Class Zero

public static void main(String args[])

This is the main driver of the program. The program will not be able to run without this method. Whenever your press run, the system will always prosecute the code in here first. The details of this method will be covered later. For now, memorize it clearly so that you can set up your program whenever needed.

System.out.print();

This is the main way to output message to the console. The message should be bounded with double quotation marks to represent String type. Anything within the parenthesis without double quotation marks will be treated as variables and if the compiler could not find the exact variable, compile error would take place.

System.out.println();

This works the same as System.out.print(), but it will print out a new line character after the message.

Escape Sequence

The escape sequence is a character preceded by a backslash (\) which has special meaning to the compiler. It only works if it is within a String.

For APCS, the ones you need to know :

- \n insert a new line in the text at this point
- \t insert a tab in the text at this point
- \' insert a single quote in the text at this point
- \" insert a double quote in the text at this point

Primitive Variables

Primitive variables are the simplest data types to store information.

The primitive types you need to know for APCS are(listed from smaller data type to bigger data type)

- boolean stores true or false
- char stores a single character, bounded with single quote
- int stores an integer
- double stores numbers with floating point(decimals)

Automatic conversion(promotion) of primitive data types takes place when you put a smaller data into a bigger data type. For example, these are valid assignments:

- `int i = 5;  double d = i;  //d will be 5.0000...`
- `char c = 'c';  int i = c;` (The meaning of this will be covered)

If you want to perform the assignment the other way, you have to put a type cast in the front. For example:

- `double d = 5.999;  int i = (int)d;  //i will be 5`

If no value is given at the initialization, default value will be automatically assigned. The default value for **int is 0, double is 0.0, char is '\0', String is "", and boolean is false.**

Operators

The operators you will need to know are + - * / %(mod). These are pretty obvious except % means the remainder of the division between the left operand and the right operand. So 14 % 3 will give you 2. These operators have a higher precedence than the assignment operator (=), so calculation will be performed first and the value will be assigned to certain variable.

- `int i = 5+3*4;  //i will be 17`
- `i /= 3;  //this is equal to i = i / 3; this works for + - % * /`

- i++; //this is equal to i = i+1; or i += 1; or ++i

The difference between i++ and ++i is that ++i (pre increment)will always prosecute before any other operations while i++(post increment) will prosecute after other operations.

- int i = 4; int j = i++; //j is **4**, i is 5
- int i = 4; int j = ++i; //j is **5**, i is 5

&& and || are operators that deal with true/false

- true && true //true
- false && true //false
- true || false //true
- false || false //false

Something to note about && and || is that for &&, if the left operand is false, there is no need to check the right operand, so the compiler **will skip the evaluation** of the second operand. For ||, if the left operand is true, the compiler **will skip the evaluation** of the second operand for the same reason.

String

String is a special data type in Java. The most common way to use it is using the double quotation mark “”. Any characters that is bounded with double quotation mark is considered a **String literal** and acts the same way as primitive types.

String is immutable, which means once a String is created, it could not be changed.

String works with the operator + which allows String literals to connect with each other to form a new String(String concatenation). String concatenation works with other primitives as well in a way that any primitive + String literal will automatically convert that primitive type to a String literal and performs String concatenation.

- `int i = 5;  int j = 6;  String s = j + “” + i + “hi";  //s is` **`65hi`**

Array

In Java, array is treated as an object. It is the only data structure that could store both primitives and objects. All other data structures can only store objects. Note that the length of array is immutable once it is created.

Array can store a list of datas with the same data type.

Some sample initializations of arrays:

- `int[] arr1 = new int[5]; //length is 5, all values are 0`
- `double arr2[] = new double[7]; //length is 7, all values are 0.0`
- `String[] arr3 = {"See", "You", "Next", "Year"}; //length is 4`
- `boolean[] arr4 = new boolean[1] //length is 1, value is false`