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

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

SRI Education
A DIVISION OF SRI INTERNATIONAL

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; however, it has been discontinued and this latest version of the unit takes advantage of Scratch.

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

Week 1: Programming vs. Playing Games

Summary

Schedule

Warm-Up	Discuss goals and objectives for the next semester.	30 min
	Brainstorm topics for the game.	
Challenge	Introduce Scratch.	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 Scratch
- LCD projector (for showing PowerPoint presentation and Scratch)
- 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 Scratch 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 Scratch games to get the feel for what is available. In the Main Activity and Discussion/Reflection, youth discover the difference between playing and programming. Both are fun and involve interactions with Scratch. 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 Scratch 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.
- **Scratch**. 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 Scratch to program a game of their own design.
- **Characters**. You create rules for characters to tell them what to do.

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/.

Scratch is a great way to introduce computer science concepts and programming to youth. 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.

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

Scratch is an intuitive program, with a huge community of users and educators. We recommend their learning guide (http://scratched.gse.harvard.edu/guide/).

Put the games you will use into a *Studio* (in Scratch, it's a favorites folder) so the youth can find them quickly and are not wasting time entering URLs. You can do the same for your students' games when the start programming.

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 13 week unit.

Make sure you have the Introduction PowerPoint (G-G D-zine) to start. Improve it with new graphics, audio, or even make a video of it. 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)

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
- For students to keep track of their work: http://scratched.gse.harvard.edu/guide/files/CreativeComputing20140820_LearnerWorkbook.pdf

Scratch Games

If you can't find the following games, search for some new ones!

- 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

- 5 x 7 Post-it notes
- Introduction PowerPoint (G-G D-zine)
- Job Description handout

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.
   a. 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 steps 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 Scratch environment.

Materials

- Chart paper
- Computers with Internet access
- Access to model 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/

To Do

1. Introduce Scratch 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 Scratch, software that allows them to make games.
   b. Demo how to use the software. For example, show the youth how to program a characters to move forward. Show them how to make a rule and use the Green start button.
   c. Explain that they will be learning how to make graphics and program games using Scratch during this unit.
2. Try out the Scratch games.
   a. Ask youth to play all the four model games. Remind them to hit the Green 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 Scratch.

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 Scratch interface and several tools.

Materials

- Computers with Internet access
- Computer connected to overhead
- https://scratch.mit.edu/projects/507/

To Do

1. Tell youth that they are going to start learning how to program by "looking inside" the game. Narrate out loud your steps.
2. Show the game (https://scratch.mit.edu/projects/507/) on the overhead projector. And, then click on "See Inside."
3. Look at the code for some of the sprites.
   a. Note that many sprites start with "When green flag is clicked".
   b. Read out-loud some of the scripts you see. Have kids try to predict what each script might do.
   c. Make sure you click on the Car sprite (the black line that the user controls). See what happens when you click on each of the scripts.
4. Click on "remix" to modify the game. Tell students that they can start learning code by remixing other people's games/scripts.
5. Demo changing a few things in the game. Then, have youth go to the game, and try their own remixes on their devices.

Discussion/Reflection

Time:	10 minutes
Purpose:	Reinforce Glossary and Scratch 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 Scratch.
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 about Coordinate Space (Math Activities)

Summary

Schedule

Warm-Up	Learn if-then rules	20 min
Challenge	Explore the coordinate plane.	40 min
Main Activity	The coordinate plane on the screen.	60 min
Discussion/Reflection	Discuss the 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 Internet access
- LCD projector
- Handout
- Blank paper
- Grid paper
-

Getting Ready

Overview

In the Warm Up, youth will learn about the math (logic) and computer science idea of "if-then" and how it applies to programming. For example, "if you don't program the character to behave in certain ways, then it doesn't know what to do." The Challenge and the Main Activity are math activities to explore the concept of the coordinate plane and how it relates to computer screens and programming. In the Discussion/Reflection, students reflect on what they've learned.

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 the "origin" for both of them. Usually, but not always, one of the two axes is horizontal, the other vertical; their positive directions are 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.[1]
- **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).

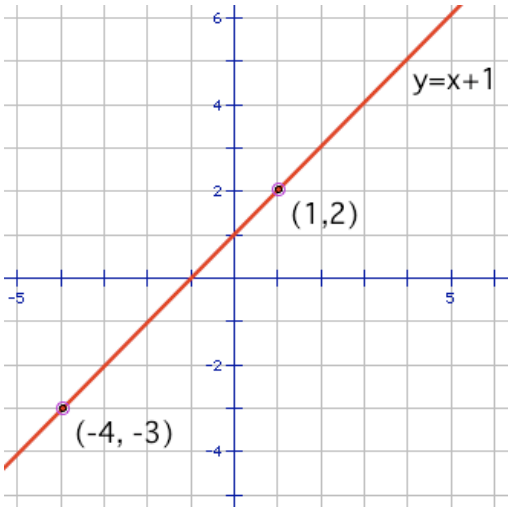


Figure 1. Cartesian Coordinate System and graph of y= x+1

- **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.
- **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

[1] 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.

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. Lots of the Scratch rules follow this logic: when the green flag is clicked, then start the game, for example.

Math Concepts

- Coordinate geometry
- Locating points on a grid
- Determining distance between points

Background

There math concepts in this week's activities is about the Cartesian coordinate system. In the Warm Up, youth explore the logic structure of causal relationships using **if-then statements**. Object-oriented programming, like the one we use in Stagecast and Scratch, uses this structure to follow through a set of rules for each character. If a set of conditions is met, then the code 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 Challenge, students describe the position of objects on paper 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. For the Main Activity, students explore the usefulness of the coordinate plane for programming and computers. 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.

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

- Computers with Internet access
- Writing If … Then… statements handout
- Animations: https://scratch.mit.edu/projects/73349820/
- About logic: https://scratch.mit.edu/projects/97246/
- Programming with logic: https://scratch.mit.edu/projects/11637247/

To Do

1. Open the animations and ask youth. How do these animations work?

   (Answer: They were programmed).

2. Click on See Inside, and click on any character. Have students try to read the script and then predict what the script will do. [This is a good chance to show that sprites can have multiple "costumes".]
3. Model saying the sentence as a when-then, or an if-then sentence. Emphasize the logic framing of the script. 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.
4. Together write the “IF…THEN…” statements for the handout

Challenge: Math Activity

Time: 60 minutes

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

Materials

- Computers with Internet access
- Grid paper the size of the screen (14 x 20)
- Grid paper overhead
- LCD projector
- Whiteboard

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.

Teaching Tips

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. 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. 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).

Main Activity: The Coordinate Plane on the Screen

Time: 20 minutes

Purpose: Work with graphing some functions

Materials
- Computers with Internet access
- Graph paper

To Do

1. 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 ($y = 3$).
2. 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.
3. Give each youth a graph paper with several "gems".
4. 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.
5. 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.
6. Have the youth share their strategies for the first two sheets. They are likely to come up with several combinations of lines.

