 minutes

Purpose: Review and give feedback for two to three games.

Materials

- Computers with Stagecast Creator software and, youth's games
- Game Review handout (2 or 3 per team)

To Do

1. Have youth organize the computers around the room so that it is possible to walk around and review other games. Ask them to place the instructions for their game near their computer.
2. Explain the rules of the review process:
 - a. Give the kind of feedback that you would like to receive (remember, other teams are looking at your game).
 - b. Give feedback that is useful to the other team (think of how easy it will be if you receive ideas and suggestions that you can actually implement).
3. In 10 minutes intervals, have teams review the other teams' games. Each team should fill out one Game Review questionnaire for each game they review. They can leave the filled-out questionnaires, folded, under the computers (to ensure some privacy).
4. After three or four reviews, (leave about 10 to 15 minutes for this), ask the teams to return to their own game. Have them read the feedback from the other teams and consider which ideas they want to incorporate in their games, and which ideas they will not have time or cannot realize at this time (perhaps they can leave them for version 2.0, aka the next version). The frequency of the same suggestion may affect their decision to ignore or incorporate.
5. Each team should write down the things they are going to change or fix in their games before they leave for the day.

 **Tech Tips**

Using a written form for feedback on their game resembles the kind of feedback interaction that game designers go through. It also creates a space for reflection and gives youth an artifact to review later. You want them to practice analyzing feedback and making decisions based on evidence.

 **Main Activity**

Time:	5 minutes
Purpose:	Continue programming the game.
Materials	<ul style="list-style-type: none">• Computers with Stagecast Creator software

To Do

1. Youth pick up where they left off programming the previous week.
2. If there are any changes youth plan to make from the peer review, they should work on those as well. Keep in mind that the suggestions from the peers could be in graphics, audio, design, storyline, or rules. Give youth the space to work on any of those, as long as they don't get too side-tracked from finishing their game.
3. If they want, some youth can continue working on their games in off hours.

 **Discussion/Reflection**

Time:	15 minutes
Purpose:	Share ideas about programming.
Materials	None

To Do

1. What did you learn from other games?
2. Why would you want to do peer reviews?
 - a. Reviewing other's work may give you ideas for your own work.
 - b. Because you are working on one Big Game, keeping the lines of communication open is important, so you don't duplicate work and you can be consistent with your approach.
 - c. To help other programmers and share strategies.

Week 11: Fifth Week of Programming

Summary

Schedule

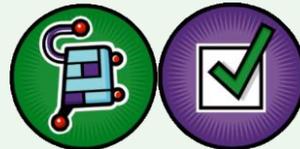
Main Activity	Continue programming the stages in the last week.	2 hrs
Discussion	Review programming successes.	20 min
Total Time		2 hrs 20 min

★ Essential Questions

- How do you decide what to make?
- What is programming?

Design Process Concepts Involved

- Test it
- Build it



Materials

- Computer with Stagecast Creator software
- Job Description handout (from week 1).
- Design Requirements handout (from week 1).

Getting Ready

Overview

Youth will have two more days to finish programming their stages. Make sure that they have addressed all the problems and bugs that were pointed out to them during the previous week. Make sure that the youth know this is the last week they have to program their stages before putting the whole game together.

Background

This is an opportunity for youth to compare the stages against the design requirements. Use the Design Requirements handout (from week 1) to check the requirements.

You can substitute any of the programming days with a site visit or visit from an IT professional.

Glossary

None

 **Main Activity**

Time:	2 hrs 10 minutes
Purpose:	Continue programming the game.
Materials	<ul style="list-style-type: none">• Computers with Stagecast Creator software

To Do

1. Have each team ensure that their stages meet the design requirements.
2. Help youth determine what could be improved.

 **Discussion/Reflection**

Time: 10 minutes

Purpose: Share ideas about programming.

Materials

- Design Requirements handout (from week 1)
- Job Description handout (from week 1)

To Do

1. As you walk around the room, make note of significant programming challenges or solutions. During the discussion/reflection time, ask the youth to share some of the challenges they discovered during the process of programming.
2. Ask youth to tell you how far they've gotten on their stages and how many of the job requirements they've met so far.

Week 12: Sixth Week of Programming

Summary

Schedule

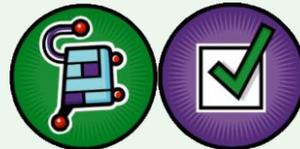
Main Activity	Continue programming the stages in the last week.	2 hrs
Discussion	Review programming successes.	20 min
Total Time		2 hrs 20 min

★ Essential Questions

- How do you decide what to make?
- What is programming?

Design Process Concepts Involved

- Test it
- Build it



Materials

- Screen shots of each game, preferably in color
- Computer with Stagecast Creator software
- Internet access

Getting Ready

Overview

This is the last week of programming for the youth.

Background

Youth need to be ready to hand over their stages to you next week, so keep reminding them of the “time budget” in the job description.

You can substitute any of the programming days with a site visit or visit from an IT professional.

Glossary

Collaboration means working jointly with others, in this case, programming a game. Collaboration includes cooperating and coordinating with other teams on the final product.

 **Main Activity**

Time:	2 hrs 10 minutes
Purpose:	Continue programming the game.
Materials	<ul style="list-style-type: none">• Computers with Stagecast Creator software

To Do

1. Have each team ensure that their stages meet the design requirements.
2. Youth should be putting the finishing touches on their games this week. They can add sound or help other teams if they are finished programming.

 **Discussion/Reflection**

Time: 10 minutes

Purpose: Share ideas about programming.

Materials

- Design Requirements handout (from week 1)
- Job Description handout (from week 1)

To Do

1. Ask youth to tell you how far they've gotten on their stages.
2. Ask youth how many of the job requirements they've met so far.

Week 13: Re-assembling Humpty Dumpty

Summary

Schedule

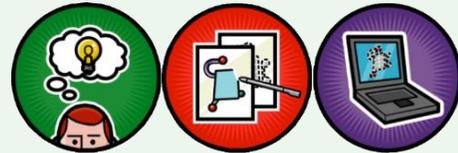
Challenge	Put Humpty Dumpty back together again. Create a New Map of the Big Game.	1 hr 10 min
Main Activity	Continue programming and incorporate final details.	1 hr 10 min
Total Time		2 hr 20 min

★ Essential Questions

- How do you decide what to make?
- What is programming?

Design Process Concepts Involved

- Brainstorm
- Sketch it
- Develop designs



Materials

- Screen shots of each game, preferably in color
- Computer with Stagecast Creator software
- Internet access
- Map of the Big Game chart (from week 5)

Getting Ready

Overview

This is the second performance task of the unit. Youth will collaborate in order to tell a story of how their stages fit together. This story may be different from the one that they started with, during the design of the game and the stages. The differences stem from all the independent programming and stage development in the intervening weeks. What's important is that youth now collaborate on creating a new story that enables them to piece the stages together in a coherent way. The collaboration is similar to putting Humpty Dumpty together again.

Youth should be ready to hand off their stages to you at the end of the week.

Enduring Understanding Assessed

Collaboration involves a strategy for dividing tasks associated with a solution into pieces that can be worked on individually and reassembling the work products into a cohesive whole to form the solution (NRC, SCANS).

Glossary

- **Follow Me character.** The Follow Me character is the character you automatically follow from stage to stage and never lose sight of. See Setting Global Variables in Creators Guide, p. 30.
- **Doors** are special characters. When a character moves on top of a door, it is transported to another place on the same stage or to another stage.

Background

After the youth have told the story of how they envision their stages fitting together, it is up to you to put all of their games physically in one sim. Read Chapter 2 of the *Stagecast Creator's Guide* thoroughly for this task.

Read about and practice how to import stages into a new file, make doors, and set the "follow me" character. Refer to the Creator's Guide for all these procedures. Also, help youth organize their rules, so that the rules do not interfere with each other's stages.

1. Import the stages into a new game.
2. Make doors.
3. Set the "Follow Me" character. Set the Follow Me character in the global follow me variable.
4. Transport to another stage rule, see *Creator's Guide*, p. 137.

