EXPANDED LEARNING BY CITIZEN SCHOOLS

# Amazing Mazes

## Materials and Teaching Checklist – lesson 7

Lesson Name: Maze Walker Programming loops and conditions
Date to be taught: 4/23/2013

**"I Can" Skills:**

| Last Time | This Time | Next time |
|---|---|---|
| **Create a basic maze walker program to control the walkers** | **Create and test maze walker programs with loops and conditions** | **Create and test more advanced maze walker programs (breadcrumb algorithm)** |

**Before the Lesson:**

| Copies to Make | Materials to Bring | Visuals to Make |
|---|---|---|
| | - Projector<br>- Computers with Internet connectivity<br>- | |

**During the Lesson:**

| Section | Time (min) | I Say / Do | They Say / Do |
|---|---|---|---|
| **Hook** | 5 | **Review and Teach-back**<br>- Quick review and **teach-back** of last lesson:<br>- left-hand walk, right-hand walk | - **Students review key points of last lesson** |
| **Activity 1** | 15 | **Inside maze view algorithms – playing a game: "hide that cheese" (target)**<br>* Teacher shows the new Java applet at http://employees.org/~hmark/courses/amazingmazes/amazing-mazes-12-programming-algorithms-1.html and explains the new elements on the User Interface:<br>○ "Draw with mouse", "Left-hand walk", "Right-hand walk"<br>* Then the teacher will explain the game "Hide that cheese" (target):<br>– Students will build a maze of their own design, either using the "draw-path" button of the User Interface, or the "Draw with mouse" button.<br>– One student in the pair will then place 1 | - **Students will play the game "Hide that cheese" (target)**<br>- **Students will try to hide the cheese so that it takes the walker the longest to find.**<br>- **The teacher will tabulate the longest paths, and declare a winning pair.** |

| | | | |
|---|---|---|---|
| | | walker anywhere in the maze, and the other student will place the target in the maze, in a way that they think will take the walker the most time/steps to reach.<br><br>– Then they will start the left-hand walk algorithm and count the steps the walker takes to get to the target.<br><br>– The students will switch roles, and see if they can find a longer path (basically trying to "hide" the target from the walker).<br><br>– Teacher will tabulate the results (longest path) for each pair. The longest (i.e. the cheese is hidden the best) is the winner. | |
| **Mini Lesson** | 10 | **Program loops**<br><br>– Using the programming algorithms applet http://employees.org/~hmark/courses/amazingmazes/amazing-mazes-12-programming-algorithms-1.html start with a very simple maze: a straight line of length 10<br><br>– draw-path (1,1)(10,1)<br><br>– put the walker at (1,1) and the target at (10,1)<br><br>– ask the students to call out the inside view program: turn-right, forward, forward, … 9 times<br><br>– introduce the loop structure:<br><br>turn-right<br><br>repeat-until-target<br><br>forward<br><br>– Explain that all commands appearing after the line repeat-until-target will be repeated from top to bottom, and then back to the top, until we reach the target, or hit the system limit of about 300 loops/iterations, and then stop.<br><br>– Now create a few other simple mazes, for example L shaped, U shaped, or staircase shaped, and ask the students to call out the repeated set of commands within the loop.<br><br>– For the U shaped program, the walker does a turn-back (tb), and then repeats the sequence of 6 times forward followed by one turn-left | - Students help create the program:<br><br>tr<br><br>fd<br><br>fd<br><br>fd<br><br>fd<br><br>...<br><br>draw-path (9,1)(9,7)<br><br>– position the walker at (3,7) and the target at (9,7)<br><br>– The program is:<br><br>tb<br><br>repeat-until-target<br><br>fd<br><br>fd<br><br>fd<br><br>fd<br><br>fd<br><br>fd<br><br>tl |

| | | | |
|---|---|---|---|
| **Activity 2** | 15 | **Program conditions (some conditions)**<br><br>• Using the programming algorithms applet http://employees.org/~hmark/courses/amazingmazes/amazing-mazes-12-programming-algorithms-1.html draw a "staircase" maze. Note the staircase should not define any pattern of forwards and turns, but should for example be:<br><br>draw-path (1,1)(3,1)<br><br>draw-path (3,1)(3,4)<br><br>draw-path (3,4)(7,4)<br><br>draw-path (7,4)(7,6)<br><br>draw-path (7,6)(12,6)<br><br>draw-path (12,6)(12,10)<br><br>draw-path (12,10)(14,10)<br><br>draw-path (14,10)(14,11)<br><br>draw-path (14,11)(17,11)<br><br>• Ask the students to try and come up with a loop pattern, which they shouldn't be able to do.<br><br>• Create a walker program with conditions and explain how it'd work:<br><br>• repeat-until-target<br><br>fd<br><br>cft<br><br>if-lto tl<br><br>if-rto tr<br><br>• For each iteration of the repeat loop, the walker move forward (fd), then checks for turns (cft).<br><br>• This means that the walker is "looking" to see if it can turn left, right, forward, and back<br><br>• Then it has 2 condition questions:<br><br>• if there is a left turn only possible (if-lto) then the walker should turn left (tl). If there is no left turn possible, the walker does nothing, and the next line in the program/loop is executed<br><br>On the next line, if there is a right turn only possible (if-rto) then the walker should turn right (tr) | - **Students should not be able to come up with a repeatable pattern for the loop, since the staircase is not "regularly repeating"**<br><br>- |