Discussion/Reflection

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

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; *x*s and *y*s useful when planning things on a screen and figuring out where your characters are; if-then statements format)
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. Scale (when you asked them to predict the path for a large screen)
    c. If-then statements (for everything that happens in a game)
    d. 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.

Original Source was Kelley, H (2002) Breaking in, from http://www.igda.org/breakingin/career_paths.htm. A reproduction of this material is still available at: http://www.ign.com/boards/threads/video-game-developer-salaries.95718155/.

Warm-Up

Time: 10 minutes

Purpose: Stimulate youth thinking about how the design process relates to game creation.

Materials

- 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

- Internet access
- Walk Through Careers in Game Design handout
- My Design Dream Team 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

- 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 Scratch Surprise Activity.	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 access to the internet
- Keeping Track & Rules You Learn handout
- Fun & Simple handout
- Designer or Programmer that poster handout
- Access to Pac-Man at http://www.thepcmanwebsite.com/media/pacman_flash/
- Use The Surprise Activity in The original scratch guide, for educators, http://scratched.gse.harvard.edu/guide/
- Scratch Card Tutorials: https://scratch.mit.edu/help/cards/

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 Scratch formally, through the tutorials.

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 goals 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 Scratch Tutorials on the Scratch Cards.

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.

You'll also need a way to keep track of each pair's games – and so will the youth. Perhaps you can make accounts for each pair, or have one shared account for all youth.

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://dukenukem.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: 10 minutes

Purpose: Review the Design Requirements (from Week 1).

Materials
- Designer or Programmer Hat handout

To Do

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

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.

Challenge

Time: 60 minutes

Purpose: Analyze the design principles in Pac-Man.

Materials

- Computer with access to the internet
- Access to Pac-Man at: http://www.thepcmanwebsite.com/media/pacman_flash/
- Fun & Simple Handout

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.

Answers to Fun & Simple chart for Pac-Man:

Principle of Game design	Aspect	Description
Main character that your users can connect with	Characters	• 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	• Maze structure (user can go to some places, but not others). • Start-up screen is different.
Clear goal	Goal	• Clear screen of all dots (to go to next stage).
Simple way for user to interact with game	Controls	• User moves Pac-Man with arrow keys.
Clear decision making points	What choices are available?	• 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	• Shiny objects (make Pac-Man invincible temporarily). • Fruit (earns more points than dots).
	Rewards	• You gain points for eating dots, power dots, ghosts, fruit. • If you win several stages, you get another life.
	Challenges	• Eating dots makes Pac-Man a little slower (than when he runs in a clear passageway). • Ghosts are faster than Pac-Man.
	Punishments	• Lose a life if you get eaten by a ghost.
Easy way for user to tell whether they are winning or losing.	Feedback	• User can see the dots disappear. • Score is displayed prominently. • New screen appears when Pac-Man dies or moves to next stage.

10. Have youth analyze one adventure game in a similar way.
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 and an adventure is whether user choices affect the game or not.
    a. **mazes**
        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**

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 Scratch

Materials

- Computers with Internet access
- Access to the Scratch Tutorial
- Scratch Cards https://scratch.mit.edu/help/cards/
- Rule notebooks for the students, based on Keeping Track of Rules You Learn handout

To Do

1. Use all the resources in Scratch Help at http://scratch.mit.edu/help/
2. Start with the Getting Started introduction, video and/or Getting Started Guide.
3. Have youth go through the Scratch Cards and Video Tutorials on this page too.
4. 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.
5. Have the youth *write down any new scripts* 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.
6. Remind them to pay attention to the "if-then" rules, especially the ones that are highlighted in the tutorial.

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 adventure

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 Scratch)
4. What are some of the principles of game design?
5. What scripts did you learn to create in Scratch?

Week 5: Designing the Big Game

Summary

Schedule

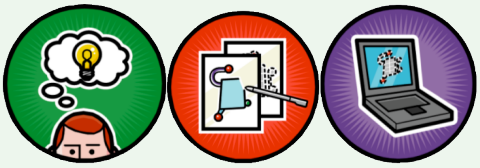
Warm-Up	Create chain stories to prepare for game design.	30 min
Challenge	Develop a design for the Big Game.	40 min
Main Activity	Map out the Big Game. Do more Scratch Tutorials.	60 min
Discussion/Reflection	Review pieces of the game youth accomplished.	10 min
Total Time		**2 hr 20 min**

Essential Questions

- What is programming?

Design Process Concepts Involved

- Brainstorm
- Sketch it
- Develop designs



Materials

- Computers with Internet access
- 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

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 continue learning Scratch.
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

There are several examples from which you can learn about storyboarding: http://www.uncc.edu/webcourse/sb/storyboard.htm. And http://multimedia.journalism.berkeley.edu/tutorials/reporting/starttofinish/storyboarding/

Warm-Up

Time: 30 minutes

Purpose: Create chain stories to prepare for brainstorming.

Materials

- 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 scripts in Scratch).
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.

Teaching 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 Scratch.

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 Scratch, 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 SCRATCH

1. 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.
2. If youth finish early, have them help others who have not finished or continue adding new features to their Starter Projects.

Discussion/Reflection

Time: 10 minutes

Purpose: Youth present their stage designs.

Materials
- 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 Scratch scripts did you learn? Can you show us?

Week 6: Prototyping & Storyboarding

Summary

Schedule

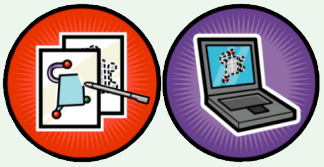
Warm-Up	Select stages in pairs.	10 min
Challenge	Create prototypes for the individual stages.	60 min
Main Activity	Create storyboards and plan for programming.	60 min
Discussion/Reflection	Discuss what pieces of the game youth accomplished.	10 min
Total Time		**2 hr 20 min**

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."[2]

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).

[2] 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 Scratch tutorial in the next activities.

Warm-Up

Time: 10 minutes

Purpose: Select stages in pairs.

Materials

- Map of the Big Game chart with topics (from week 5)
- Job Description (from week 1)
- Brainstorm chart

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	• Rapid Prototype of Your Stage handouts • 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 Scratch file and import all images.	25 min
Challenge	Plan and organize coding.	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 access to the internet
- The original scratch guide for educators: http://scratched.gse.harvard.edu/guide/
- Scratch Backpack: http://wiki.scratch.mit.edu/wiki/Backpack

Getting Ready

Overview

Youth begin programming their games. As they begin programming, they should keep track of their codes in their Design Notebooks and update it as they work on new sprites or objects. During the Main Activity continue programming and in the Discussion/Reflection, youth reflect on and share what they learned.

Glossary

- **User input**. Users can control characters in Scratch with keystrokes or mouse clicks.

Background

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.

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

During programming, encourage youth to copy scripts (using the backpack) from previously designed games or each other. Also, help them to learn how to troubleshoot their rules. Make sure they know how to test a script, and how to eliminate problems by looking at one aspect of the rule at a time. Does the game look like the rule? If not, what can you do to change this? Do you need another 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?

