LEARNING MODULE

GK-12 DISSECT at New Mexico State University

Title: Computer Free: Diagramming

Author: Jezreel Bassett and Melody Hagaman

Discipline or Area: Computer Science, applied to all courses

Teacher: Melody Hagaman

School: Centennial High School

Subject of class: Beginning Computer Science

Grade: 9 - 12 th

COVERAGE OF COMPUTATIONAL TOPICS

Vocabulary: Variables, Branching, Iteration, Loops, Parameters

OBJECTIVES

This module should inspire students to logically plan out a project in a sequence of steps.

EQUIPMENT AND MATERIALS

Diagramming: Construction paper, Tape (Wide, clear, packing tape and scotch tape), and large dry erase boards and markers (Black worked best for all colors)

Game Design: Art materials for developing game board and/or cards, dice, many knickknacks for game pieces (we suggest you have students use and return the dice and knickknacks so they are reusable), index cards, etc.

May need clear plastic shower curtain liners (see note in text below).

Background and References

What is the purpose of this module, and how does it relate to the content of the teacher's current lesson plans? Where are you getting your information and theories? Have any other researchers been involved in development?

This module is designed to teach planning through diagramming. Planning for any project is extremely important especially when involved with many people. In this modules students will work in groups to design a board game first by diagramming out the rules, followed by creating the game (board, cards, etc.), finally play it out to see their plan in action and how to improve the process.

Procedure

Provide detailed instructions on how this module is taught.

Diagramming Lecture Prep: In diagramming the shape and color are important they indicate the intent of the step. To begin creating the lecture diagramming components cut out 2 large round or rounded edge squares from yellow construction paper, label them "START" and "FINISH", these depict where to begin and end the diagramming process. The rest of the lecture diagram components were reused multiple times so they will not initially explain what to write on them.

(NOTE: If you cover diagram components in clear packing tape dry erase markers will write on them and rub off making the pieces reusable.)

Then cut out a large orange parallelogram, this indicates when a user (or in our case a player of our game) has to input/output a value. A computer science example of this is when a user inputs a password or a variable; a game example is when a player answers a trivia question or rolls a die.

Then a large blue square, indicating an internal calculation such as " i++ " in computer science (which is a increment of the variable " i " from the programming language C++, shorthand for " i = i + 1 "), or adding to a player score for game play. Although this may seem trivial to the students it is important to remember to have the students diagram when and where scores are increased and decreased in their games.

Finally a large green diamond, this diagramming component indicates a choice or a branching point in the diagram. A computer science example of this would be "If the password is valid continue to grant access, if not re-ask the user for password." A game example could be "If the dice roll is a double multiply the sum by 3 and add to score, if not add the sum to score."

After you have finished creating your lecture pieces attach a tape roll to the back so they will stick to the dry erase board. Drawing arrows from one to another will allow you demonstrate diagramming to your students. NOTE: Arrows have only one direction and directions must be followed.

Lecture demonstration should cover three main diagram examples; a linear diagram, a branching diagram, and a looping diagram. Linear diagrams are the most basic of the three examples; they follow a linear path and are the best way to demonstrate the diagram components input/output and calculation (see Example 1). Branching diagram has a path that has a decision or a choice breaking the path into two (see Example 2). It is important to note that only one path can be taken at a time. If the decision is not either of 2 choices but a combination of both create a both path. Looping diagrams are branching diagrams in which one of the paths connects to a previous point in the diagram. NOTE: Looping without a alternative branch path is possible but will produce an infinite loop which in computer science will allow a program to indefinitely run (not good). It is important to pick variables and calculations that will clearly lead students thru the loop to understand the looping effect. (Lecture took a part of one class period)

Diagram Activity Prep: Cut out the diagram components as before in diagram lecture. Each group of student will need a few of the input/output, calculation, and branching components. We suggest taping all the components so the students have an opportunity for trial and error. Break class into groups and designate each group a space on the dry erase board. **NOTE:** If you do not have enough space on the dry erase boards we used clear plastic shower curtain liners as tablecloths on desks, it worked as a dry erase board.

Give them a task to complete. Once finished, have the groups rotate and the new group attempt to follow the diagram of the old group to see the output. Do as many times as needed. (Diagram Activity suggested 2 class periods)

Board game diagramming Activity: Now the students fully understand diagramming have them break into groups and brainstorm to create a board game. We had our students do a class period of research (playing games) before this. They should choose a game name, write down a list of rules, diagram out game play, and create a board, cards, and game pieces. These games are to be coherent, playable and hopefully fun. Once the students finish their game we had groups rotate and play each other's games and write a review for student assessment. (Board game took 4 class periods: 1 for research, 2 for design, and 1 for student assessment)

What were the "learning goals?"

Become familiar with diagramming and group planning.

How did you introduce CT?

Introduced logical thinking through designing a sequence of steps. We also introduced computer science vocabulary and methods.

How could you assess the understanding of CT in this module?

The diagramming activity can be assessed due to clarity and correctness of the tasks given to the groups, as well as group participation.

