# Coding

## Get with the Program!

Coding is the process of writing instructions for humans or computers to produce certain outcomes for users. It's the combining of minute details to form a cohesive whole. More than anything, it is a creative and problem-solving activity. The problem solving aspect of coding will become very evident as we move forward. For now, I'd like to talk about the creative side.

Coding is a beautiful thing. However, I warn you that learning to program isn't trivial. You should be able to learn the basics with relative ease. Unfortunately, going from these basics to being able to write a useful program isn't easy. In fact, it can be quite frustrating. I tell you this so that you know what to expect, not to discourage you. How far you take this journey is up to you, this is but a start.

You already know what a program is. You use an email program, surf the web in a browser and interact with various websites every day. You text, take pictures, and check the weather on your phone. A program is the output, or goal, of programming. Much like a book is made up of chapters, paragraphs, sentences, phrases, words and finally punctuation and letters, so too is a program broken down into smaller and smaller components. For now, the most important is a *statement*. A *statement* is analogous to a sentence in a book. On its own, it has structure and purpose, but without the context of the other *statements* around it, it isn't that meaningful. A *statement* is more commonly known as a *line of code*. That's because *statements* tend to be written on individual lines. As such, programs are read from top to bottom, left to right. You might be wondering what *code* is. That happens to be a broad term which can refer to the whole of the program or the smallest part. Therefore, a *line of code* is simply a line of your program.

Who uses this code and why? Computer Science is the field that uses lines of code to generate programs for people to use. It is an amazing form of expressional art that produces a functional and useful product that can entertain and inform the masses. When most people think of computer science they envision people sitting in front of a computer screen all day typing in code. In reality, computer science is an engaging and collaborative activity that includes much more than typing on a computer.

Computer science starts with a problem that needs to be solved. Once the issue has been identified, the computer scientist looks at the problem and breaks it into manageable pieces. Computer scientists are not just programmers, they are problem solvers. You will use your problems solving skills in today’s activities to invent code to accomplish various tasks.

### TimeLine:

| | |
|---|---|
| 9:00 – 9:30 | Cultural Connection |
| 9:30 – 9:45 | Intro PowerPoint |
| 9:45 -10:15 | Intro Activity – Algorithm for Camp |
| 10:15 – 10:45 | Activity 1 – Binary Coding |
| 10:45 – 11:30 | Activity 2 – Graph Paper Coding |
| 11:30 – 12:15 | Activity 3 – Coding Cups |
| 12:15 – 12:45 | Lunch |
| 12:45 – 2:00 | Activity 4 – Physical Coding |
| 2:00 – 3:00 | Activity 5 – Angry Bird Coding |

### ND State Standards Addressed:

| | |
|---|---|
| 9-10.1.1 | Explain how models can be used to illustrate scientific principles |
| 9-10.1.2 | Describe the interaction of components within a system |
| 9-10.1.4 | Describe the relationship between form and function |
| 11-12.1.1 | Explain how scientists create and use models to address scientific knowledge |
| 9-10.2.1 | Explain how scientific investigations can result in new ideas |
| 9-10.6.1 | Use appropriate technologies and techniques to solve a problem |

## Materials List

### Introductory Activity

No Materials Needed

### Activity 1

1 sheet of graph paper for each student

### Activity 2

No Materials Needed

### Activity 3

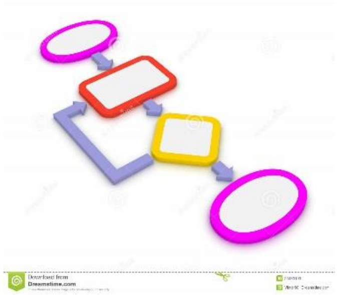
6 Plastic Cups for Each Pair of Students

### Activity 4

Various obstacles to use in the course.
Folding Chairs
Folding Tables
Broom Sticks
A Few Objects for Students to Pick Up (Golf Balls, Baseballs, etc.)
Blindfolds

### Activity 5

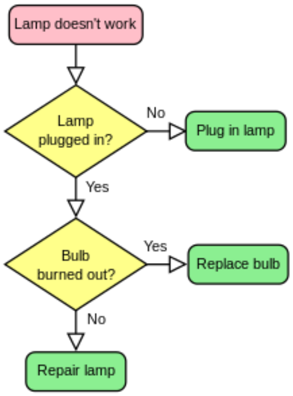
Computer Lab



# Introductory Activity Coding for Camp

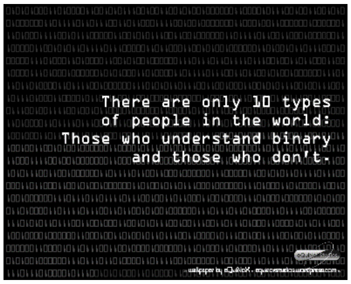
Throughout today's activities, you will see the words algorithm and code. While these 2 words have a very strong connection, they have different meanings. An algorithm is a process or set of rules that is used to solve a problem while code is a system of words, letters or figures used to describe the steps of any certain process. In other words, the algorithm is how you will solve the problem and the code is what you will use to solve the problem. Algorithms are frequently represented in diagrams called flow charts. These flow charts may contain various shaped figures with instructions contained in them. There will also be arrows directing the "flow" of the algorithm to direct you from the beginning to end.