Scratch is based on the idea of DJ's scratching and making remixes from other people's work/art. Youth should look for inspiration in the work of previous scratchers, especially when they don't know how to start or how to proceed.

Warm-Up

Time: 25 minutes

Purpose: Open a new Scratch file and import all images.

Materials

- Computer with access to the internet
- List of characters for each pair

To Do

1. Determine how you want youth to save their programs (in pairs), and help youth set up their games, so that you and they can find them in the future.
    a. For example: Have them save their project as youth's name 1 + youth name 2 v01.
    b. 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.
    c. Explain that it's not uncommon to go up to 50 or 100 versions!
2. All youth should at least share 1 sprite – their main character.
3. Have youth pick additional the sprites and backgrounds they will use for their levels.
    a. 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.
    b. Encourage the youth to share any other characters that they may want across the games.

Challenge

Time: 45 minutes

Purpose: Write down a first *draft* of the scripts to be programmed.

Materials

- Computers with Internet access
- Making Character Scripts handout (Make 5 copies per team.)

To Do

1. Using the Making Character Scripts handout, ask youth to review all the scripts they think they will use to program their main character. This is the planning phase for the Main Activity. What are all the scripts they need?
    a. Action scripts without user input
    b. Action scripts with *user input* (key stroke/mouse click) for main character
    c. Interaction with other characters
    d. Interaction with objects
2. If teams run out of ideas, have them compare with other teams to see whether they find more scripts they were missing.
3. 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 games, and write down all the scripts they can think of, before they start programming.
4. Once the youth have finished writing down the scripts for the main character, have them start writing down the scripts for secondary characters.

Main Activity

Time: 60 minutes

Purpose: Begin programming main character.

Materials
- Computers with Internet access
- Design Notebooks

To Do

1. Ask the youth to use their Design notebooks (and scripts from the challenge) to start programming the main character.
2. Walk around, make sure that the youth are writing down the names of their scripts, and that they are keeping track of their scripts in their Design Notebooks.
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 /remix the scripts. Encourage them to use copying as a problem-solving strategy while programming.

Discussion/Reflection

Time: 10 minutes

Purpose: Share ideas about organizing scripts.

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 scripts allows you to see easily whether you've created all the scripts you need; it helps you find scripts 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 or ICT Visitor

Summary

Schedule

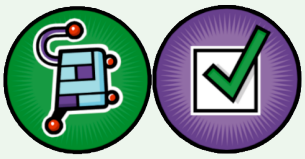
Warm-Up	Trying out new Scratch cards	15 min
Main Activity	Continue programming the game.	1 hr 40 min
Discussion/Reflection	Share insights in programming.	10 min
Total Time		**2 hr 20 min**

Essential Questions

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

Design Process Concepts Involved

- Test it
- Build it



Materials

- Computer with access to the internet

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 ICT professional. There are a lot of people who use Scratch, in schools or for fun. Use the Scratch website to find someone near you.

Youth will need a lot of support in organizing their scripts 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 script if it's taking them too much time. In addition, you should encourage them to analyze and copy scripts 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 Scratch.

Warm-Up

Time: 15 minutes

Purpose: Trying out new Scratch cards

Materials
- Computer with access to the internet

To Do

1. Have youth do a tutorial on how to create scripts for the user to control the main character.
2. Have them learn how to add sounds and program new ones. 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 Scratch.

Main Activity

Time: 1 hr 40 minutes

Purpose: Continue programming the game.

Materials
- Computers with Internet access

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 scripts for one character at a time. As the youth create scripts for each of their characters, remind them to check script and character interactions. Sometimes, a script they create for one character is in conflict with the script 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 scripts they need to create.
    d. Some youth will have an easier time writing scripts 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.

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, Test it).
4. How many more weeks do youth have before they should be done with their games?

Week 9: Testing Stages

Summary

Schedule

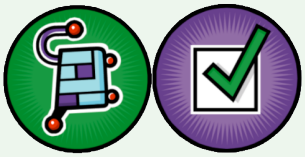
Warm-Up	Get stage ready for review.	20 min
Challenge	Review at least two other stages and provide feedback.	50 min
Main Activity	Debug games based on colleagues' feedback.	55 min
Discussion/Reflection	Share insights about programming.	15 min
Total Time		**2 hr 20 min**

Essential Questions

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

Design Process Concepts Involved

- Test it
- Build it



Materials

- Computer with access to the internet
- People Involved in Testing Games handout
- 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 scripts to fix them. It is an organized and methodical way of checking every script to find and fix the problem.
- **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 games so that other 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. Then, they have a chance to iterate: debug and refine the instructions.

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 scripts for their games.

Important. Read "Debugging Scripts" strategies for debugging Scratch code: http://wiki.scratch.mit.edu/wiki/Debugging_Scripts

Warm-Up

Time: 20 minutes

Purpose: Prepare stages for peer review.

Materials

- Computer with access to the internet
- People Involved in Testing Games handout
- Game Instructions 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 game (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. 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 script 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 Internet access 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.

Teaching 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
- Computers with Internet access

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 scripts. Give youth the space to work on any of those, as long as they don't get too sidetracked 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 10: Third Week of Programming

Summary

Schedule

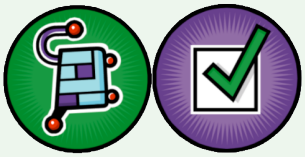
Main Activity	Continue programming the stages in the last week.	2 hrs
Discussion	Review programming successes.	20 min
Total Time		**2 hrs 20 min**

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 access to the internet
- 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
- Computers with Internet access

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 11: Re-assembling Humpty Dumpty

Summary

Schedule

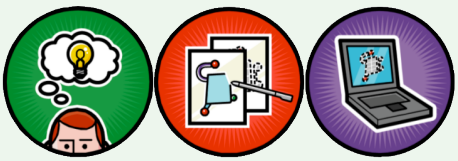
Challenge	Put Humpty Dumpty back together again. Create a New Map of the Big Game.	1 hr 10 min
Main Activity	Continue programming and incorporate final details.	1 hr 10 min
Total Time		**2 hr 20 min**

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 access to the internet
- 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

- N/A

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 game. Search the Scratch website to learn how to combine games. For example: https://scratch.mit.edu/discuss/topic/9045/. There will likely be newer and better suggestions when you teach this.

Challenge

Time: 1 hour 10 minutes

Purpose: How to put Humpty Dumpty back together again.

Create a New Map of the Big Game

Materials

- Map of the Big Game chart (from week 5)
- Screen shots of each game, preferably in color
- 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 student 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 Dumpty[3]

Humpty Dumpty sat on a wall;	Sentado en un muro.
Humpty Dumpty had a great fall.	Humpty Dumpty
All the King's horses	Se ha caído muy duro.
And all the King's men	Todos los caballeros
Couldn't put Humpty together again!	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?

[3] 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
- Computers with Internet access

To Do

1. Youth should incorporate the pieces they are missing in their games. For example, they may be missing ways to connect to each other's games.
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.
- Games need to relate to one another and follow pathways that allow movement from stage to stage so that players can visit every stage.

