Daniel Gopal

**Level 1: Hello World**

1. Install and run the Dr. Java development environment from the network drive:

4Students\OUT\Nestor\Java\drjava-beta….exe

2. Edit, compile, and run the “Hello World” program as described online at:<http://www.cs.utsa.edu/~cs1063/resources/gettingstarteddrjava.html>

3. Explain the function of each line in the “Hello World” program below. The following reference will help you answer this question:<https://www.learnjavaonline.org/>

Line 1 “public class HelloWorld {“ defines the name of the java file being “HelloWorld” and is a public class type that can be used within other java classes. This bracket “{“ opens an area to store code and then it is closed using “ } “. The curly brackets are used in bodies of loops, methods and classes like this one. Where it works like a box/container storing specific code for that type in which creates what is known as code blocks. Line 2 “public static void main(String[] args) { “ allows for the main method must be declared as public, static, and void. “public” allows for any other class can access this. Static is an access modifier that is associated with the class, not a specific instance (object) of that class. This means that you can call a static method without creating an object of main/class. Void is a return type is used when you do not want a function to return anything as it doesn't return any value. Line 3 that is “System.out.println("Hello World");” allows for the String type written in the quotation marks to be displayed. “System” is a pre-designed class that holds various methods and variables that are used. “Out” is a variable within “System” that represents the output of the program . The “print” allows for the string to be printed on the line and in this case the “println” moves the cursor to a new line and prints the string. Line 4 has a curly bracket “ } “ that closes the java class and line 5 has another to close main.

**Level 2: Variables and Types**

1. Read through the “Variables and Types” tutorial at<https://www.learnjavaonline.org/>

2. Explain the difference between “byte’, ‘short’, ‘int’, and ‘long’.

All these types are number based. A byte has 1 byte, short has 2 bytes, int has 4 bytes and long has 8 bytes.

The ‘byte’ data type is an 8-bit signed two's complement integer. It has a minimum value of -128 and a maximum value of 127. The ‘byte’ data type can be useful for saving memory in large arrays, where the memory savings actually matters.

The ‘short’ data type is a 16-bit signed two's complement integer. It has a minimum value of -32,768 and a maximum value of 32,767. As with byte, the same applies as you can use a short to save memory in large arrays, and situations where the memory savings actually matters.

The ‘int’ data type is a 32-bit signed two's complement integer, which has a minimum value of -231 and a maximum value of 231-1. In Java SE 8 and later, you can use the int data type to represent an unsigned 32-bit integer, which has a minimum value of 0 and a maximum value of 232-1.

The ‘long’ data type is a 64-bit two's complement integer. The signed long has a minimum value of -263 and a maximum value of 263-1. In Java SE 8 and later, you can use the long data type to represent an unsigned 64-bit long, which has a minimum value of 0 and a maximum value of 264-1. You can use this data type when you need a range of values wider than those provided by int. It also contains methods like ‘compareUnsigned’ and ‘divideUnsigned’ that support arithmetic operations for unsigned long.

3. Which type would you use most often for a whole number?

The ‘int’ type that is short for integer.

4. When would you use the type ‘long’?

You can use this data type when you need a range of values wider than those provided by int.

As Java’s ‘int’ type is designed to use 32 bit memory while as the long type does 64 bit. You can use this data type when you need a range of values wider than those provided by int.

5. How is the type ‘float’ different from the type ‘int’?

‘float’ is used to define a variable with a fractional value while as ‘int’ defines a whole number.

6. How is the type ‘float’ different from the type ‘double?

Though both the float and double datatype are used to represent decimals based numbers in Java, a double data type is known to be more precise than float. A double variable can provide precision up to 15 to 16 decimal points as compared to float precision of 6 to 7 decimal digits.

The float data type is a single-precision 32-bit IEEE 754 floating point. Its range of values is quite vast, but is specified in the Floating-Point Types, Formats, and Values section of the Java Language Specification. The float is used instead of the double if you need to save memory in large arrays of floating point numbers.

7. When would you use the type ‘char’ and when would you use the object “String”?

Char is a single alphabet type while as a String is a set or sequence of characters.

8. What is an easy to concatenate (append) strings together?

Create an array as it stores more than 1 of the same variable/types such as string into a set or collection. A String can also easily concatenate with other strings just by storing the string variables already created into another.

For example:

// Create a string with a constructor

String s1 = new String("Who let the dogs out?");

// Just using "" creates a string, so no need to write it the previous way.