5. Organize rules (all stage rules go in one folder, and so on).

Instructions for Putting Stages Together

1. Save all games on flash drive.
2. Open intro stage sim.
3. Click Open Second Sim.
4. Click Stages in both sims.
5. Drag the stage needed from the second sim into the Stages box on the intro stage sim.
6. Close the second sim.
7. In the Stages box of the intro stage, click the button that splits the screens on the top right (next to New).
8. Drag the second stage into the split screen
9. Place a Door in the intro stage.
10. Place the Exit in the second stage where you want the character to start the stage.
11. Once the character goes through the door, it will end up in the new second stage.
12. Changing characters:
 - a. Have main character enter door.
 - b. Once main character enters door, two characters will be on top of each other.
 - c. Create a rule to delete the character not needed for the stage.
13. Create a Follow Me rule:
 - a. Put the rule marker on the main character of the stage
 - b. Click on the yellow "And If" tab.
 - c. Click on the purple global circle.
 - d. Drag the "Follow Me" box into the And If portion of the rule.
 - e. Click on the yellow Actions tab.
 - f. Click on the purple global circle.
 - g. Drag the Follow Me box into the actions portion of the rule.
 - h. Drag the character into the blank box next to Put.
 - i. Name the rule Follow Me and the character's name.
 - j. Click Done.
14. Now the game will follow that character through the stages it's present in.
15. You need to create a Follow Me rule every time you want the game to follow a character into another stage.
16. Whenever a character is deleted a blank space appears in the Follow Me space.
17. You're making a rule that if there is not a character in the Follow Me space to put the character on the current stage in the Follow Me space.

 **Challenge**

Time:	1 hour 10 minutes	
Purpose:	How to put Humpty Dumpty back together again. Create a New Map of the Big Game	
Materials	<ul style="list-style-type: none"> • Map of the Big Game chart (from week 5) • Screen shots of each game, preferably in color 	<ul style="list-style-type: none"> • Big whiteboard space or wall, where the youth can re-assemble the game design as a whole

To Do

1. Elicit prior work. Ask youth if they remember the chain story they created a few weeks ago. (Students sat in teams. Each students wrote a sentence that built on what went before. The goal was to tell a story).
2. Ask youth if they've heard of Humpty Dumpty. If yes, have them tell the story and if not, share the rhyme with them. Explain that their big game is Humpty Dumpty: They started with an idea, then they worked separately in pairs, and now they have to put the pair work back together to fit into the big game again.

Humpty Dumpty⁶

Humpty Dumpty sat on a wall;
Humpty Dumpty had a great fall.
All the King's horses
And all the King's men
Couldn't put Humpty together again!

Sentado en un muro.
Humpty Dumpty
Se ha caído muy duro.
Todos los caballeros
Y jinetes del rey,
Fueron a levantarlo
Y no pudieron con él.

3. Explain that youth are now going to tell the story of their whole game, with all their stages together to see how they fit, and what, if anything, they need to change to better tell the story. This process is important, because youth only have one week left to program, and they need to coordinate the final product. They should be explicit with each other and with you, because you are the programmer who has to pull all the stages together and you need clear instructions.
4. Start by reviewing and discussing the Map of the Big Game chart (from week 5):
 - a. What was the story here?

⁶ Source: http://www.smart-central.com/humpty_dumpty.htm

- b. Where did you start?
 - c. What did you do to keep going or advance in the game?
 - d. How did the game end?
 - e. Each team tells what happens or how to play their stage of the game in the sequence on the wall.
5. Ask the youth whether the story still makes sense with their stages, as they create them. What is missing? Maybe it's time to revise the map of the big game.
6. Have youth create a NEW Map of the Big Game on chart paper or the big wide space. Ask questions once youth have assembled their pieces. This performance task is for them to run, once you've set up the activity.
 - a. Tell the story again, with any changes that are needed given your stages.
 - b. Where will the user start?
 - c. What will the user do next?
 - d. Does the game as a whole have a story you can follow?
 - e. Is it clear where players need to go next when they exit or complete a stage? What kind of path does the user follow?
 - f. Are the characters the same or different from stage to stage? If they are the same, are they predictable, in terms of what they can do, or do they change a lot from stage to stage (e.g., flying in one stage, but have to move on land in another)?
 - g. How much does the game as it stands now follow your original plan or story?
 - h. What do you need to do to change it? Whose stages might need to change? Are there connections between stages that need to change?

 **Main Activity**

Time:	1 hr 10 minutes
Purpose:	Continue programming the game.
Materials	<ul style="list-style-type: none">• Computers with Stagecast Creator software

To Do

1. Youth should incorporate the pieces they are missing in their games. For example, they may be missing doors.
2. Let youth also continue programming. Tell them that by the end of day, they should be ready to hand off their stages to you, for pulling the whole game together.

Checklist for Observers

When observing youth, be sure to check the following:

- Game should have a coherent narrative.
- Characters should show some continuity (look and feel), or signal/make clear when new appearance, powers, and abilities emerge.
- Stages need to relate to one another and follow pathways that allow movement from stage to stage so that players can visit every stage.

Weeks 14 and 15: Family Tech Night

Summary

Schedule

Warm-Up	Youth brainstorm projects and examples they want to share for FTN (Family Tech Night)	20 min
Main Activity	Youth complete project and develop presentations.	50 min
Discussion/Reflection	Youth discuss enduring understandings and reflect on what they've learned.	20 min
FTN Presentations	Youth and leaders host FTN, which may extend into the evening to accommodate adults' schedules.	2 hrs
Total Time		3 hr 30 min

Materials

- Charts and Design Notebooks completed during the unit
- Computers with Stagecast
- LCD projector
- Markers
- Display boards
- Posters
- Rulers
- Pencils
- Cameras
- Extension cords
- Tape
- Food
- Utensils
- Plates

Getting Ready

Overview

This is a chance for the youth to showcase to their family, peers, staff, and school community the work that they completed during the unit.

Background

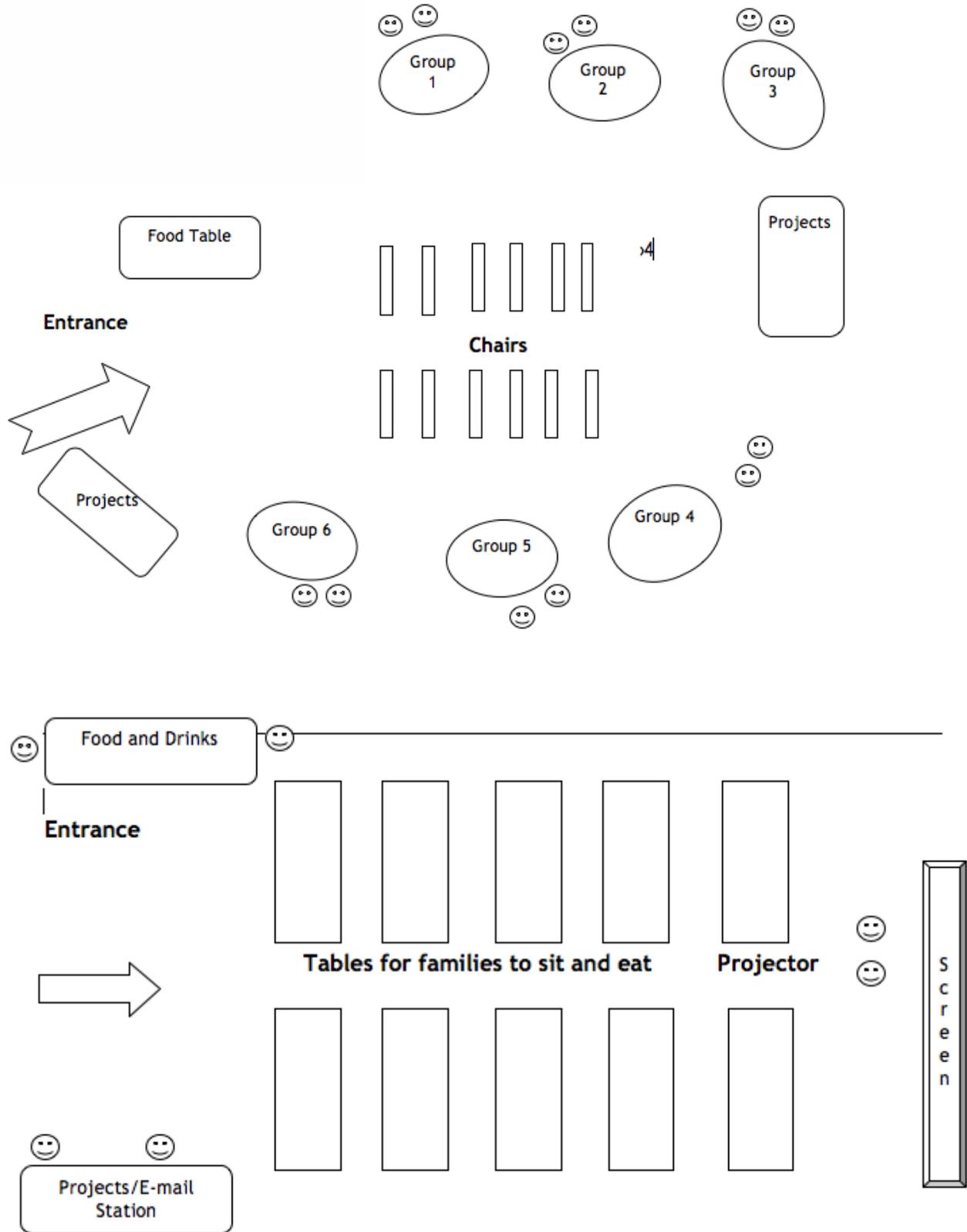
Refer back to the unit's lessons. Family Tech Night gives youth a chance to revisit lessons completed throughout the unit and use the information they learned when presenting their projects to the guest coming to Family Tech Night.