Weeks 12 and 13: 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 Internet / Access to Scratch
- 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

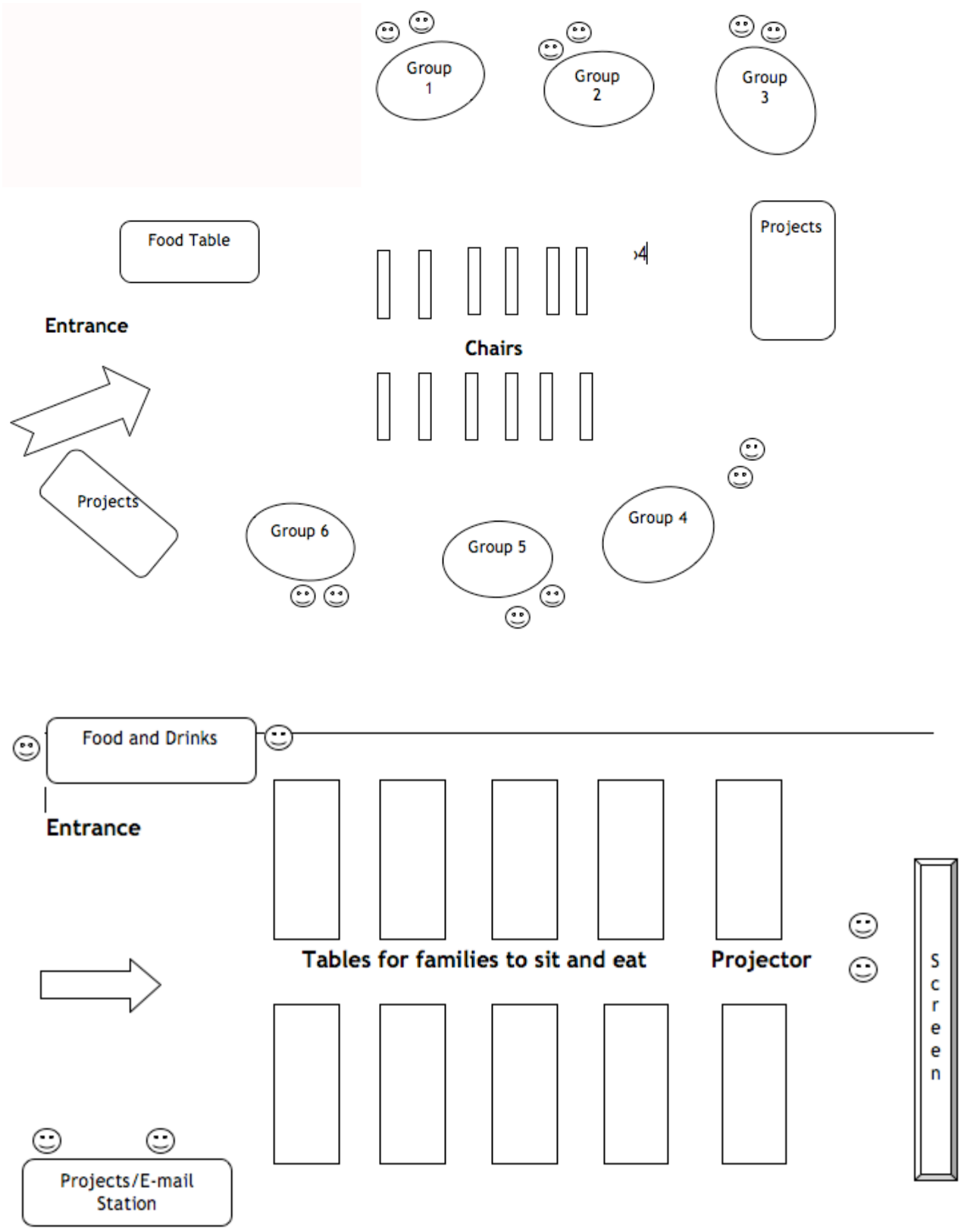
Refer back to the unit's lessons. Family Tech Night gives youth a chance to revisit lessons completed throughout the unit and use the information they learned when presenting their projects to the guest coming to Family Tech Night.

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 access to Scratch 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 Scratch 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.


Group 1
Group 2
Group 3
Food Table
Projects
Entrance
Chairs
Projects
Group 6
Group 5
Group 4
Food and Drinks
Entrance
Tables for families to sit and eat
Projector
Screen
Projects/E-mail Station

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

- 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
- 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 Scripts 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 Scripts

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 game level (with your partner).
3. Put everything back together (as a whole group).

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

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

Materials and tools you have for this job:

1. Users - students entering middle school
2. Scratch – the programming software environment
3. Game design principles
4. Game structures to use (maze, adventure, both)

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 4 weeks.
4. You will have 2 weeks to prepare and present your final product.

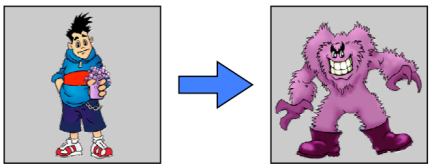
Design Requirements

Whole game	Your *plan* for *A Day in the Life of a Middle Schooler* game includes: ☐ **A context or story** (What is the point of this game? How do the stages fit together?) ☐ **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?) ☐ **Learning goals about middle school** (What are you hoping to teach younger students about middle school?)
	Your *game, A Day in the Life of a Middle Schooler*, includes: ☐ A starting page with instructions about how to play and win the game ☐ A way to visit all the levels ☐ A clear way for user to know when the game has ended ☐ 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	Your *plan* for your stage includes: ☐ **A gameplan** (ways to win, strategies for winning, and ways to lose the game) ☐ **A storyboard** (layout of the stage; critical changes when user moves, finishes game) ☐ **Learning goal** (What are younger users going to learn when they play this stage?)
	Your *level* (developed in pairs) includes: ☐ The main character (controlled by user input) ☐ A way to get to the next game or return to the main page ☐ Instructions about how to play and win the game ☐ Two or more characters that the main character interacts with ☐ A game (e.g., a maze or an adventure) ☐ Clear way for user to know when she has finished the game ☐ Something the user learns about middle school ☐ Scripts organized so that a friend (another software engineer) could easily find or add a script
FTN presentations	Your *presentation* at Family Tech Night should: ❏ Describe how your game satisfies the design requirements (given by the client, G-G D-zine) ❏ Describe what users will learn about being a middle schooler in your game ❏ A map of the big game, detailing the authors of each stage and a typical path the user may take ❏ Storyboards that you used to create the game

Game Comparison Table

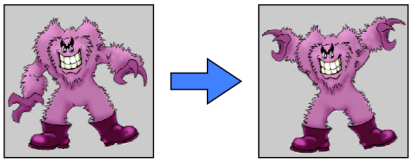
Game Name	**Goal & Story** (What's the point of the game, and what's the story?)	**Type of Game** (Maze or puzzle, adventure)	**How do you complete the game?**	**Fun for 4th-5th graders? Why?**

Writing **If…Then Statements**



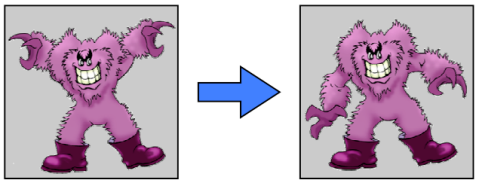
If the kid has a potion in his hand,

then, change him into a monster.



