JAVA

**Whitespace** is one or more characters (such as a space, tab, enter, or return) that do not produce a visible mark or text. Java will ignore whitespace in code, but it is important to know how to use whitespace to structure code well.

There are 2 types of comments: single line-comments, one line comments that begin with two forward slashes (**/ /**); and multi-line comments that can span multiple lines, begging with **/\*** and ending with **\*/**.

The data type **int** is short for integer, which are all positive and negative numbers, including zero, between -2,147,483,648 and 2,147,483,647.

A Boolean is a data type that can only be either **true** or **false**.

The **char** data type is used to represent single characters. That includes the keys on a keyboard that are used to produce text. All char values must be enclosed in single quotes, like this: 'G'.

**Boolean** operators follow an order called precedence as follows:

! **Not** operator

&& **And** operator

|| **Or** operator

The **ternary** conditional statement provides a shortcut to write if/else statements in a single line of code. Is is compose of three parts: a boolean expression, a single statement that gets executed if the Boolean expression is true, and A single statement that gets executed if the Boolean expression is false.

A **switch** statement executes code blocks based on whether a block is equal to a specific value.

If //example { // is not follow by a ;

//code;

} else if //example {

//code;

} else {

//code;

}

variable2 = (variable1 > 0) ? //if statement : //else statement; //ternary conditional

variable1 = //example; //switch

switch (variable1) {

case //example: //code;

break;

case //example: //code;

break;

default: //example;

break;

}

-For:

The **for** loop repeatedly runs a block of code until a specified condition is met.

Java provides a shortcut to reduce the amount of code required to write the loop called the **for each loop**.

for (int varaible; //test condition; intVariable++) {

//code

}

-Array list:

The **ArrayList** stores a list of data of a specified type.

ArrayList<//data type> //name = new

ArrayList<//data type>();

ArrayListName.add(element); //adds elements to the ArrayList

ArrayListName.add(index, element); //adds an element in the index

ArrayListName.get(index); //gets the element at the index

ArrayListName.size(); //returns an int of how many elements

for (Integer //element : ArrayListName) { //for each element in ArrayList do code

//code;

}

-HashMaps:

Contains a set of keys and a value associated with each key.

HashMap<key type, value type> HashMapName = new //creates a HashMap

HashMap<key type, value type>();

HashMapName.put(key, value); //adds a key, value pair to the HashMap

HashMapName.get(key); //get the value for that key

HashMapName.size(); //gets how many pairs of key, values are

HashMapName.keySett(); //provides a list of the keys

-Class:

A class is a set of instructions that describe how a data structure should behave.

A class constructor will allow you to create instances and set some information about the class.

An instance variable has specific details that we want the class to include.

To use a class you have to create an instance of the class, know as object in Java.

A method is a pre-defined set of instructions. Methods are declared within a class.

The **void** keyword (which means "completely empty") indicates to the method that no value is returned after calling that method. If we do want the method to return a value after it finishes running, we can specify the **return** type.

A parameter are values that can be specified when creating an object or calling a method.