Tips

- Gather youth's charts and products created during the first 6 weeks of the unit to give you an idea of what is complete and what needs to be finished from the unit in order to display them for FTN.
- Make sure the youth take a leadership role in the planning and for the night itself. For example, have youth lead a short lesson with their guests and families. Make the night creative, interactive, and fun for the youth.
- Secure a space for FTN that will accommodate the displays and all the guests.
- Make sure computers have Stagecast and the youth's game. Also have a backup low-tech version in case the technology does not work or is slow.
- Since all youth worked on the Big Game together, youth may want to present different steps of the design and programming process (such as the Brainstorm of Issues chart, the Map of the Big Game, and the Rapid Prototypes). Visitors should be able to play all or parts of the Big Game. Make sure that there are enough computer stations with Stagecast and the youth's game.

Sample Layout

Layout can vary depending on number of students in your class. If you have extra students that are not assigned to a group, they can be at the project tables.



Warm-Up: Brainstorming a Plan for FTN

Time: 10 minutes

Purpose: Brainstorm projects that will be presented to parents/guardians/school staff on Family Tech Night.

Materials

- Large paper and markers
- Dry Erase Board and markers

To Do

1. You should have an idea of what projects youth want to show and the food for the evening.
2. Have youth come together in a large circle to brainstorm. A youth or facilitator can be the scribe.
3. Have youth think of the steps of the design process and products they want to show. Write down each one and tally how many times each one is suggested.
4. After a list is created, students will choose what they want to work on together in small groups. Make sure that each group has a task to complete.
5. You can also focus on a single project to show. Use this time to brainstorm tasks that need to be assigned to youth to complete for the event. Emphasize to the youth that they will show this project to their families and guest so that they know the level of work you expect.

Tech Tips

Keep in mind that you are facilitating the brainstorming session and you should already have some choices youth can select from.

6. Although only one project is highlighted during the night (in this case, the “Perfect Hangout”). It is important for students to know that all ICT4me work is important—for this reason all work is displayed.
7. It’s a good idea to ask Staff to visit those stations that aren’t being visited as often as other stations.

 **Main Activity: Creating a Plan for FTN**

Time: 50 minutes

Purpose: Make and create displays to show off the projects for Family Tech Night.

Materials

- Markers
- Paper (white/color)
- Note cards
- Scissors
- Pencils
- Display boards

To Do

1. Once a list of roles is created, share the list of roles with youth.
2. Assign roles to youth.
3. Students work in teams to determine what their stations will look like and how they will communicate knowledge of their project.
4. Have youth work with a facilitator to ensure appropriate language and organization to discuss the way the youth's projects reflect the learning from the unit.
5. Allow time for youth to practice their presentations.

 **Discussion/Reflection**

Time:	15 minutes
Purpose:	Reflect on what has been learned in the unit.
Materials	Presentations and project materials that will be used at FTN

To Do

1. Have youth gather in a circle and reflect on what they've learned during the unit. Remind youth of the essential questions and ask them to respond to these questions now that they are at the end of the unit. The youth's answers to these questions should be reflected in their presentations at FTN.

Essential questions for Unit 5

- How do you decide what to make?
- What is programming?

FTN Presentations

Time:	2 or more hours
Purpose:	Youth present what they've learned and designed
Materials	<ul style="list-style-type: none">• Presentations and project materials• Computers with Internet access• Food - It's a celebration!

To Do

1. Set up the stations as planned. Make sure each youth has a role and responsibility.
2. Provide food. It's a good idea to have tables for eating separate from the technology.
3. Back up plan if technology is not working. For example, for Unit 2 you may want to have a PowerPoint capture of the youths' blogs and clubhouses so you are not relying on the Internet. Have copies of the PowerPoint on a pen drive or multiple computers.
4. Encourage visitors to circulate. For example, you may want to create an information gathering game that encourages visitors to go to every station and ask the youth questions.
5. Have fun!

Activity Pages

Week 1

- Design Process chart
- Job Description
- Design Requirements
- Game Comparison Table
- Sample Answers for Game Comparison Table

Week 2

- Writing If-Then Statements
- Pondlife on a Grid
- Pondlife on a Grid - Possible Answers
- Longer Pond
- Blank Grid

Week 3

- Walk Through Careers in Game Design
- Game Design: Who's in charge?
- My Design Dream Team

Activity Pages Week 4

- Fun & Simple
- Keeping track of Rules You Learn
- Designer or Programmer Hat?

Activity Pages Week 5

- Brainstorming the Big Game
- Map of the Big Game

Activity Pages Week 6

- Rapid Prototype of Your Stage
- How to Make Storyboards

Activity Pages Week 7

- Making Character Rules

Activity Pages Week 10

- People Involved in Testing Games
- Game Instructions
- Game Review

G-G D-Zine Job Description

As **designer**, your job is to:

1. Design a whole game (with everybody).
2. Design a stage (with your partner).
3. Put everything back together (as a whole group).

As a **programmer**, your job is to:

1. Write out what rules you will need in your game.
2. Make a new appearance for the main character that is unique to your stage.
3. Import other characters you will need to your stage.
4. Make the rules so that your stage works.
5. Ensure that the rules have names and are organized.

Materials and tools you have for this job:

1. Users - students entering middle school
2. Stagecast - the programming software environment
3. Game design principles
4. Game structures to use (maze, adventure, both)
5. A main character graphic (see right)
6. A library of additional characters
7. A library of stages



Main Character

Time constraints:

1. Your preparation time is limited to 3 weeks.
2. Your design time is limited to 4 weeks.
3. Your programming budget is 6 weeks.
4. You will have 2 weeks to prepare and present your final product.

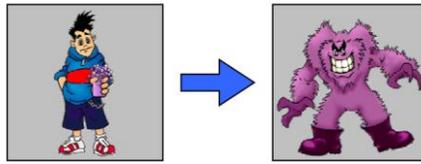
Whole game	<p>Your <i>plan</i> for <i>A Day in the Life of a Middle Schooler</i> game includes:</p> <ul style="list-style-type: none"> ⑥ A context or story (What is the point of this game? How do the stages fit together?) ⑥ A map of the game (How are the stages connected? What is the layout of each stage? How does the user get from the starting stage to the game stages? Where are the doors?) ⑥ Learning goals about middle school (What are you hoping to teach younger students about middle school?)
	<p>Your <i>game</i>, <i>A Day in the Life of a Middle Schooler</i>, includes:</p> <ul style="list-style-type: none"> ⑥ Starting stage ⑥ Doors to all the stages ⑥ Instructions about how to navigate and win the game ⑥ Clear way for user to know when she has finished the game ⑥ An “About” section that includes the description of the game and information about how and when it was created, who the authors are, and who the intended user is
Each stage	<p>Your <i>plan</i> for your stage includes:</p> <ul style="list-style-type: none"> ⑥ A gameplan (ways to win, strategies for winning, and ways to lose the game) ⑥ A storyboard (layout of the stage; critical changes when user moves, finishes game) ⑥ Learning goal (What are younger users going to learn when they play this stage?)
	<p>Your <i>stage</i> (developed in pairs) includes:</p> <ul style="list-style-type: none"> ⑥ One stage ⑥ One door to return to main page ⑥ The main character (controlled by user input) ⑥ Instructions about how to navigate and win the game ⑥ Two or more characters that the main character interacts with ⑥ A game (e.g., a maze or an adventure) ⑥ Clear way for user to know when she has finished the game ⑥ Something the user learns about middle school ⑥ Fun for users new to middle school or younger than middle school ⑥ Rules organized so that a friend (another software engineer) could easily find or add a rule
FTN presentations	<p>Your <i>presentation</i> at Family Tech Night should:</p> <ul style="list-style-type: none"> ⑥ Describe how your game satisfies the design requirements (given by the client, G-G D-zine) ⑥ Describe what users will learn about being a middle schooler in your game ⑥ A map of the big game, detailing the authors of each stage and a typical path the user may take ⑥ Storyboards that you used to create the game