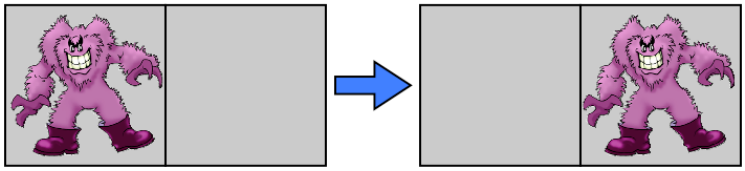
If _____

then, _____.



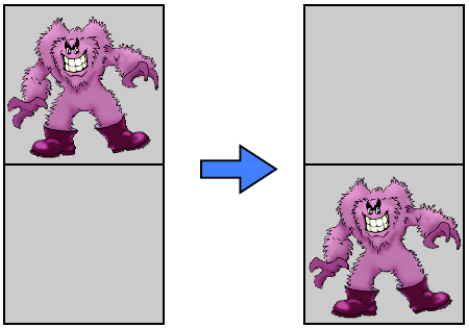
If _____

then, _____.



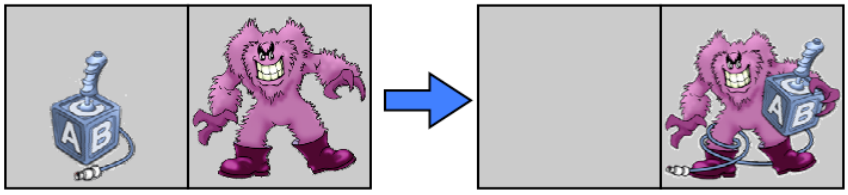
If _____

then, _____.



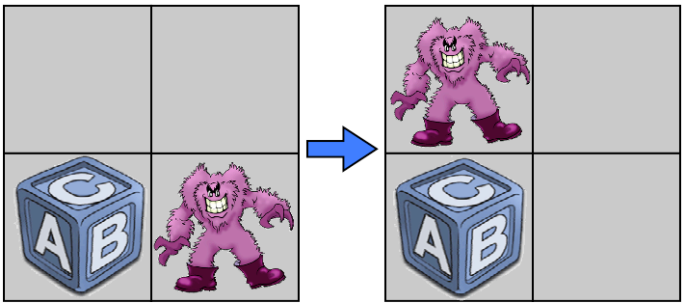
If _____

then, _____.



If _____

Then, _____.



If _____

Then, _____.

Blank Grid

Walk Through Careers in Game Design

Check out these careers in game design

Programming

No matter their specialization--AI, physics, graphics engine, networking, and so forth--programmers continue to earn relatively more than developers of all other disciplines across all levels of experience. This is especially true for industry veterans of six or more years, likely due to the rarity of experienced console engineers.
Interestingly, salaries for the least experienced programmers dipped somewhat compared to the previous Salary Survey, possibly due to a proliferation of college graduates moving into an ever-increasing raft of entry-level positions, as well as existing programmers becoming older, more seasoned, and thus shifting brackets.

Average yearly salary:

	programmer/engineer	lead programmer	technical director
<3 years	$54,300	$58,486	$63,750
3-6 years	$68,072	$81,155	$77,129
>6 years	$86,243	$93,067	$115,087

Years experience in the industry:

>6 years - 45% 3-6 years - 31% <3 years - 24%

Percentage receiving additional compensation:

Receiving additional compensation - 72%

Not receiving additional compensation - 28%

Average additional compensation - $21,872

Type of compensation:

Annual bonus - 53% Royalty - 23% Profit sharing - 21%

Project bonus - 25% Stock options - 44%

Average salary by gender:

Male - 96.1%; $78,186 average salary

Female - 3.9%; $67,669 average salary

Highest salary - $211,500

Art and animation

Lead artists/animators reported that they made significantly more money in 2004 than 2003 across all levels of experience, but the increase was particulary fruitful for animators with three or more years experience; for those straight-forwardly titled "artists," salaries increased only marginally.

Also, though some artists worry about burnout, it's clear that many are sticking around. Fourty-four percent are veterans of six or more years, compared to just 37 percent last year.
Finally, although artist salaries generally come in at significantly less than coders' for those with similar experience, the highest individual salary for any artist was $220,000, beating out the top programming salary of $211,500.

Average yearly salary:

	Artist	animator	lead artist/animator
<3 years -	$42,512,	$44,778	$64,036
3-6 years	$55,594,	$65,619	$62,411
>6 years -	$64,870,	$73,031	$78,700

Years experience in the industry:

>6 years - 44% 3-6 years - 32% <3 years - 24%

Percentage receiving additional compensation:

Receiving additional compensation - 66%

Not receiving additional compensation - 34%

Average additional compensation - $19,168

Type of compensation:

Annual bonus - 53% Royalty - 37% Profit sharing - 16%

Project bonus - 31% Stock options - 35%

Average salary by gender:

Male - 93.2%; $62,520 average salary

Female - 6.8%; $53,574 average salary

Highest salary - $220,000

Game design

"I want to make video games when I grow up." For the bright-eyed, children who make this statement, it loosely translates to "I want to be a game designer." (Here, the term comprises game designers, level designers, and writers.) The job sounds idyllic to those not really in the know, but it's competitive as hell. College graduates (or even talented dropouts) vying for a position as a game designer will find adequate entry-level salaries as a result of the competitive nature of the title. Increases in pay tend to be more commensurate to experience than title, at least for the first few years in the industry.
The best advice I've heard for budding designers: Find yourself an experienced mentor. Listening and asking questions of others might be the best way to negotiate the path between designer and lead, even if the pay is relatively static between titles.

Average yearly salary:

	game designer	creative director/lead designer

<3 years	$44,176	$43,778
3-6 years	$52,604	$51,777
>6 years	$67,840	$78,913

Years experience in the industry:

>6 years - 41% 3-6 years - 38% <3 years - 21%

Percentage receiving additional compensation:

Receiving additional compensation - 69%

Not receiving additional compensation - 31%

Average additional compensation - $16,890

Type of compensation:

Annual bonus - 44% Royalty - 32% Profit sharing - 26%

Project bonus - 32% Stock options - 36%

Average salary by gender:

Male - 94.5%; $59,865 average salary

Female - 5.5%; $43,844 average salary

Highest salary - $190,000

Production

Producers are the shepherds of the daily grind, managing all the details from relationships, to schedules and budgets, and make a pretty decent base salary. From our reported figures, producers who stick with it for at least six years will earn significant payoffs for their time; and compensation other than salary is among the highest in the industry for this group.
But where do producers come from? It's been said in the past that the producer's title is what many QA personnel and support staff aspire to. But as the game industry stretches its wings in the entertainment sector, we could see more experienced professionals coming into games from the other industries by using this directional title to their advantage.