Here is an example of a simple flow chart that shows what to do if your lamp does not work.



Create your own flow chart that represents the algorithm you used to get to camp this morning.
Use any shapes you choose for your flowchart.

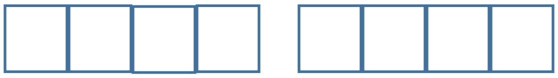
# Activity 1
# Binary Coding


There are only 10 types of people in the world: Those who understand binary and those who don't.

The best computer scientists understand what it's like to "be" a computer. In this activity, you are going to learn to understand how computers read and store data.

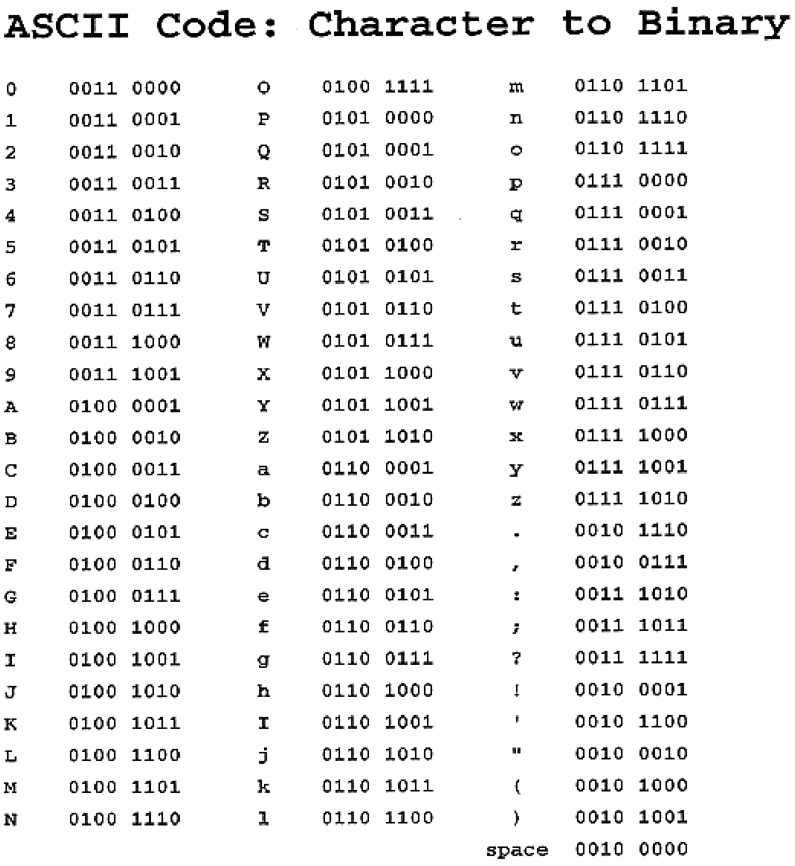
Computers operate using a binary system. Binary meaning 2. Computers are machines that are only capable to 2 operations, (on and off) (0 or 1), so any task that a computer completes must be coded into a series of operations that have only 2 choices. The advantage of a computer is that they can complete many millions if not billions of operations per second.

Computers use a series of 8 blocks (1 bit) split into 2 groups of 4 to store the letters of the alphabet with each letter represented by a different pattern of 0's and 1's



This code is called the ASCII Alphabet it stands for American Standard Code for Information Interchange.

In this activity, you will learn to interpret ASCII and create code of your own that represents words and sentences.