Game Comparison Table				
Game	Goal & Story (What's the point of the game, and what's the story?)	Type of Game (Maze or puzzle, adventure)	How do you complete the game?	Fun for 4th-5th graders? Why?
Sokoban				
Jessi's Winter World				
Eliot's Duck				
GIAC				

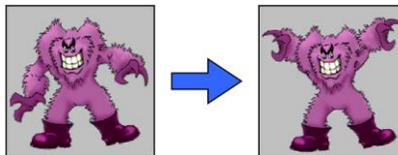
SAMPLE ANSWERS for Game Comparison Table

Game	Goal & Story (what's the point of the game, and what's the story?)	Type of game (Maze or puzzle, adventure)	How do you complete the game?	Fun for 4 th -5 th graders? Why?
Sokoban	You have to push certain objects in a certain way to get the rewards. Sometimes, you have to move different pieces around before you can get them to their destination.	This is a puzzle.	You win when you get all the objects into the right places. You lose when you get stuck.	The first ones are easy, but then you get stuck if you move the objects too close to the wall, and then you have to restart the game. So, I don't think it's very easy. But maybe it's fun.
Jessi's Winter World	Jessi is a girl in a wintery landscape. She has to find the hot chocolate hidden in one of the houses, and she has to plow snow, jump on snowflakes, and make a snowman.	There is a maze, and there is a puzzle when Jessi has to find the snowplow. And, it's like an adventure too.	You win when you get to the end and drink the hot chocolate. You lose when the snowballs get stuck.	Yes, because it gets harder and it's not too difficult. Yes, because it was fun for me too. Yes, because the rules are easy. No, because I got stuck, and it didn't work.
Eliot's Duck	Eliot loses his duck and asks his neighbors to help him, and they in turn ask him to do favors in exchange for information. And, at every stage, Eliot has to complete a challenge before he can move on.	It's an adventure, but there is also a maze, and there is a puzzle, and you have to collect things.	You win when you help everybody with their errands and they tell you where your duck is. You lose if you don't complete the tasks.	No, because it's too long. Yes, because there are a lot of things to do. There are many challenges. And then, you helped the man find his cats and saved a drowning girl.

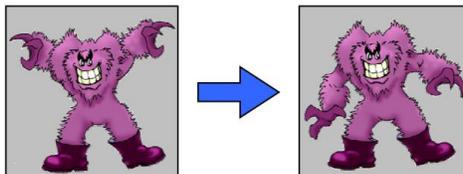
Writing If...Then Statements



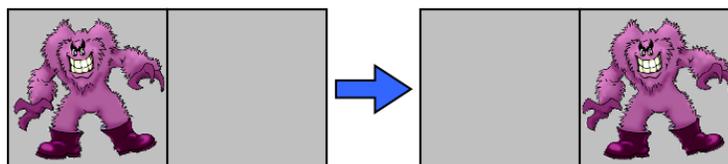
If the kid has a potion in his hand,
then, change him into a monster.



If _____
then, _____.

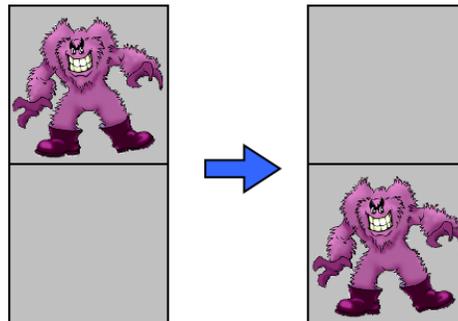


If _____
then, _____.

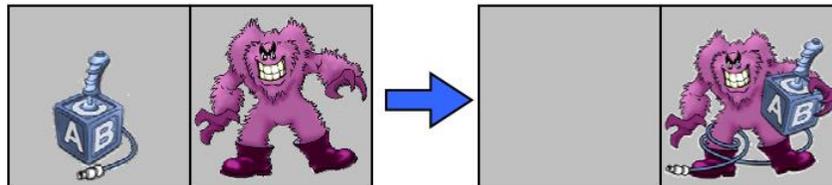


If _____
then, _____.

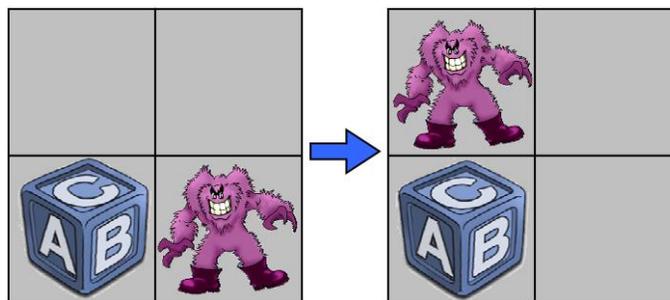
Build IT



If _____
then, _____.