The game design assessment will be over group participation (assessed in group), playability (coherences of rules and diagram), and creativity (how visually appealing the board and game components are, how interesting the game idea is).

Notes and Observations

What were challenges you encountered in the overall development of the module?

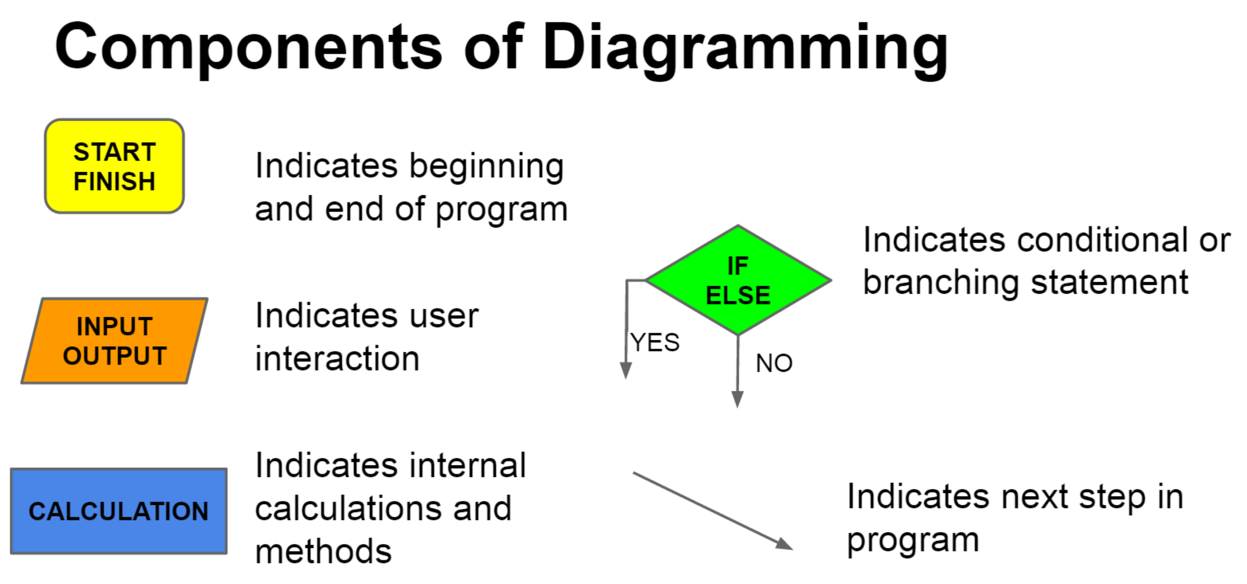
The greatest challenge encountered was the explanation of the diagramming. This was a new and difficult concept but highly valuable across multiple disciplines and directly linked to computational thinking. We suggest spending a few class periods on the diagramming activity for this reason.

What was successful?

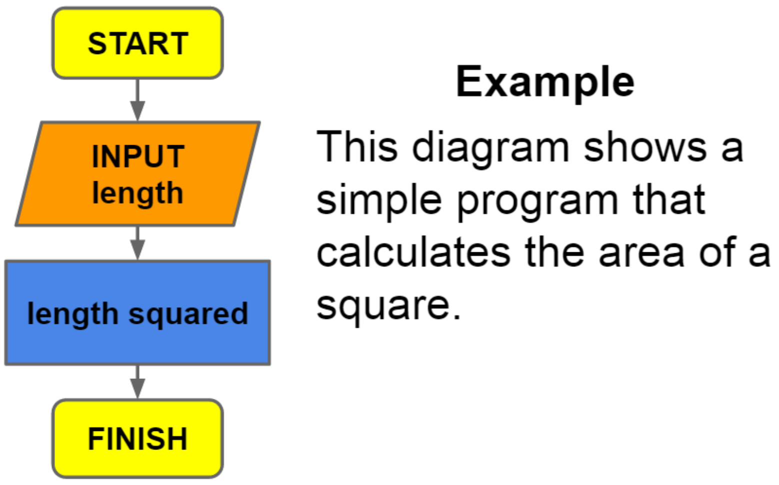
The game design was highly successful. The students worked diligently and were very proud of the results. The games showed that good diagrams could be read by students who have not worked with them and they could understand the game with few further directions.

How was the students' reception to the content of the module?

The students enjoyed developing the game and making it their own. The diagramming was highly challenging but by the end they understood the value of it.