## ASCII Code: Character to Binary

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0011 0000 | O | 0100 1111 | m | 0110 1101 |
| 1 | 0011 0001 | P | 0101 0000 | n | 0110 1110 |
| 2 | 0011 0010 | Q | 0101 0001 | o | 0110 1111 |
| 3 | 0011 0011 | R | 0101 0010 | p | 0111 0000 |
| 4 | 0011 0100 | S | 0101 0011 | q | 0111 0001 |
| 5 | 0011 0101 | T | 0101 0100 | r | 0111 0010 |
| 6 | 0011 0110 | U | 0101 0101 | s | 0111 0011 |
| 7 | 0011 0111 | V | 0101 0110 | t | 0111 0100 |
| 8 | 0011 1000 | W | 0101 0111 | u | 0111 0101 |
| 9 | 0011 1001 | X | 0101 1000 | v | 0111 0110 |
| A | 0100 0001 | Y | 0101 1001 | w | 0111 0111 |
| B | 0100 0010 | Z | 0101 1010 | x | 0111 1000 |
| C | 0100 0011 | a | 0110 0001 | y | 0111 1001 |
| D | 0100 0100 | b | 0110 0010 | z | 0111 1010 |
| E | 0100 0101 | c | 0110 0011 | . | 0010 1110 |
| F | 0100 0110 | d | 0110 0100 | , | 0010 0111 |
| G | 0100 0111 | e | 0110 0101 | : | 0011 1010 |
| H | 0100 1000 | f | 0110 0110 | ; | 0011 1011 |
| I | 0100 1001 | g | 0110 0111 | ? | 0011 1111 |
| J | 0100 1010 | h | 0110 1000 | ! | 0010 0001 |
| K | 0100 1011 | I | 0110 1001 | ' | 0010 1100 |
| L | 0100 1100 | j | 0110 1010 | " | 0010 0010 |
| M | 0100 1101 | k | 0110 1011 | ( | 0010 1000 |
| N | 0100 1110 | l | 0110 1100 | ) | 0010 1001 |
| | | | | space | 0010 0000 |

Interpreting ASCII Code

Convert each Code to its corresponding number, letter or symbol.

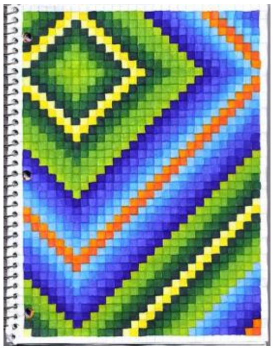
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | _______ |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | _______ |
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | _______ |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | _______ |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | _______ |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | _______ |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | _______ |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | _______ |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | _______ |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | _______ |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | _______ |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | _______ |

Make up a sentence.  It should be 6-7 words long, should have spaces and may contain letters, numbers or symbols.

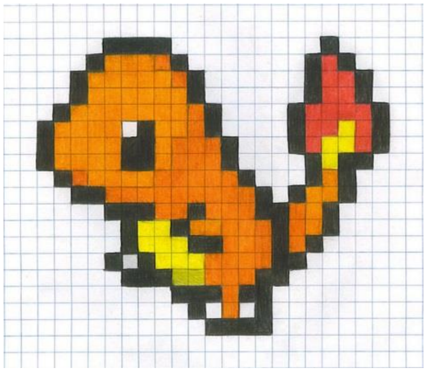
Generate the ASCII code for your sentence in the blocks on the graph paper provided.
Split up into pairs and exchange your code to interpret each other’s sentences.

<u>Question</u>

1.  Explain a pattern that was used to generate the ASCII binary code for numbers, upper and lower case letters and symbols.



# Activity 2
# Graph Paper
# Coding



By writing code, you can program each other to draw pictures. You will begin to understand what programming is really about. You will instruct each other to color in squares on a graph paper to reproduce an existing picture, then you will create a picture of your own and write the code.

In this activity, you will understand the difficulty of translating real problems into programs, learn that ideas that may seem clear can be misinterpreted by a computer (your classmate).

By using a series of symbols to represent an algorithm (instruction for completing a task), you will generate code to produce a picture and vice versa.

<u>Programming Key</u>

| | |
|---|---|
| → | Move 1 Square Forward |
| ← | Move 1 Square Backward |
| ↑ | Move 1 Square Up |
| ↓ | Move 1 Square Down |
| | Fill in Square with Color |

Time to Practice, using the picture below, generate a set of code that will reproduce the picture. When generating code, we will assume that we are always starting in the upper left block.

| | | |
|---|---|---|
| ⬜ | ⬛ | ⬜ |
| ⬛ | ⬜ | ⬛ |
| ⬜ | ⬛ | ⬜ |

What have we done? Let's analyze the process.

1. How many symbols of code did you use?

2. How did you arrange your symbols of code?

3. Should there be a standard process for coding a grid?

## More Advanced Pictures

| | | | | | | |
|---|---|---|---|---|---|---|
| | 🟥 | | | | 🟥 | |
| 🟥 | 🟥 | 🟥 | | 🟥 | 🟥 | 🟥 |
| 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 |
| 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 |
| | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | |
| | | 🟥 | 🟥 | 🟥 | | |
| | | | 🟥 | | | |

Generate a set of code to reproduce this picture.