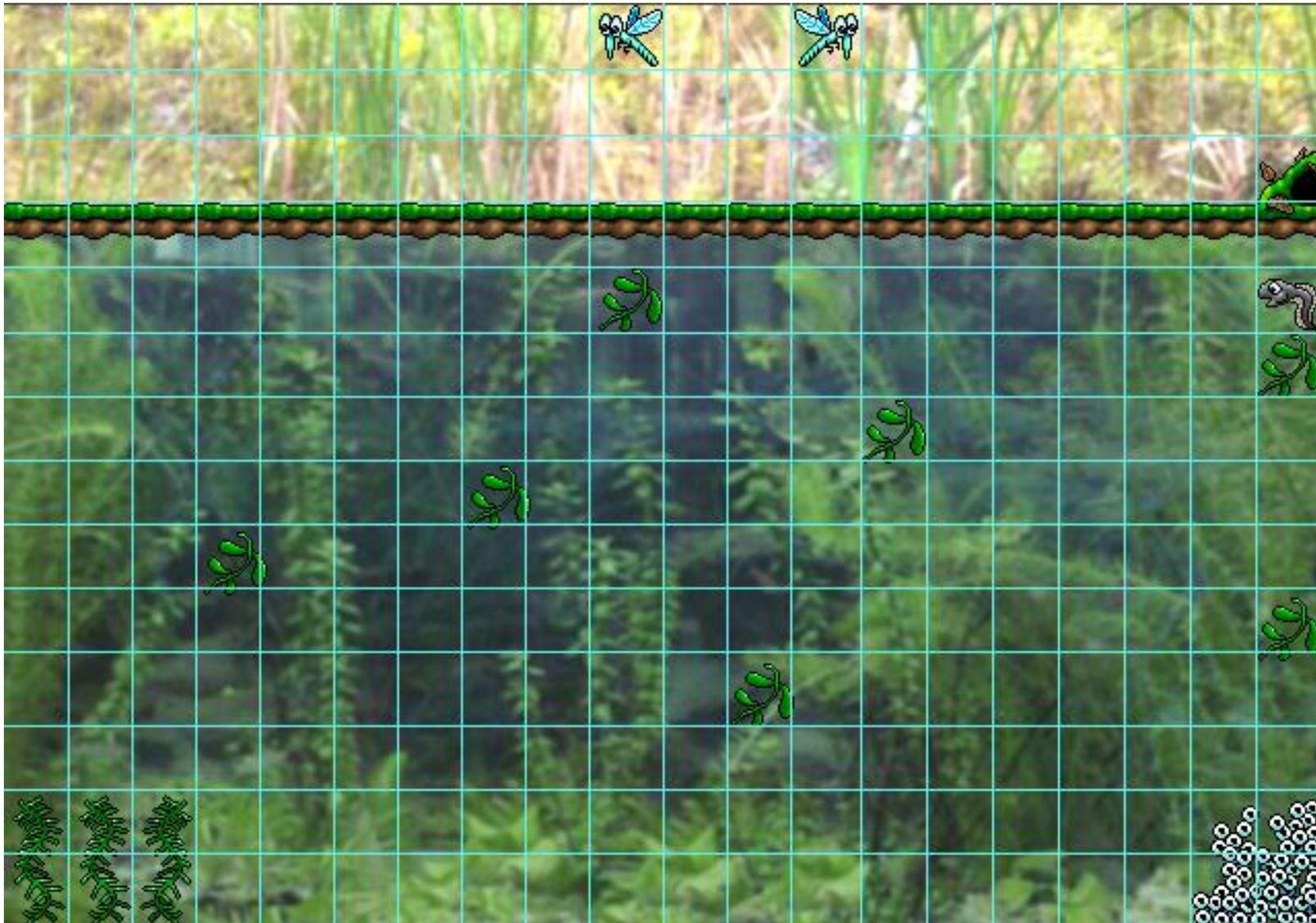


If _____
Then, _____.

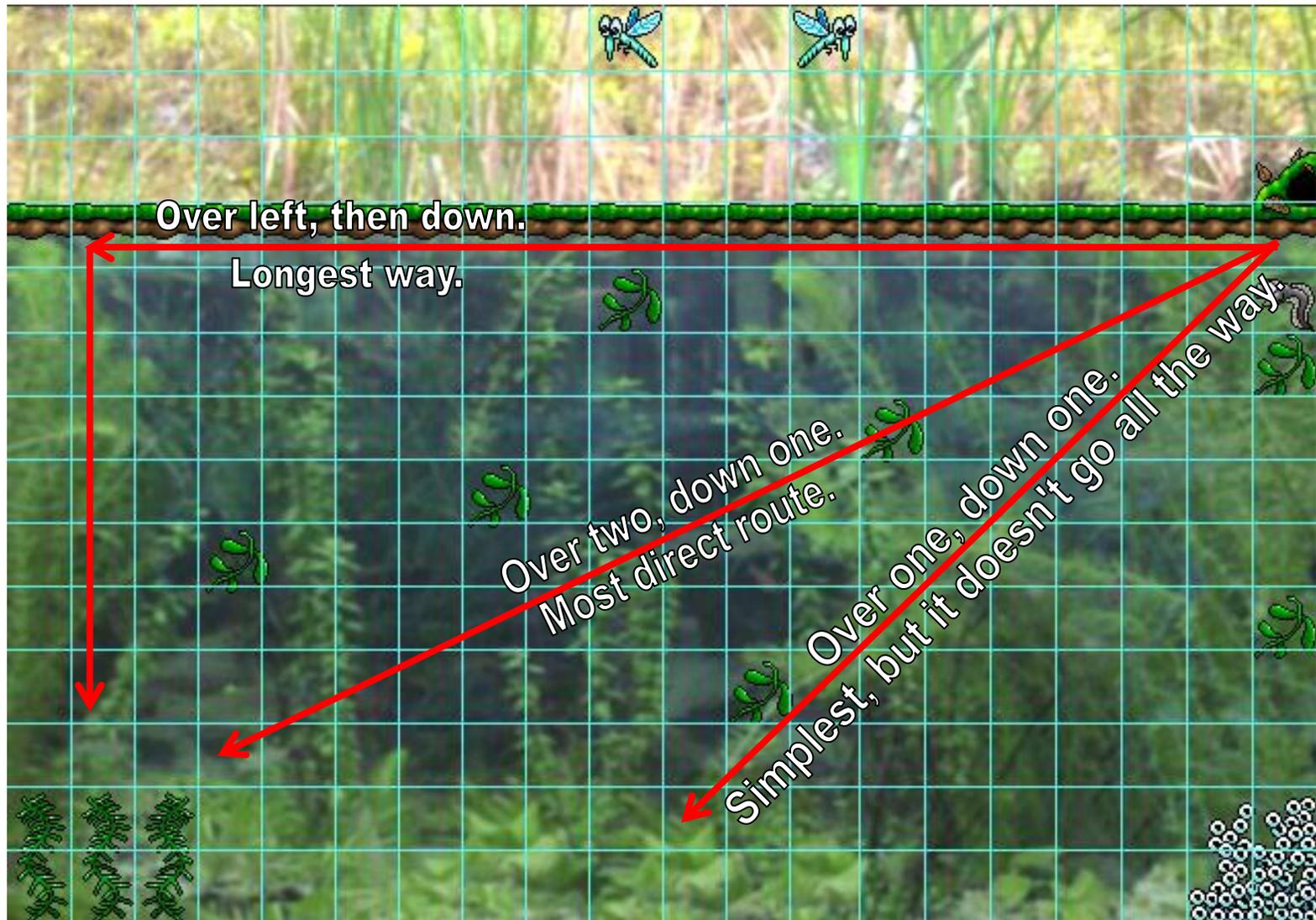


If _____
Then, _____.

PondLife on a Grid



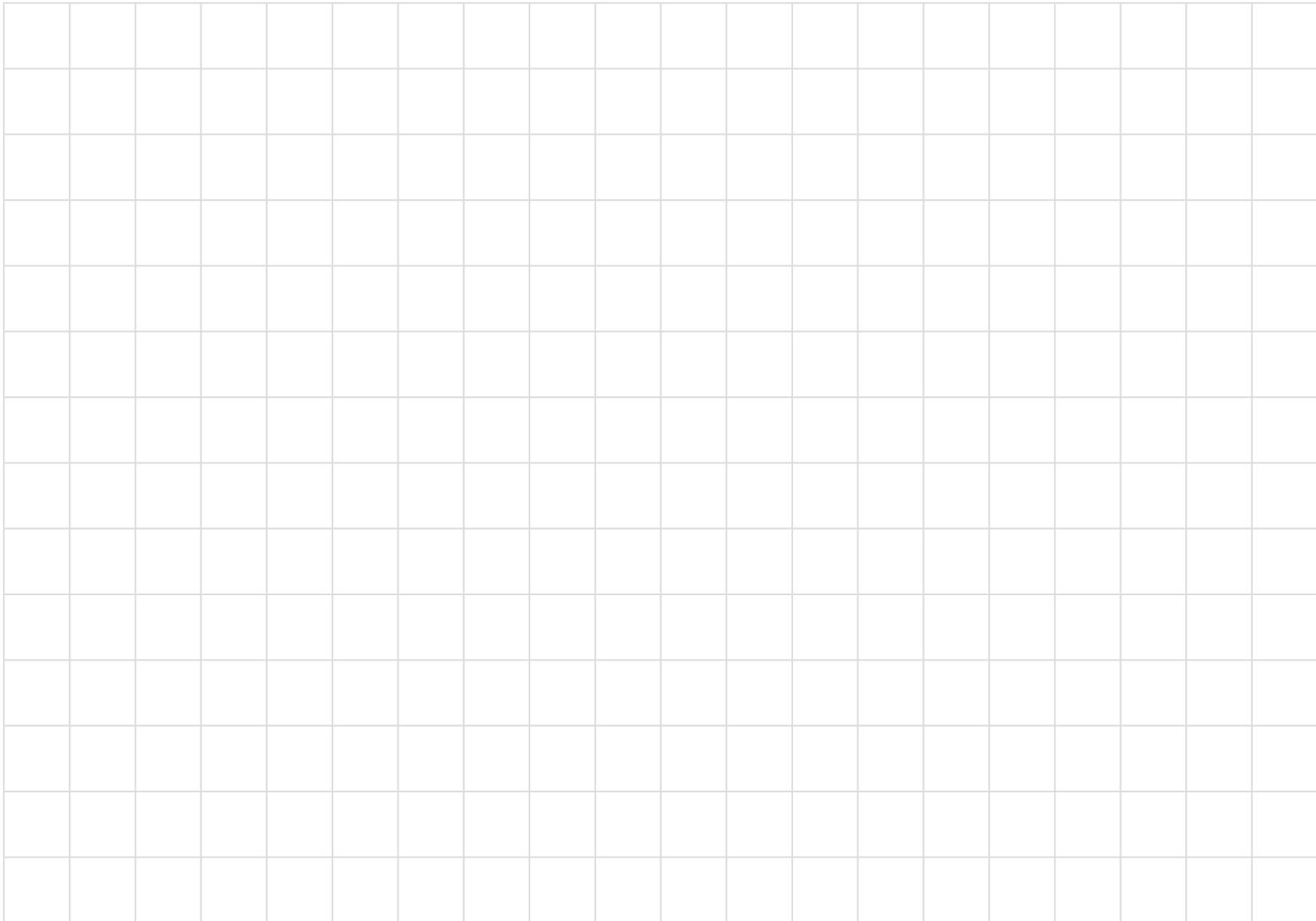
PondLife on a Grid - Possible Answers



Longer Pond



Blank Grid



Walk Through Careers in Game Design

1. Check out the careers in game design:

http://www.igda.org/breakingin/career_paths.htm

2. Click on Audio, Design, Production, Programming, and Visual Arts to answer these questions in preparation for selecting your team.