String s2 = "Who who who who!";

// Java defined the operator + on strings to concatenate:

String s3 = s1 + s2;

‘s1’ and ‘s2’ are the created string variables. They are then stored/added into a another String variable name ‘s3’.

9. What is the type ‘boolean’ used for?

The Boolean type can defined within two choices being commonly true or false, yes or no, 1 or 0. It is used in a instance of finding an outcome of something and is a simple conditional type in code as opposed to something like if statements.

10. How is the Java ‘boolean’ different from the Python ‘boolean’ type?

The python boolean type is called when used “bool([x]” While as in java you simply use “boolean”. Python has boolean expressions such as the "and" and "or" operators that would then be used to create multiple conditions and outcomes.

11. Write a program that prints ‘H3110 w0r1d 2.0 true’. You must use each of the variables discussed above.

//Write a program that prints ‘H3110 w0r1d 2.0 true'

public class MixedVariableOutput {

public static void main(String[] args) {

// TODO Auto-generated method stub

//Create variables that correspond with parts of the output

short three = 3;

int one = 1;

byte zero = 0;

char space = ' ';

float two = 2.0f;

boolean t = true;

//String that prints characters and concatenates variables in order

String Output = "H" + three + one + one + zero + space + "w" + zero + "r" + "l" + "d" + space + two + space + t;

System.out.println(Output);

}

}

12. How does your program compare to the solution provided at the bottom of the Variables and Types tutorial?

The solutions posted on the tutorial site provides a brief on how the String can be stored in others Strings as String ‘s1’ is created as a string variable, in addition to another string variable named s2 in which they both correspond as separate sentences. They are then linked together in another String named s3 to print“"Who let the dogs out? Who who who who!". My program utilises the different java types such as int, string, short, byte, char, float and boolean. By which are used to create individual variables that are then stored in a String variable similarly to the solution on LearnJavaonline as an orderly set and outputted to print “H3110 w0r1d 2.0 true”

The output is a return type that is boolean with an int variable represented in a if conditional statement to return a boolean output of true or false. My program utilizes the different java types such as int, string, short, byte, char, float and boolean. By which are used to create individual variables that are than stored in a String variable as an orderly set and outputted to print “H3110 w0r1d 2.0 true”

**Level 3: Conditionals and Arrays**

1.Read through the “Conditionals” tutorial at https://www.learnjavaonline.org/

2. For each Boolean operator (“>”, “<”, “<=”, “>=”, “!=”, “==”), write an if…else… statement that demonstrates what they do.

//This program demonstrates the >= (greater than or equal to) and the <= (less than or equal to) operator

// ‘>=’ Checks if the value of left operand is greater than or equal to the value of the right operand

// ‘<=’ Checks if the value of left operand is less than or equal to the value of right operand

//Like in this program where the user has to input their grade and they are then given an output/result based on it.

//Ask the user for a mark

System.out.print("Please type a mark: ");

int mark = sc.nextInt();

if (mark >= 90)

{

System.out.println("Level 4");

}

else if (mark <= 85)

{

System.out.println("Level 4-");

}

else if (mark >= 80)

{

System.out.println("Level 3+");

}

else if (mark <= 75)

{

System.out.println("Level 3");

}

else if (mark >= 70)

{

System.out.println("Level 3-");

}

else if (mark <= 65)

{

System.out.println("Level 2+");

}

else if (mark >= 60)

{

System.out.println("Level 2");

}

else if (mark <= 55)

{

System.out.println("Level 2-");

}

else if (mark >= 50)

{

System.out.println("Level 1+");

}

else if (mark >= 50)

{

System.out.println("Level 1");

}

else

{

System.out.println("Level R");

}

}

//This program demonstrates the > (greater than) and the < (less than) operator

// ‘>’ Checks if the value of left operand is greater than the value of right operand

// ‘<’ Checks if the value of left operand is less than the value of right operand

//Like in this program where the user has to input their score and it checks to see if there … score is greater than or less than.

//Highest Score for arcade game

int BestScore = 45606;

//Ask the user for their score

System.out.print("Please your score: ");

int Score = sc.nextInt();

if (Score > BestScore)

{

System.out.println("That is a New personal best! ");

}

else if (Score < BestScore)

{

System.out.println("Try again!");

}

}

}

//This program demonstrates the == (equal to) and the != (not equal to) operator

// ‘==’ Checks if the values of two operands are equal

// ‘!=’ Checks if the values of two operands are not equal

//This program is a simple concept of how the boolean type works when declared as . . true or false and is then used in a if statement to see if it is true or false.