Average yearly salary:

	producer/project lead	executive producer
<3 years	N/A	$52,533
3-6 years	$60,802	$79,989
>6 years	$82,715	$118,417

Years experience in the industry:

>6 years - 61% 3-6 years - 26% <3 years - 13%

Percentage receiving additonal compensation:

Receiving additional compensation - 79%

Not receiving additional compensation - 21%

Average additional compensation - $39,707

Type of compensation:

Annual bonus - 60% Royalty - 29% Profit sharing - 16%

Project bonus - 25% Stock options - 55%

Distribution of gender:

Male - 82%

Female - 18%

Average salary - $78,401 (average salary by gender N/A)

Highest salary - $210,000

Quality Assurance

Results from this year's salary survey indicate that not everyone in the industry is rewarded for several years of service. QA employees who have been in the industry for three, four, or five years, on average, don't make much more than their fledgling counterparts. What's the message? That a rookie is as valuable as an experienced QA person?
Fortunately, there's an upside: QA has traditionally been a position in which non-technical game enthusiasts could get their foot in the door, and, anecdotally speaking, that doesn't seem to have changed much over the years--it's still an excellent path to game design and production-related positions.

Average yearly salary:

	Tester	QA lead
<3 years	$33,362	$43,195
3-6 years	$33,386	$38,340
>6 years	$48,435	$60,929

Years experience in the industry:

<3 years - 44% 3-6 years - 36% >6 years - 19%

Percentage receiving additional compensation:

Receiving additional compensation - 38%

Not receiving additional compensation - 62%

Average additional compensation - $8,543

Type of compensation:

Annual bonus - 64% Royalty - 10% Profit sharing - 15%

Project bonus - 10% Stock options - 44%

Distribution of gender:

Male - 91%

Female - 9%

Average salary - $39,933 (average salary by gender N/A)

Highest salary - $225,000

Audio

Licensing music is in, or is it?
Experienced audio engineers and musicians, directors, and composers this year have recorded their salaries as being markedly higher than last year, especially for entry-level musicians. Audio and musical people with more than six years experience can, in the game industry, make a salary on par with programmers with equal years of experience.
Unfortunately, we received the fewest number of (usable) responses from people in the audio category: a scant 63, perhaps reflecting the fact that many musicians don't work solely in games. Additionally, so few lesser-experienced audio recruits responded that it discourages a deeper analysis of wages in this discipline.

Average yearly salary:

	sound/audio designer/engineer	composer/musician
<3 years	$44,176	$43,778
3-6 years	$52,604	$51,777
>6 years	$67,840	$78,913

Years experience in the industry:

>6 years - 48% 3-6 years - 33% <3 years - 19%

Percentage receiving additonal compensation:

Receiving additional compensation - 62%

Not receiving additional compensation - 38%

Average additional compensation - $29,885

Type of compensation:

Annual bonus - 46% Royalty - 49% Profit sharing - 24%

Project bonus - 49% Stock options - 35%

Average salary by gender:

Male - 96.8%; $69,352 average salary

Female - 3.2%; $78,500 average salary

Highest salary - $225,000

Business and legal

Maturation of the game business is a sure thing, and as it expands, the scope of its people must also expand. New to the salary survey this year are respondents who classify themselves in the business and legal category--people whom we must include in our community and game creation business plan, since without them, no thriving company would have a business plan at all.

For the purposes of the survey, business and legal breaks down into three groups: marketing, public relations, and sales; executive; and other staff or administration.
There are a surprising number of business and legal professionals who are acclimated to the game industry's unique climes. Their higher salaries might be attributed to commission or simply rank and file, in the case of executives. But like developers of all disciplines, these people usually know their games, so let's not discount them just because they don't AI script like maniac.

Average yearly salary:

	marketing/PR/sales	executive	other staff/administration
<3 years	$60,490	$69,598	$76,120
3-6 years	$56,765	$93,282	$65,938
>6 years	$83,708	$113,646	$94,188

Years experience in the industry:

>6 years - 46% <3 years - 30% 3-6 years - 24%

Percentage receiving additonal compensation:

Receiving additional compensation - 75%

Not receiving additional compensation - 25%

Average additional compensation - $45,188

Type of compensation:

Annual bonus - 56% Royalty - 16% Profit sharing - 29%

Project bonus - 21% Stock options - 46%

Average salary by gender:

Male - 81.8%; $90,332 average salary

Female - 18.2%; $71,050 average salary

Highest salary - $256,000

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

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

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

4. Which job pays the highest potential salary?

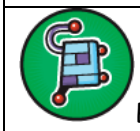
5. Which job pays the next highest?

6. What does a sound engineer do?

Check out the following profiles:

- **Tricia Harris**, Web Designer & Writer
  http://www.mobygames.com/developer/sheet/view/developerId,28665/
- **Chris Degnan**, Lead Designer
  http://www.mobygames.com/developer/sheet/view/developerId,66379/
- **Darryl Duncan**, Composer/Musician
  http://www.mobygames.com/developer/sheet/view/developerId,42820/
- **Seonaidh Davenport**, Program Manager
  http://www.mobygames.com/developer/sheet/view/developerId,141040/
- **Douglas Noel**, QA Game Test Coordinator
  http://www.mobygames.com/developer/sheet/view/developerId,104369/
- **Genevieve Picard**, Software Developer
  http://www.mobygames.com/developer/sheet/view/developerId,167551/
- **Tammy Yap**, Programmer
  http://www.mobygames.com/developer/sheet/view/developerId,81137/
- **Tito Pagan**, Senior Artist/Animator
  http://www.mobygames.com/developer/sheet/view/developerId,4280/