- A. What department or person is in charge of user play testing?

- B. Which department or person is in charge of quality assurance testing?

- C. Do you think you will need both user testing and quality assurance testing for your game design?

- D. Which job pays the highest potential salary?

- E. Which job pays the next highest?

- F. What does a sound engineer do?

3. Check out the following profiles:

- **Tricia Harris**, Web Designer & Writer
http://www.igda.org/breakingin/profile_tricia_harris.htm
- **Chris Degnan**, Lead Designer
http://www.igda.org/breakingin/profile_chris_degnan.htm
- **Darryl Duncan**, Composer/Musician
http://www.igda.org/breakingin/profile_darryl_duncan.htm
- **Seonaidh Davenport**, Program Manager
http://www.igda.org/breakingin/profile_seonaidh_davenport.htm
- **Douglas Noel**, QA Game Test Coordinator
http://www.igda.org/breakingin/profile_douglas_noel.htm
- **Genevieve Picard**, Software Developer
http://www.igda.org/breakingin/profile_genevieve_picard.htm
- **Tammy Yap**, Programmer
http://www.igda.org/breakingin/profile_tammy_yap.htm
- **Tito Pagan**, Senior Artist/Animator
http://www.igda.org/breakingin/profile_tito_pagan.htm

Game Design: Who's in charge?

Design Tasks	What department or person is in charge of this task?
 Define problem	market research designer
 Brainstorm	
 Sketch it	
 Research it	
 Develop designs	
 Create prototype	
 Build it	
 Test it	
 Use it	

My Design Dream Team

Pictures go here		
	Name:	
	Title:	
	Role/Job:	
	Name:	
	Title:	
	Role/Job:	
	Name:	
	Title:	
	Role/Job:	
	Name:	
	Title:	
	Role/Job:	
	Name:	
	Title:	
	Role/Job:	

Fun & Simple

These principles of game design will help you design your game. Fill out the following chart as you play the maze and the adventure games.

Principle of Game Design	Aspect	Description
Main character that your users can connect with	Characters	
Fun, simple layout so that the player always knows where they are and where they're going	Layout	
Clear goal	Goal	
Simple way for user to interact with game	Controls	
Clear decision making points	What choices are available?	
Trade-offs: Every key decision the player makes has both a positive and negative side.	Distractions	
	Rewards	
	Challenges	
	Punishments	
Easy way for user to tell whether they are winning or losing.	Feedback	

Keeping Track of Rules You Learn

Rule name			
Character name			
Describe what happens in rule			
Starting position & appearance			
User input (if any)			
Other characters involved in rule			
Ending position & appearance			
Write the rule (e.g., If...then...)			

Rule name			
Character name			
Describe what happens in rule			
Starting position & appearance			
User input (if any)			
Other characters involved in rule			
Ending position & appearance			
Write the rule (e.g., If...then...)			

Designer or Programmer Hat?



Designer

Programmer

Brainstorming the Big Game

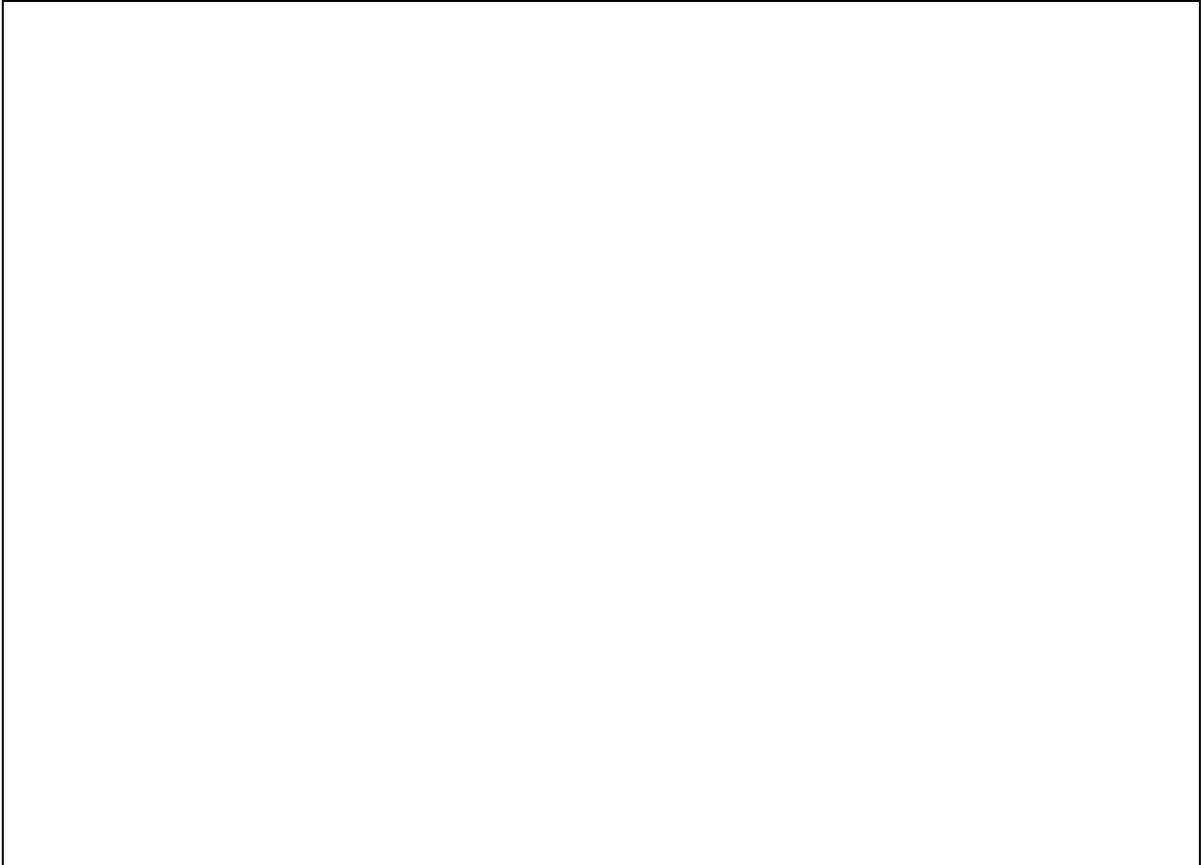
1. The first step in creating your game is to brainstorm ideas. Brainstorming is an exercise in pure creativity.
2. As you come up with ideas, share them with the group. Listen to what others are saying—listening can help you come up with new ideas, too.
3. When you are finished, consider the following

What makes a good first-person game?