// Declare a variable of type boolean

boolean value = true;

// If value is true

if (value == true) {

System.out.println("It's true");

}

// Else

else if (value != true)

{

System.out.println("It's false");

}

}

}

3.Explain the function of the “||” operator.

Called the Logical OR Operator. If any of the two operands are non-zero, then the condition becomes true.

4.Write an if…else… statement that demonstrates how the “||” operator works.

This program checks for a telemarketer 2 digit pattern being 8\*\*9 or 9\*\*8.

//Read four digits of a phone number

int d1 = sc.nextInt();

int d2 = sc.nextInt();

int d3 = sc.nextInt();

int d4 = sc.nextInt();

if ((d1 == 8)|| (d1==9))

{

//Telemarkter output

System.out.println("ignore");

}

else if ((d4 == 8) || d4==9))

{

System.out.println("Answer");

}

}

}

5.Explain the function of the “&&” operator.

Called Logical AND operator. If both the operands are non-zero, then the condition becomes true.

6.Write an if…else… statement that demonstrates how the “&&” operator works.

This program checks for a telemarketer 4 digit pattern being 8\*\*9 or 9\*\*8 and most importantly the same 2nd and 3rd digits.

//Read four digits of a phone number

int d1 = sc.nextInt();

int d2 = sc.nextInt();

int d3 = sc.nextInt();

int d4 = sc.nextInt();

if (((d1 == 8)|| (d1==9))&&

((d4 == 8)|| d4==9)&&

(d2 == d3))

{

System.out.println("Ignore");

}

else

{

System.out.println("Answer");

}

}

}

7.Explain how the “? … : …” operator works.

Called the ternary operator it is the only conditional operator that takes three operands. Java ternary operator is a one liner replacement for if-then-else statements. The first operand within this operator contains a boolean or a statement with boolean result. If the first operand is true then the ternary operator returns second operand else it returns third operand. A ternary operator allows you to assign one value to the variable if the condition is true, and another value if the condition is false.

8.Explain why the statement: String(“wow”) == String(“wow”) returns false.

== checks if both objects point to the same memory location whereas .equals() evaluates to the comparison of values in the objects. When we use == operator in the comparison the result is false because both have different addresses in memory. “==” is a comparison operator which is mostly useful for comparing boolean, numbers and other primitives but when it comes to string objects it was not designed for that purpose and that is why the equals() method class was created for.

9.Explain why the statement: String(“wow”).equals(String(“wow”)) returns true.

In Java, string equals() method compares the two given strings based on the data/content of the string. If all the contents of both the strings are same then it returns true. If all characters are not matched then it returns false. So basically, when we are comparing 2 String objects by .equals() operator you are checking that both objects contain the same data or not.

10.Read through the “Arrays” tutorial at https://www.learnjavaonline.org/

11.Create some code to do the following:

a.Define an array of integers.

b.Initialize the array to hold 4 values.

c. Add 4 integers to the array.

d.Print out the length of the array.

e. Print out the value of the last element of the array.

f. Add the elements of the array together and print out the result.

**Method for Int Array:**

class Main {

public static void main(String[] args) {

Int [ ] list;

list = new int[ ]{1,3,5,7};

for (int i=0; i < list.length; i++)

{

System.out.print(list[i] + " ");

}

int ArrayLength = list.length;

int LastNumber = list [list.length-1];

int sum = 0;

for (int i=0; i<list.length; i++)

{

//Update the total

sum += list[ i ];

}

System.out.println(" "); //Spacer to print output on next line

System.out.println("The length of the array is " + ArrayLength);

System.out.println("The last element in the array is " + LastNumber);

System.out.print("The sum/total is " + sum);

}

}

12. Repeat the above question for an array of Strings.

**Method for String:**

public class Main {

public static void main(String[] args) {

//Create a String

String List = "Pear";

int length = List.length();

//Finds the last character in the String

char LastCharacter = List.charAt(List.length() - 1);

System.out.println(List);

System.out.println("The length of the array is " + length);

System.out.println("Last word of the sentence is : " + LastCharacter );

}

}

**Method for Char Array:**

public class CharArrayLength {

public static void main(String[] args) {

//Create an array of characters

char[] List = new char[]{'B','l','u','e'};

//Print the array

for (int i = 0; i < List.length; i++)

{

System.out.print(List[i]);

}

int ArrayLength = List.length; //The length of the array

char LastCharacter = List[List.length-1]; //The last element in array

System.out.println(" "); //Spacer to print on next line

System.out.println("The Length of the array is: " + ArrayLength);

System.out.println("The last element in the array is " + LastCharacter);

}

}