| | | | |
|---|---|---|---|
| **Activity 3** | 15 | **Adding more program conditions**<br><br>• Using the programming algorithms applet http://employees.org/~hmark/courses/amazingmazes/amazing-mazes-12-programming-algorithms-1.html copy and paste **2 parts** of the "Hs maze" below the applet (only the first 2 H shapes)<br><br>• Place the walker at (1,5) and the target at (6,1)<br><br>• Run the partial left-hand program above (activity 2), and show that it will fail to reach the target<br><br>• Ask the students to explain why, and come up with additional conditions to be added to the full left-hand-walk program:<br><br>• repeat-until-target<br><br>fd<br><br>cft<br><br>if-lto tl<br><br>if-rto tr<br><br>if-slo tl<br><br>if-lro tl<br><br>if-slr tl<br><br>if-sro ss<br><br>if-n tb | - **Students will explain why the walker fails: not all conditions and junction cases are being addressed/handled by the partial program**<br><br>- **Students will help identify and define the missing conditions:**<br><br>○ **Left-and-right-turn (if-lro)**<br><br>○ **Left-and-straight-turn (if-lso)**<br><br>○ **Right-and-straight-turn (if-rso)**<br><br>○ **Left-eight-and-straight-turn (if-lrs)** |
| **Activity 4** | 15 | **Summary of left/right-hand algorithms**<br><br>* The teacher copies and pastes the maze below the program at http://employees.org/~hmark/courses/amazingmazes/amazing-mazes-12-programming-algorithms-1.html (this is a "replicated H shape" from previous lessons).<br><br>* The teacher creates a maze walker at (1,5), and asks the students to call out the location of the target which would result in the longest search (path, number of steps) by the walker.<br><br>○ This is a trick question, since the target location that's "hidden the best" depends on the walker algorithm.<br><br>○ For a left-hand walk the best target location is at (3,4) | - Students call out the "best location" for the target. But this actually depends on the walker algorithm/walk that we select to execute:<br><br>- for left-hand walk (3,4)<br><br>for right-hand walk (3,6) |

| | | | |
|---|---|---|---|
| | | ○ For a right-hand walk the best target location is at (3,6)<br><br>* The teacher places the target in both places and shows how both algorithms/walks find the target | |
| **Exit Tix** | 10 | If we have time:<br><br>- Hands-on activity:<br><br>- the students can try to create a maze and hide the target in a way that **neither a left-hand walker nor a right-hand walker will find the target** (it is possible!). | - **Students in pairs, build a maze and try to hide the target so that neither walking algorithm hits/finds it.** |
| **Dismiss** | | Remind students of our goal regarding programming:<br><br>- We will learn in the next lessons how to program "fool proof" programs (programs that don't fail or break), but we'll do it in stages, refining and strengthening our programs every time. | |

## Thumbnails lesson outline:

- **Review and teach-back** of lesson – what's a left-hand-walk maze walking program?

- **Student Game "Hide the cheese" - NetLogo -Programming Algorithms**
  - **Students create a complex maze** – 1 student places walker, 1 places target
  - see if left-hand or right-hand algorithms can find the target in the largest number of steps

- **teacher – explain loop in a straight line - NetLogo -Programming Algorithms**
  - staircase maze

- **teacher – explain turn check and some conditions - NetLogo -Programming Algorithms**
  - replicated H maze

- **teacher – explain all conditions - NetLogo -Programming Algorithms**
  - replicated H maze

- **Teacher demos/summarizes "Hide the cheese" - NetLogo -Programming Algorithms**
  - using the "replicated H shaped maze", place walker at (1,5), ask students where the best location for hiding the target is? (depends on which type of walk/algorithm)

- **If we have time:**

- **students program Inside Maze view program for the Complex Maze, trying to create a maze and target location which both left and right hand walks will not find**