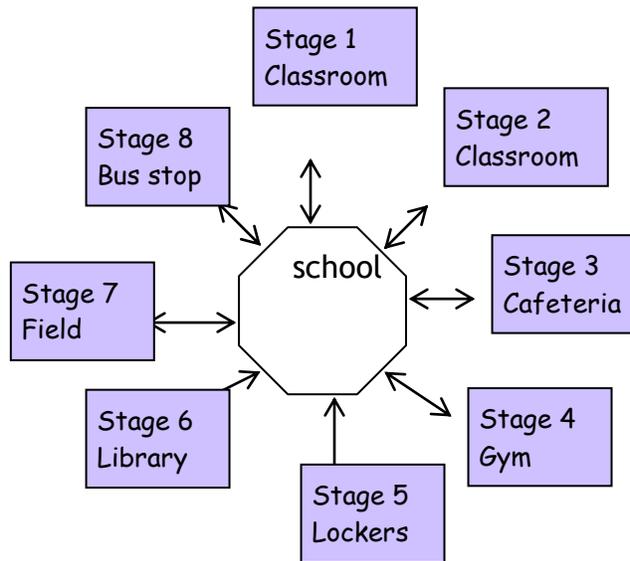
- Every game has a **main character**. The user will control the actions of this main character.
- Every game happens in a specific **location**, whether imaginary or real. Many games have rooms or stages. (Remember Jessie's Winter World?) You need to decide on a place or places for your game.
- Every game has a **goal**. The main character has to solve a need, a desire, or a problem.
- Every game has **actions**. The central character does something about the need, desire, or problem. This creates a result. The result is the goal of your game. You need to define the tasks before you start designing and programming.
- Every game has **challenges**. Games would not be fun if the goal were easy to achieve. You will have to think of ways to make the game difficult and fun for the user.
- Every game can be **won or lost**. Think about what constitutes winning your game (maybe collecting objects, getting points, eliminating the competition).

3. When you are done brainstorming, write down the whole group's idea for the game:

- a. What is the point of your game?
- b. What are the challenges for the main character?
- c. How will the main character go about winning the game?
- d. How does the game end?

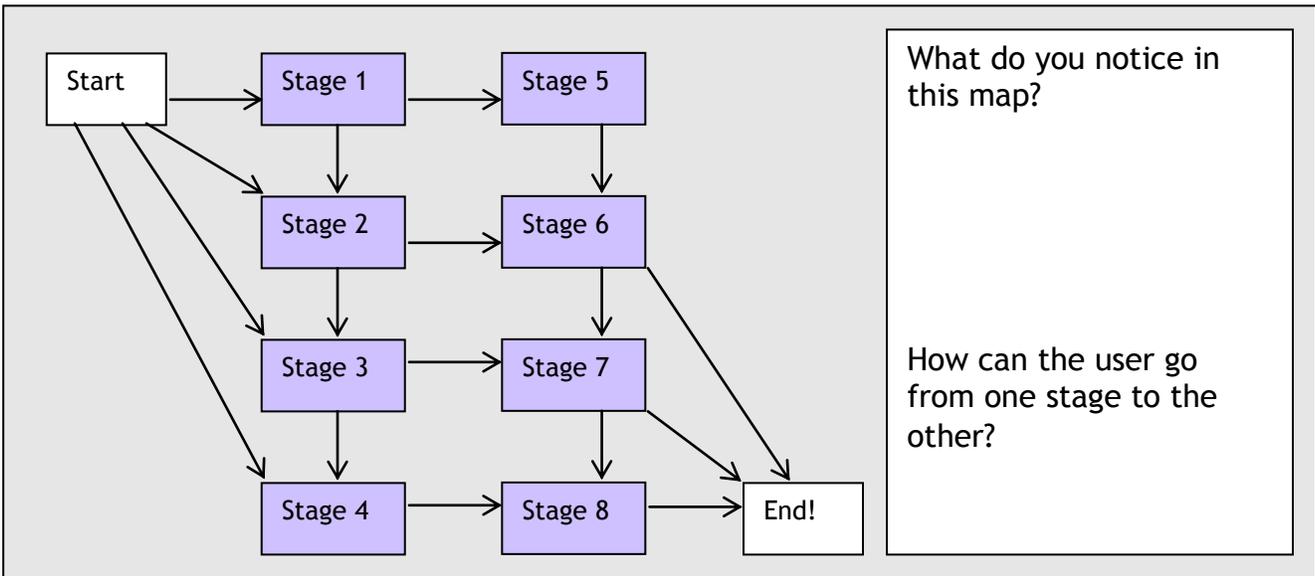
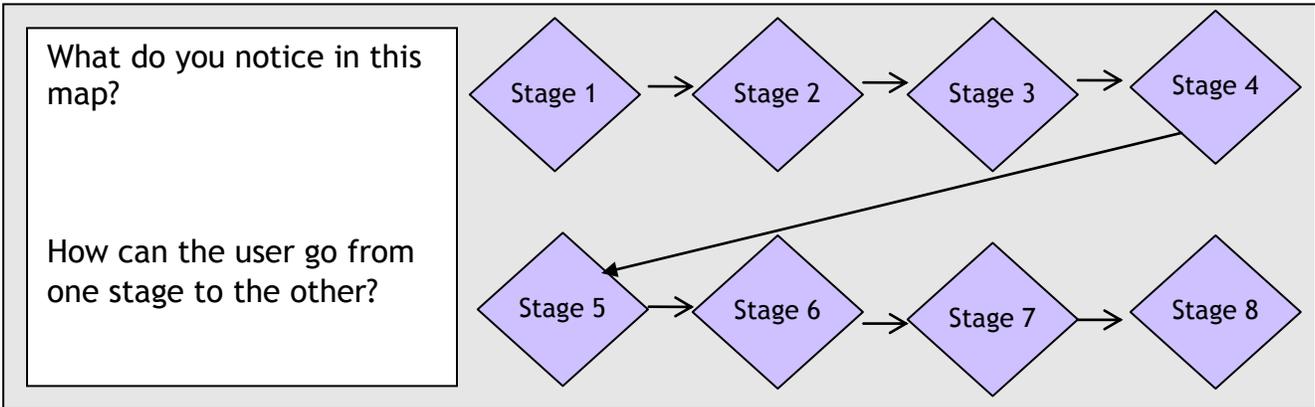


Map of the Big Game



What do you notice in this map?

How can the user go from one stage to the other?



Rapid Prototype of Your Stage

Project Title _____

Use the grids to sketch rapid prototypes of your game.

Include information about the location, the characters, the objects, and the actions taking place in each scene.

Rapid Prototype 1 Location: _____



What does the player do first? Second? Third?

The image shows a large rectangular frame with a thick black border. At the top of the frame is a horizontal grey bar. Below this bar, the frame is divided into two main sections. On the left is a grid of 20 columns and 20 rows of small squares. On the right is a large, empty rectangular box with a thin black border.

How to Make Storyboards

1. Create a sketch to represent the opening scene of the stage. Underneath the drawing explain what happens in the interaction. Write about:

Visual cues—what the user can see

Audible cues—what the user can hear

Tactile cues—what the user can touch

User input—how the user communicates with the computer

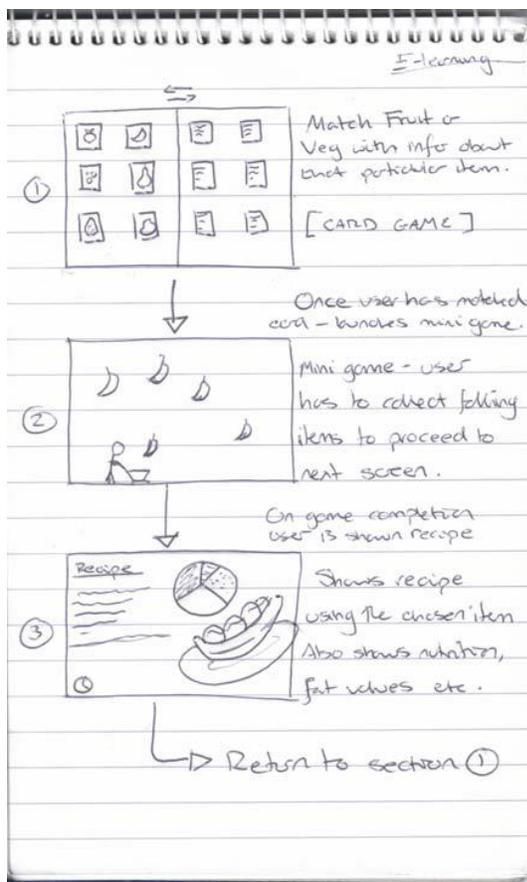
Computer output—how the computer responds to the user

Rules—what rules do you have to create to perform this task

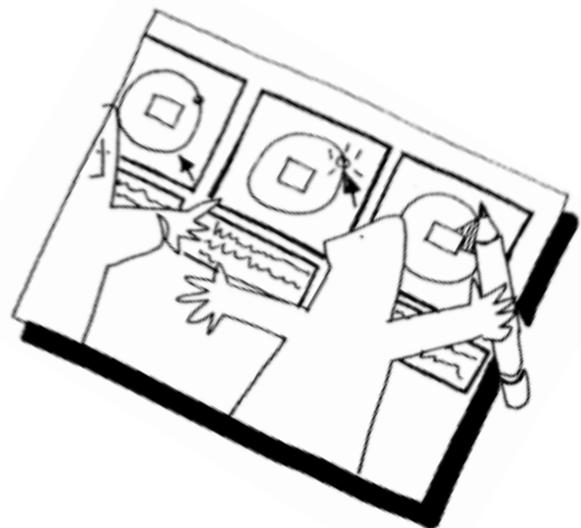
2. Describe interaction details and emotional responses verbally when no visual representation is effective.

