**Unit 2:** Fundamental DataTypes & Decisions

Answer the questions and complete the programs in preparation for the end of Unit exam

**Review Questions:**

1. What is encapsulation? Why is it useful?  
   *It wraps variables and data as a single unit so the data is specific to the class.*
2. What is the *this* reference? Why is it used?  
   *It refers to a current instance of a variable so it stays specific to a method.*
3. Suppose you want to extend the car viewer program given in the previous unit example code to show a suburban scene, with several cars and houses. Which classes do you need?  
   *GraphicsRunner, BigHouse, and the car viewer program.*
4. What is wrong with the following sequence of statements?

int mystery = 1;

mystery = mystery + 1;

int mystery = 1 – 2 \* mystery;  
*mystery is initiated twice.*

1. Write pseudocode for a program that reads a word and prints the first character, the last character, and the characters in the middle. For example, if the input word was “Harry”, the program would print H y arr.  
   *import java.util.Scanner;*

*class Main {*

*public static void main(String[] args) {*

*Scanner keyboard = new Scanner(System.in);*

*String str = keyboard.nextLine();*

*System.out.print(str.charAt(0) + " " + str.charAt(str.length()-1) + " " +str.substring(1,str.length()-1));*

*}*

*}*

1. Complete the following truth table by finding the truth values of the Boolean expressions for all combinations of the Boolean inputs p, q, and r

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **p** | **q** | **r** | **(p && q) || !r** | **!(p && (q || r))** |
| f | f | f | *f* | *t* |
| f | f | t | *t* | *t* |
| f | t | t | *t* | *t* |
| f | t | f | *f* | *t* |
| t | t | t | *t* | *f* |
| t | t | f | *t* | *f* |
| t | f | f | *f* | *t* |
| t | f | t | *t* | *f* |

**Exercises:**

1. Write a class SavingsAcount that is similar to the BankAccount class, except that it has an added instance variable *interest*. Supply a constructor that sets both the initial balance and the interest rate. Supply a method addInterest(with no eplicit parameter) that adds the interest to the account. Write a SavingsAccountTester class that constructs a savings account with an initial balance of $1000 and an interest rate of 10%. Then apply the addInterest method and print the resulting balance.

*public double addInterest(double balance, double interest) {*

*return balance + .01\*interest\*balance;*

*}  
class SavingsAccountTester {*

*public static void main(String[] args) {*

*double balance = 1000;*

*double interest = 10;*

*SavingsAccount lab = new SavingsAccount();*

*System.out.println(lab.addInterest(balance, interest));*

*}*

1. Write a program that displays the Olympic rings. Color the rings with the Olympic colors. Provide classes OlympicRing, OlympicRingViewer, and OlympicRingComponent
2. Write a program that prompts the user for two intergers and then prints
   1. The sum
   2. The difference
   3. The product
   4. The average
   5. The distance (absolute value of the distance)
   6. The maximum (largest of the two)
   7. The minimum (smallest of the two)

*import java.util.Scanner;*

*import java.math.\*;*

*class Main {*

*public static void main(String[] args) {*

*Scanner keyboard = new Scanner(System.in);*

*System.out.println("Input first integer: ");*

*int int1 = keyboard.nextInt();*

*System.out.println("Input second integer: ");*

*int int2 = keyboard.nextInt();*

*int sum = int1 + int2;*

*int difference = int1 - int2;*

*int product = int1 \* int2;*

*int average = (int1 + int2)/2;*

*int distance = 0;*

*if (difference>0){*

*distance = difference;*

*}*

*else{*

*distance = -1 \* difference;*

*}*

*int max = 0;*

*int min = 0;*

*if (int1>int2){*

*max = int1;*

*min = int2;*

*}*

*else{*

*max = int2;*

*min = int1;*

*}*

*System.out.println("Sum: " + sum);*

*System.out.println("Difference: " + difference);*

*System.out.println("Product: " + product);*

*System.out.println("Average: " + average);*

*System.out.println("Distance: " + distance);*

*System.out.println("Max: " + max);*

*System.out.println("Min: " + min);*

*}*

*}*

1. Write a program that reads in three integers and prints “in order” if they are sorted in ascending or descending order, or “not in order” otherwise. For example,
   1. 1 2 5 prints: in order
   2. 1 5 2 prints: not in order
   3. 5 2 1 prints: in order
   4. 1 2 2 prints: in order

*import java.util.Scanner;*

*class Main {*

*public static void main(String[] args) {*

*Scanner keyboard = new Scanner(System.in);*

*int int1 = keyboard.nextInt();*

*int int2 = keyboard.nextInt();*

*int int3 = keyboard.nextInt();*

*if ((int1 <= int2 && int2 <= int3)||(int3<=int2 && int2 <= int1)){*

*System.out.println("in order");*

*}*

*else{*

*System.out.println("not in order");*

*}*

*}*

*}*