Components of Diagramming
START FINISH
Indicates beginning and end of program
INPUT OUTPUT
Indicates user interaction
CALCULATION
Indicates internal calculations and methods
IF ELSE
YES
NO
Indicates conditional or branching statement
Indicates next step in program

Example 1

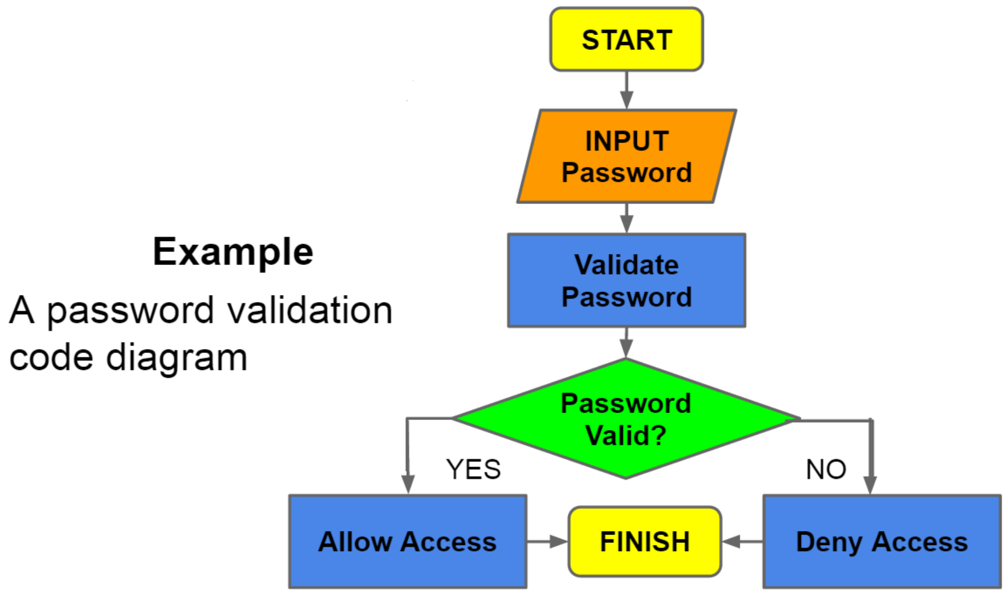


Example

This diagram shows a simple program that calculates the area of a square.

This is an example of a linear diagram

Example 2

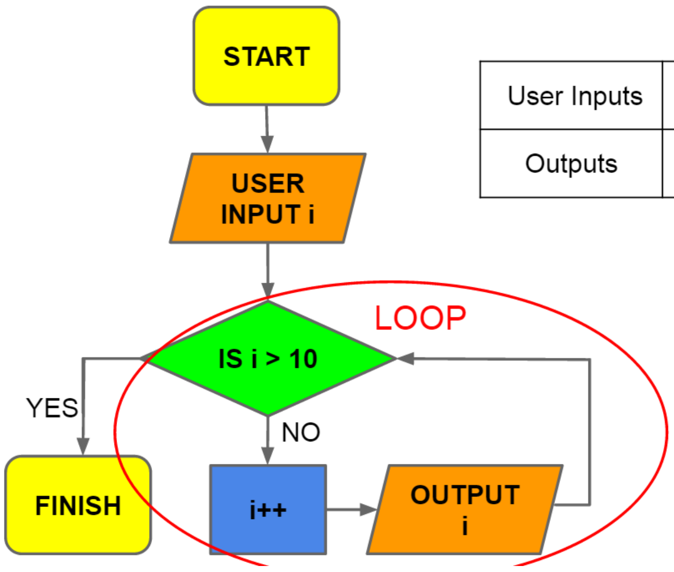


Example

A password validation code diagram

This is an example of a branching diagram. The branch occurs at the green diagram, the 2 paths are represented by the “yes” and “no” arrows. Notice that both path lead to the finish.

Example 3



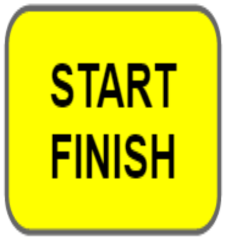
User Inputs	7	16	10
Outputs	8 9 10 11		11

Example

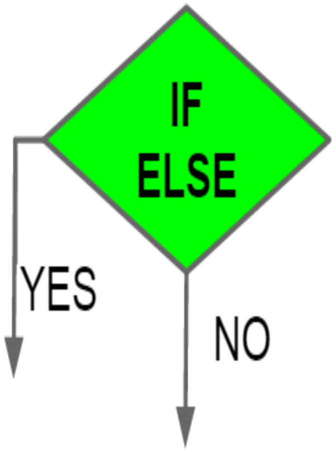
Simple check to see if the variable i is greater than 10 (output for effect)

This is an example of looping. NOTE: “ i++ ” is a representation of “ i = i + 1 ”,
“ IS i > 10 ” means “Is the value of i greater than 10” if the value of “i” is 10 then the answer is no.

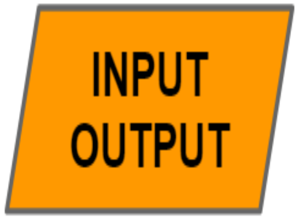
Components of Diagramming



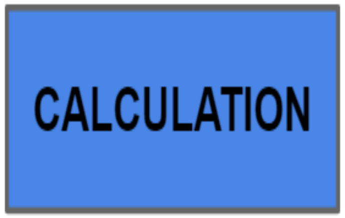
Indicates beginning and end of program



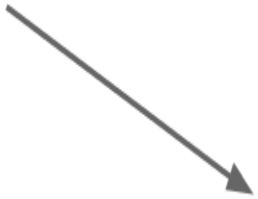
Indicates conditional or branching statement



Indicates user interaction



Indicates internal calculations and methods



Indicates next step in program