Don't worry too much about the making sketches perfect; just use them to get the main idea across.

3. Do another sketch for what happens next, until you get to the end of your game.



Here are examples of a storyboards. Notice that the drawings are not really detailed. But both the sketches and the explanations help you see what this game is about.



Making Character Rules																
Rule Name																
Describe what happens in the rule																
Starting position and appearance																
Other characters involved in rule																
User input																
Ending position and appearance																
Write the rule (e.g., If...then...)																
Rule grouping																

People Involved in Testing Games

Here are some people involved in testing games. Some of their work sounds very complicated, but once you do it yourself, you'll see how easy it is once you get started.

Test Expert

The test expert is the person that owns the process and is responsible for gathering, analyzing, and effectively communicating the information. She should have strong game design, statistical analysis, and test development knowledge.

Internal Customer / Stakeholder

The internal customer is the person who requested the information (about testing the game) and will usually be responsible for developing a portion of the product or providing feedback on it. In some cases, soliciting stakeholder feedback can be used to bring in internal customer perspectives on divisive issues.

Play-Testers (Target Audience)

The play-testers are the carefully selected group that represents the target audience. Selecting play testers for the target audience is an important part of the process.

Play Test-Coordinator

The play-test coordinator is responsible for planning, organizing, and managing the play-test sessions according to the test expert's requirements. This includes everything from finding the play-testers all the way to ensuring they receive compensation.

Developers

The developers design and implement the tools and technology that allow for the efficient gathering and analyzing of all this information.

The Test Processes And Reports

What questions would you like answered about the product? Many developers want to know how customers are using their products and what they like or don't like about them.

Extracted from:

http://www.gamasutra.com/view/feature/1546/tracking_player_feedback_to_.php

Game Instructions

Team: _____

a. What is the way to play the game?

b. What is the goal of the game?

Game Review

When filling out this form, make sure your comments are helpful to the other programmers. Explain your reasons for each comment and suggestion.

Stage reviewed: _____

1. Did the instructions make sense with the game? If not, explain.
2. Comments about the graphics:
3. Comments about the stage:
4. Comments about the audio:
5. Comments about the rules:
6. What do you think this stage teaches 5th graders?
7. How can the programmers improve this game? (Give concrete answers).

Unit 5: Substituting Scratch for Stagecast

When we wrote the curriculum in 2005, Scratch didn't exist. Since then, Scratch has blossomed and there are a lot of resources and tutorials in addition to a large user community. If you are already using Stagecast or have access to a copy, you can continue using it. Below we compare Stagecast and Scratch to give you an overview of what each has to offer. For directions on how to integrate Scratch into each week of Unit 5, please follow the directions below in the Scratch Week by Week section.

Comparing Stagecast and Scratch

	Stagecast Creator 2	Scratch
Website	http://www.stagecast.com	http://scratch.mit.edu/
Support	See website for updates. It is possible that the Stagecast website will no longer be available.	Community-resource support
Cost	No new versions for sale.	free
Software	Lives on your computer. No new versions.	Can download software to computer, and can also use online.
Materials	Book and pdf with curriculum ideas; Online community of users; Tutorials and sample games to program	Huge online community, offering tutorials, information for parents and educators, tutorials, examples, etc.
Examples	Good game examples, made by publisher and other curriculum developers; strong educative value.	Search site to find examples from community, of varying degrees of sophistication; nor particularly educative.
Programming	Object-oriented programming. Stagecast is a visual programming environment. Users use the location of sprites (characters) on the screen to create rules. Whatever situation is present on screen will determine how the rule is made. If / then framework applies to all	Object-oriented programming. While Scratch is also visual programming, language is more prominent than in Stagecast – but most codes are already provided. Users are scaffolded in using the language-based commands with “fill in the blanks” scripts. Not all rules involve using an If /

	the rules. Order of rules is important.	then framework. Order of rules is important.
Sharing	Online. Users have to download the player (free) to use a Stagecast program.	Online.
Putting multiple Stages / Levels together	Requires a little help from instructor, but it's easily done, if following unit 5 instructions.	Not tested. It's possible. Instructor would need to find out from the Scratch tutorials and community.
Programming by	Rules	Scripts
Characters	Characters / Sprites	Sprites
Navigation to next stage / level	Door	Next Level Script
Background Stage	Stage	Level
How to start	Green Arrow	Green Flag

Week by Week Recommendations

Week 1: Programming vs. Playing Game

Challenge: Find three or four Scratch games for youth to play. Consider types of games, sophistication, leveling, and appropriateness in your choices. In particular, we want to provide two types of game situations for youth to explore: mazes and adventure games.

- Arcade: <http://scratch.mit.edu/projects/909079/>
- Puzzle: <http://scratch.mit.edu/projects/3124143/>
- 3D Maze: <http://scratch.mit.edu/projects/1652740/>
- 2D Maze: <http://scratch.mit.edu/projects/507/>

For the rest of the references to Stagecast substitute Scratch. For example, “Introduce Stagecast software” becomes “Introduce Scratch software”.

Main Activity: Use one of the games listed above in Week 1 Challenge, a game you found, or create one yourself. Start with games that have few or no programming rules so that you can demonstrate how to program in a whole class situation. With any of these starter games, you'll be remixing a new version of the game as you add to it.

Discussion/Reflection: From this point forth, where applicable, refer to Scratch rules or scripts learned.

Week 2: Learning Stagecast with PondLife

Warm Up: Using the game from the Main Activity of Week 1, ask youth to explain how a Sprite (choose one and refer to it by name) knew what to do. (Answer: Sprite was programmed). Next go through the rules with youth and have them tell you what each one tells the Sprite to do. Use the If...then...Handout for the rules that are If...then... statements. For rules that are not If..then... statements have youth describe what's happening in the code. In other words, where did the Sprite start and where did it end up.

Challenge: Find another game to demonstrate rule creation. Use one of the games listed above in Week 1 Challenge, one you found, or create one yourself so that you can demonstrate how to program in a whole class situation. With any of these starter games, you'll be remixing a new version of the game as you add to it.

Main Activity: Do the first part of Math Activity option 1 (playing battleship with the idea of coordinate plane), and then proceed to Math Activity option 2.

Week 4: Principles of Game Design & Stagecast Tutorial

Main Activity: Replace Stagecast Tutorial with all the resources in Scratch Help at <http://scratch.mit.edu/help/> Start with the Getting Started introduction, video and/or Getting Started Guide. Have youth go through the Scratch Cards and Video Tutorials on this page too. Choose one of the Explore These Starter Projects (http://scratch.mit.edu/starter_projects/) for youth to try each of the steps in the Getting Started Guide.

Week 5: Designing the Big Game

Main Activity: Have youth start where they left off last time with their Starter Projects (http://scratch.mit.edu/starter_projects/) They should practice all the programming in the Getting Started Guide and the Scratch Cards. If youth finish early, have them help others who have not finished or continue adding new features to their Starter Projects.

Week 7: First Week of Programming - Week 12: Sixth Week of Programming

Activities: Use Scratch for all programming activities. Have youth think through how they will organize themselves to create different levels that then become one game. They can use Backpack (<http://wiki.scratch.mit.edu/wiki/Backpack>) and Remix

(<http://wiki.scratch.mit.edu/wiki/Remix>) features to integrate their games. They should understand how these features work before starting to build the different levels of their game.

Week 13: Putting Humpty Dumpty Back Together Again

Getting Ready and Activities: Youth use Backpack and Remix to put together multiple games in Scratch. Check the wiki information, tutorials, and advice from the Scratch community for current information on how to use these features.