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	

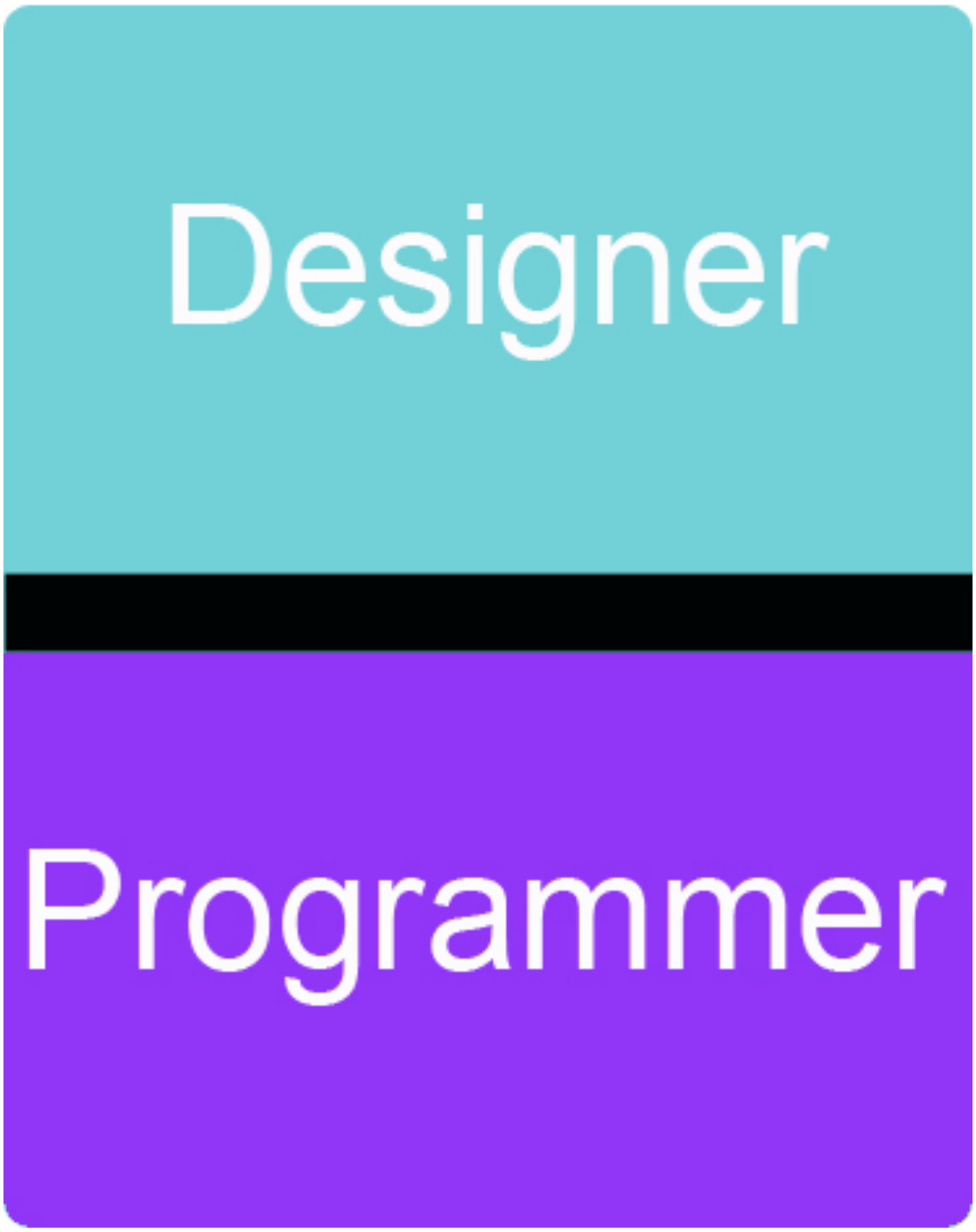
Keeping Track of Scripts You Learn

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

Script name			
Character name			
Describe what happens in script			
Starting position & appearance			
User input (if any)			
Other characters involved in script			
Ending position & appearance			
Write the script			

(e.g., If…then…)			

Designer or Programmer Hat?



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

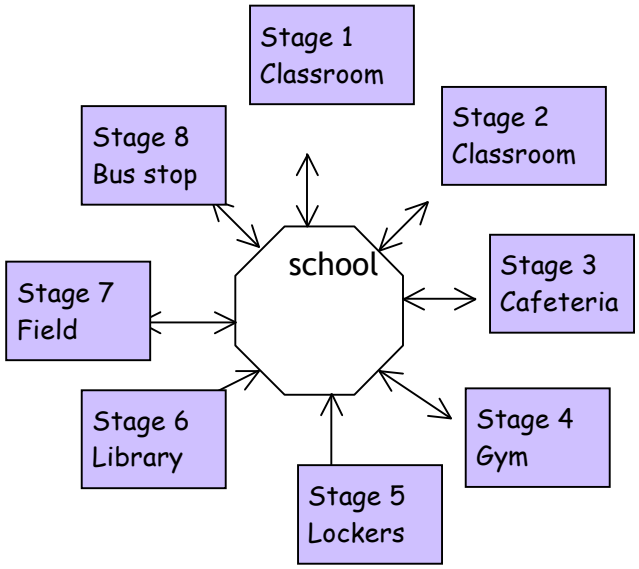
What makes a good first-person game?

- 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).

4.

5. 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

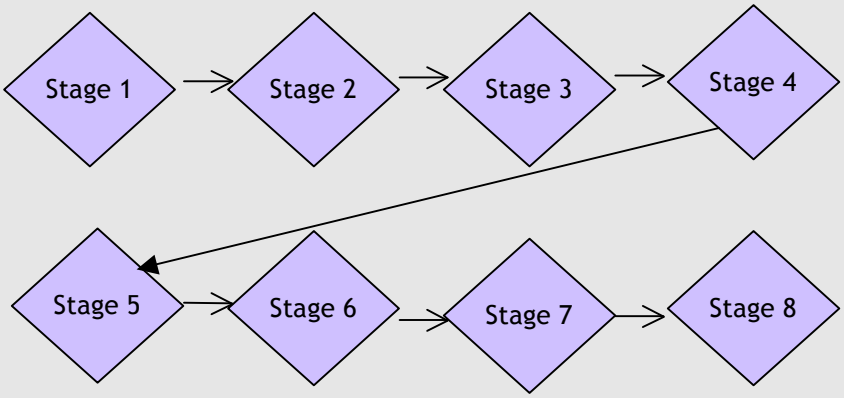


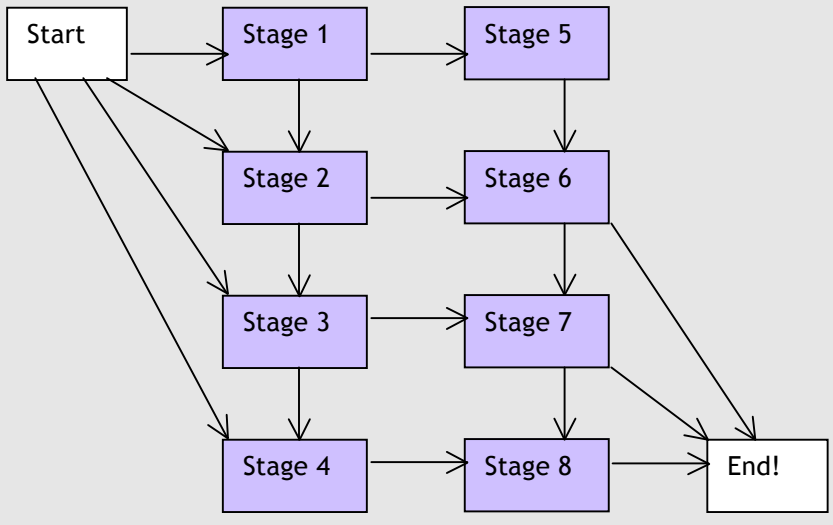
What do you notice in this map?

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

What do you notice in this map?

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





What do you notice in this map?