## Analyze the Process

1. What makes this picture more complicated than the 1st one?

2. What suggestions can you make to streamline the process? List them.

Make Your Own

Create your own design in the grid below, then generate a set of code **on a separate** piece of paper. Exchange only your code (not the design) with someone in the class and have them reproduce your design while you try to reproduce theirs in the bottom grid.

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Activity 3 Coding Cups

Hello my Robotic Friends, in this activity you will figure out how to guide one another to accomplish a specific task without discussing it first. This activity will teach you how to make the connection between symbols and actions as well as the valuable skill of debugging.

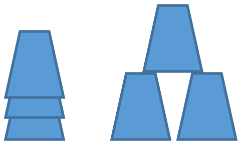
How do robots work? Do they actually think? NO! They are just machines that follow a set of instructions written by an intelligent human being (you).

You will split up into pairs and take turns being the robot and the programmer. For this activity you will generate a set of code that will instruct your "robot" to build a specific cup stack.

Our Robot Code

| | |
|---|---|
| → | Move Cup ½ Cup Width Right |
| ← | Move Cup ½ Cup Width Left |
| ↑ | Pick Up Cup (high enough to add to stack) |
| ↓ | Put Down Cup (set down on whatever is under it) |
| ↺ | Turn Cup Left 90º |
| ↻ | Turn Cup Right 90º |

Let's practice a simple one first. Generate a set of code to instruct a robot to build this stack.
The cups always start stacked.



## The Process

1. Pair up.
2. Each person will write their set of code first, then the robots will take over.
3. Split up the pairs so 1 of each pair is on opposite sides of the room, this is where the programmers will generate their code.
4. Each programmer will choose 1 of the 4 cup stacks to code.
5. When each programmer has finished coding their stack they can retrieve their robot.
6. Upon return, each programmer now becomes the robot. One at a time, each robot will receive the set of code from the programmer, read the symbols and translate them back into movements.
7. The programmer should watch for incorrect movements, then work together to debug their program before it is their turn to become the robot for their partner programmer.
8. **NO TALKING DURING THE ROBOT SESSION.** Let the robot try to follow the code.

## Analyze the Process

1. What was the toughest part of this activity? Explain why?

2. In what ways could you improve the code we have used? Explain?

# Activity 4 Physical Coding

You have been selected to program a robot to navigate an obstacle course. As robots have no intelligence of their own, they will be required to navigate the obstacle course based only on the code you have written, blindfolded. Therefore, you will need to be very specific with your code.

Your robot will be required to pick up a ball in the obstacle course, navigate through the course, then drop the ball in a basket at the end of the course.

The Process

1. Each student will be given 10 minutes to write a set of code that will navigate the robot through the course.
2. Students will pair up and take turns navigating the course blindfolded (acting as the robot) as the programmer gives the robot their code.
3. At each obstacle the programmer will make notes on their code that will be used to debug their code.
4. After the 1st robot has completed the obstacle course, switch roles and have the programmer now become the robot to navigate the course.

Use the symbols below to write your code for the obstacle course.

| | |
|---|---|
| → | Step Right |
| ← | Step Left |
| ↑ | Step Forward |
| ↓ | Step Backward |
| ↑↑ | Step Over |
| ↓↓ | Squat |
| ⤴ | Pick Up |
| ⤵ | Drop |

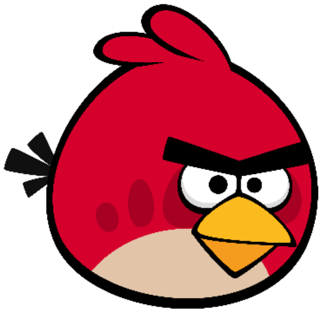
| Your Code | Your Debugged Code |
|---|---|
| | |

Analyze the Process

1. What issues did you encounter when you wrote your initial set of code? How could you fix them?

2. What issues did you encounter when you had your robot navigate the obstacle course? How could you fix them?

3. What issues did you encounter when you acted as the robot? How could you fix them?



# Activity 5
# Angry Bird
# Coding

In this final activity, you will use a computer simulation to generate code that will lead the Angry Bird through a series of increasingly difficult mazes that will lead to the demise of the Pig. You will watch an introductory video describing the process of coding and other short videos along the way that will teach you techniques to make your coding more efficient.

1. Go to the website: http://code.org
2. Click on the LEARN link at the top of the page.
3. Click on the HOUR OF CODE link under it.
4. Scroll down to Tutorials for Beginners and Click GO.
5. The initial introductory 2 minute video should automatically begin. Follow the instructions given to lead yourself through the Angry Bird Coding activity.