Activity

CHAPTER 3
Programming Concepts, Repetition

Name:
Date:

Purpose: Group activity teaching the concept of repetition as used in Arduino programming. Text like this denotes actual Arduino code.

Materials: Cones, large boards to display loop headers and pseudo-code, equipment for physical activities, and a field or gym.

Vocabulary to be explained prior to activity:

loop or repetition:
A section of code that repeats.

repetition header:
The line at the very beginning of a loop that tells the computer how the code inside the loop will repeat. This section is different for each different type of loop.

Conditional or question:
This is the statement that is checked to see if the loop is completed. Conditionals are present in loop headers and often look like this: $x < 10$. This indicates that the loop will continue until $x < 10$ is false.

Increment:
The section of code (may be in the header or may be in the loop body code) used to change the variable that is checked in the conditional. Using the example above, $x = x + 1$, one is added to x getting it a little closer to being larger than or equal to 10.

Nested repetition:
A loop inside of a loop. This concept is key for any type of even slightly advanced programming.

Types of loops:

- **loop:**

This loop is the most basic of all loops (that's why it's called loop) and is present in all Arduino sketches. loop( ) repeats as long as there is power to the Arduino. Inside this form of repetition is where you will find all other forms of repetition.

Header:
loop( )

Increment:
N/A

Conditional:
Power must be on.

- **while:**

This loop repeats as long as the conditional listed inside the parenthesis is true. This loop's conditional is incremented in the body code or through an Arduino input.

Header:
while( )

Increment:
In body code or Arduino input

Conditional:
Inside header parenthesis.

- **for:**

This loop repeats as long as the conditional listed inside the parenthesis is true. The for loop header declares a variable, checks a conditional and increments the conditional variable all inside the parenthesis...

Header:
for (int x = 0; x < 10; x = x + 1)

Increment:
Inside header parenthesis, in this example $x = x + 1$.

Conditional:
Inside header parenthesis, in this example $x < 10$.

Activity
CHAPTER 3
Programming Concepts, Repetition

Name:
Date:

Preparation:
This activity is a physical activity and you will need to set up an obstacle loop or course that reflects the repetitions you have decided to include in this activity. You may wish to work with a gym teacher in order to set this activity up.

The examples in this activity require three different stations. These include a "loop" station at the beginning of the obstacle course with a teacher or student helper, a "while" station with jump ropes and an area for spinning in circles, and a "for" station with an area for doing jumping jacks and shooting basketballs.

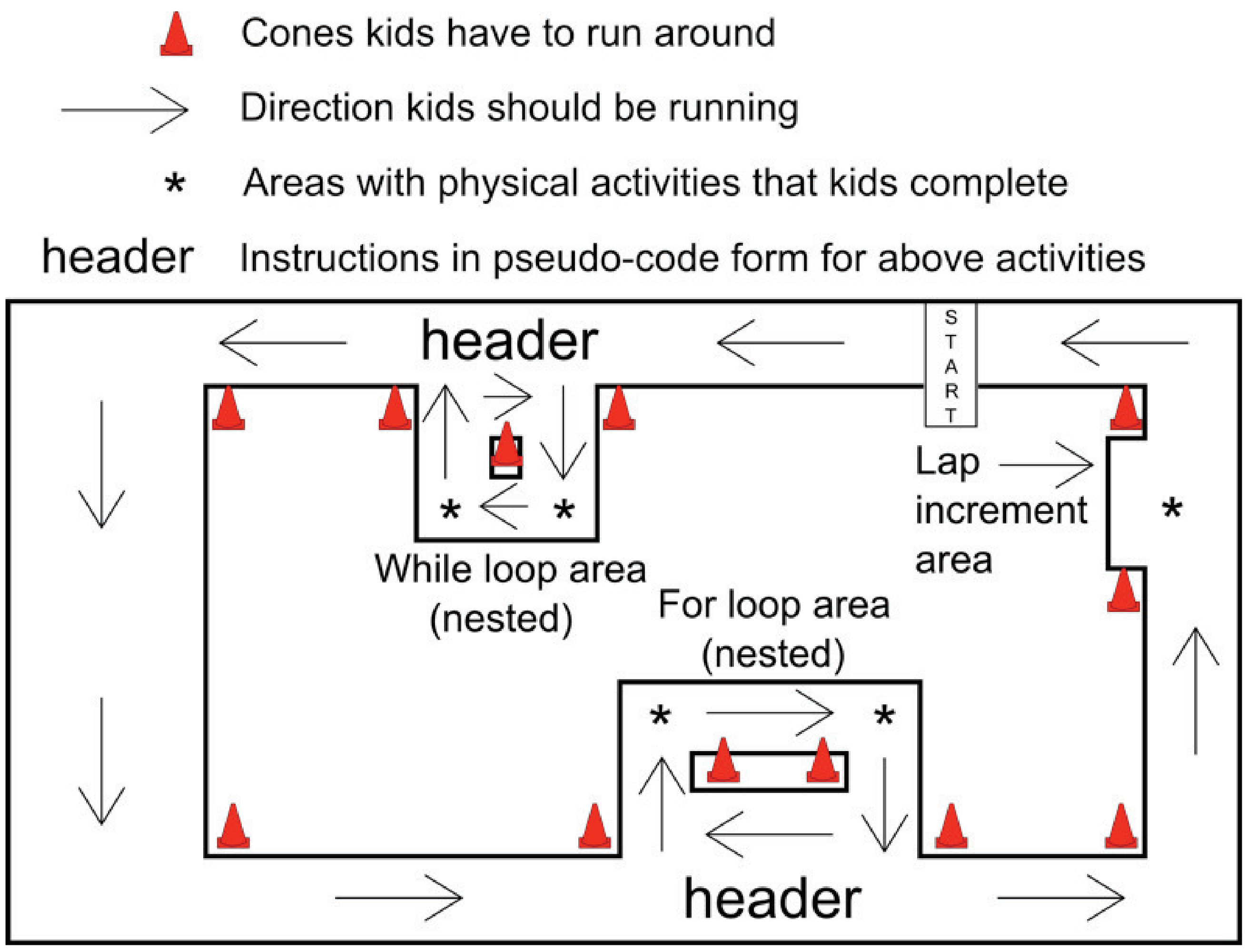
Each station will need a poster displaying the pseudo-code that students need to follow in order to complete the obstacle loop. The poster materials are included with the rest of the activity materials in the folder programming in the file called LoopActivityMaterials.