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

Rapid Prototype of Your Stage

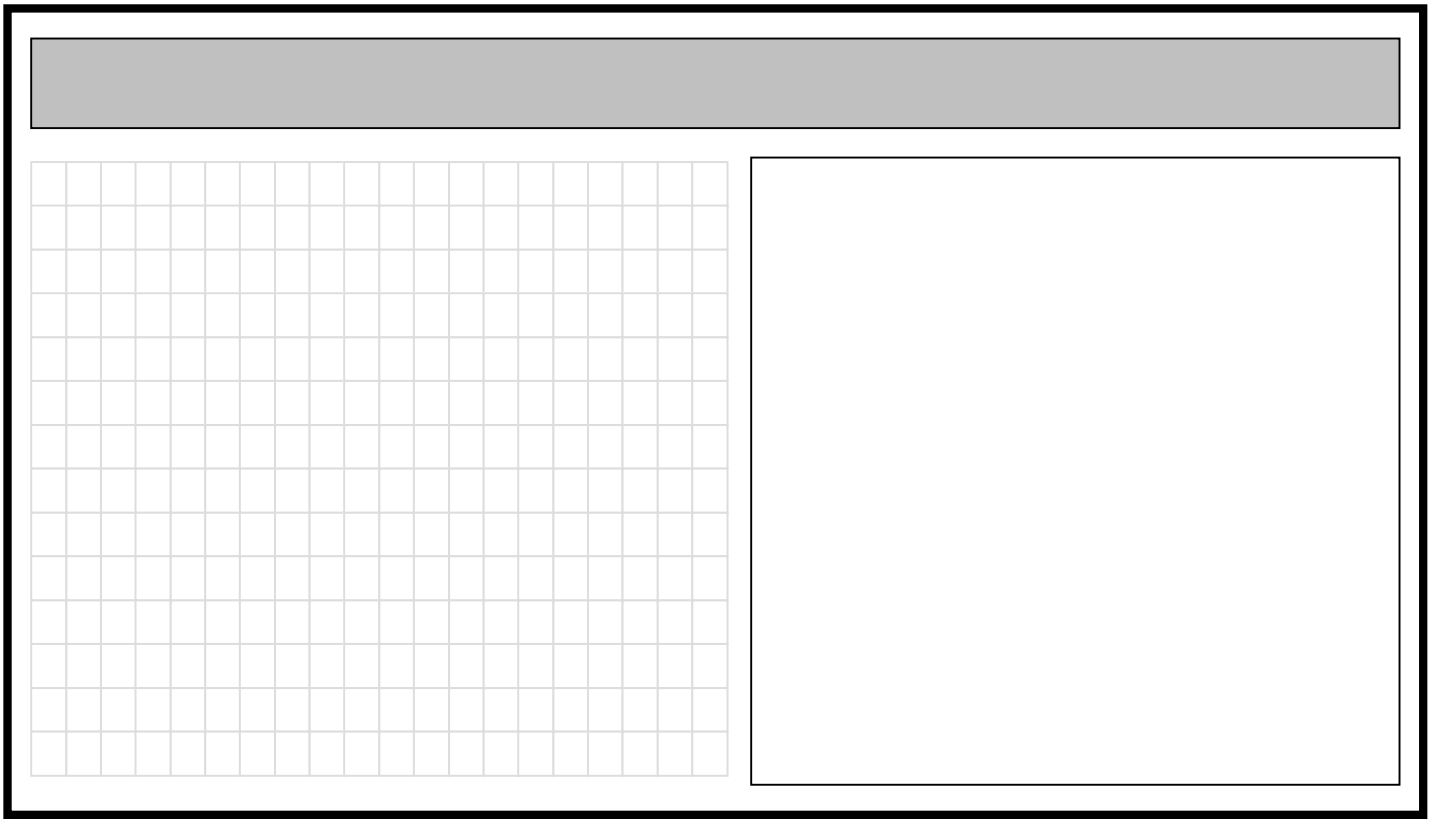
Project Title _____

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?



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

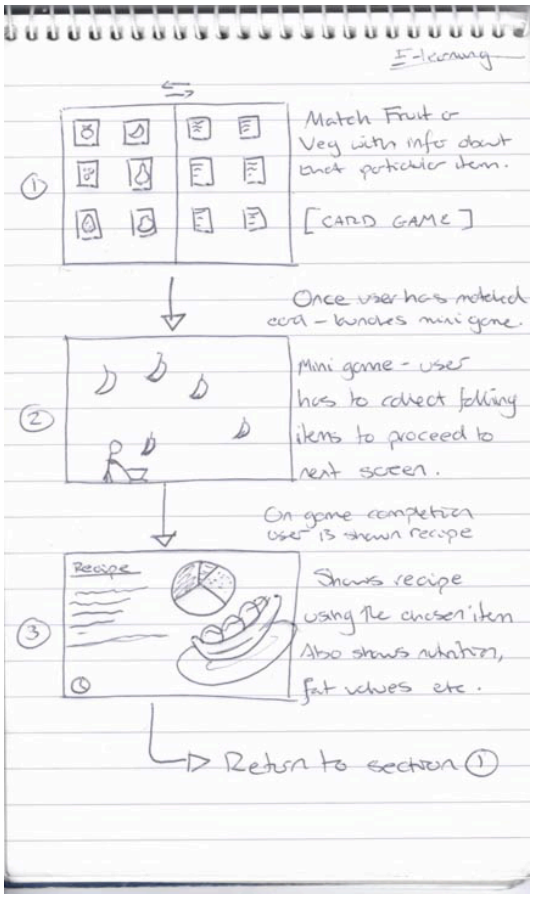
   Computer output—how the computer responds to the user

   Scripts—what scripts do you have to create to perform this task

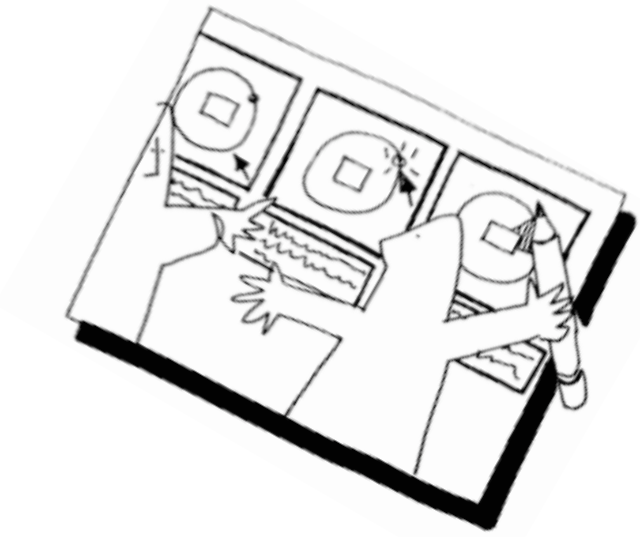
2. Describe interaction details and emotional responses verbally when no visual representation is effective.

   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 Scripts

Script Name				
Describe what happens in the script				
Starting position and appearance				
Other characters involved in script				
User input				
Ending position and appearance				
Write the script (e.g., If…then…)				
Script 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 scripts:

6. What do you think this stage teaches 5th graders?

7. How can the programmers improve this game? (Give concrete answers).