You will also need a field for kids to run around or cones to set up an area for kids to run around inside a gym.

Also- this is a really big activity. It takes a lot of prep and will probably be chaos the first time you try it, but it is easily customizable to age or skill level and should be lots of fun if you stick with it.

Activity:
Students should have completed the introduction to repetition worksheets that come with this activity. Students should also be familiar with variables and if statements.

What your loop activity might look like:


Cones kids have to run around
Direction kids should be running
Areas with physical activities that kids complete
header
Instructions in pseudo-code form for above activities
header
START
Lap increment area
While loop area (nested)
For loop area (nested)
header

Name:
Date:

The idea is that the complete obstacle course from the start position back to the start position represents the loop ( ) function. Inside this loop ( ) function are two nested loops, a while ( ) loop and a for ( ) loop.

At the beginning of the obstacle course each student needs to declare an integer variable called lapNumber or something similar. This variable will be used in each of the loop activity areas and the lap increment area. The lapNumber variable can also be used to end the obstacle course if you do not wish to have students run the obstacle course until the end of the period. When students start the obstacle course lapNumber should equal zero since they have not run any laps yet.

Nested loop activity areas: These areas are nested loops where students will perform a certain number of tasks depending on what your loop headers say. You can have as many or as few activity areas as you like. You may also tailor the number of physical tasks inside these nested loops to make your obstacle course more fun for your students.

These activity areas should look like little loops that the students can run around completing tasks. The headers should follow the format of the loop type it represents. For examples see the end of the activity. Once inside the nested loop activity area students must complete the physical activities according to the pseudo-code posted inside the nested loop activity area. Once students are done with the first repetition of the physical activities inside the nested loop activity areas they should look at the header again and decide if they have completed the nested loop represented by the header. With younger students you may want to have someone helping them with this step. (This can be fun, the observer can yell out error in a friendly voice if students exit the loop too quickly) Once students have completed the nested loop activity area they continue around the obstacle course to the next activity.

Header examples:

If a student's lapNumber is equal to three and the pseudo-code header reads:

```
while (lapNumber > basketsMade) {
do (lapNumber * 2) jump ropes at jump rope station
shoot lapNumber basketball baskets
}
```

This time around the obstacle course, the student would run through the nested loop activity area once, jumping rope six times and shooting three baskets along the way.

If a student's lapNumber is equal to three and the pseudo-code header reads:

```
for (int x = 0; x < lapNumber * 2; x = x + 1) {
do (x * 2) jump ropes at jump rope station
shoot lapNumber basketball baskets
}
```

The student would run through this nested loop activity six times jumping rope a different amount and shooting three baskets each time for a total of thirty six jump ropes and eighteen baskets.

There is a lot of room for personalization in this activity; it's an opportunity to really solidify the loop concept as well as getting your kids some exercise.

Name:
Date:

Lap increment area:
The lap increment area is where students will add one to their lapNumber variable to keep track of how many times they have run the obstacle course. You can also set up the headers and nested loop activities to use the lapNumber variable. The lap increment area is where you might insert an if statement to end the obstacle course after students complete a certain number of laps.

Additional thoughts:
Definitely call the obstacle course a loop( ) instead of an obstacle course in order to really get kids comfortable with the concepts. You may also wish to include your students in the planning of the obstacle course. Planning the obstacle course is another opportunity to talk about the loop concept and it gives them a stake in the learning exercise. Lastly, not that this needs pointing out, but this is a great activity just prior to computer lab time. Instead of having kids bouncing off the monitors they will be calmer and ready to sit still applying the concepts they just solidified through physical activity. This is great for kinesthetic learners in particular.