**CSCI 1100 – Summer 2015**

**Assignment 4**

**Assignment 4 – Due Thursday June 10th (11:00pm – night time)**

**Submit on Moodle**

**Name: Liam Gowan**

**Student ID:** B00673126

**Assignments are to be your own work. If you have questions, you can ask your Instructor, course TAs or TAs in the learning centre.**

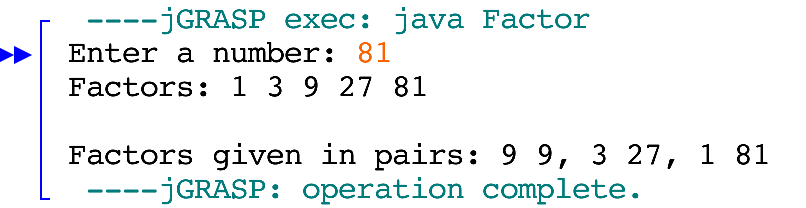
|  |  |  |
| --- | --- | --- |
| **Declaration: Please complete this declaration** | | |
| 1 | “This document is entirely my own work.” | Yes/no |
| 2 | I obtained help to complete this document. | Yes/no. If Yes, give Details. |
| 3 | This document contains some material copied or cut and pasted from the internet or another document or file or program. | Yes/no. If Yes, give details and provide references. |

**Exercise 1.** Write a program that asks a user to enter a number. Your program will then call a method called *factors* where you will pass the user entered number. The method *factors* will print out all the positive factors of that number. See the sample code below. You should have 3 test cases (all different from the sample).

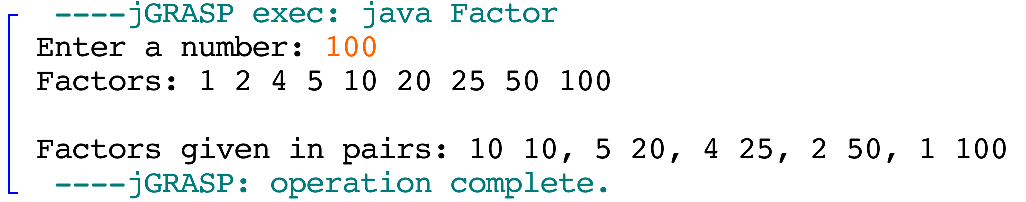
**Program:**

import java.util.Scanner;  
  
public class Factor{  
 public static void main(String[] args){  
 Scanner input = new Scanner(System.in);  
 System.out.print("Enter a number: ");  
 int num = input.nextInt();  
 factors(num); //calls factors() method  
   
 }  
   
 public static void factors(int num){  
 System.out.print("Factors: ");  
 /\*When i is less than or equal to num, the program will   
 test to see if the remainder of num/i is 0, and if it is   
 it will print i, the factor\*/  
 for(int i=1; i<=num; i++){  
 if(num%i == 0){  
 System.out.print(i + " ");  
 }  
 }  
   
 /\*While not required, this program will also display  
 the two factors that multiply to give num in a way that  
 is easier for the user to read, using commas to separate   
 the pairs   
 The outer for loop runs each interation while i is less   
 than or equal to num, and the inner for loop will run while j   
 less than or equal to i. The inner for loop checks to see  
 if i\*j is equal to num, and whether or not i is equal to num.  
 If i\*j = num, it will print j and i, and it will only place a  
 comma if i does not equal num, meaning there are still more   
 numbers to test. \*/  
 System.out.print("\n\nFactors given in pairs: ");  
 for(int i=1; i<=num; i++){  
 for(int j=1; j<=i; j++){  
 if(i\*j == num && i != num)  
 System.out.print(j + " " + i + ", ");  
 else if(i\*j == num && i == num)  
 System.out.print(j + " " + i);  
   
 }  
 }   
 }  
}

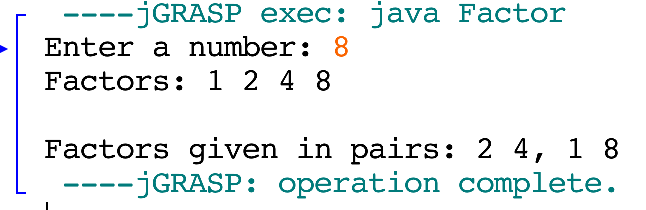
**Test 1.**

****

**Test 2.**

****

**Test 3.**

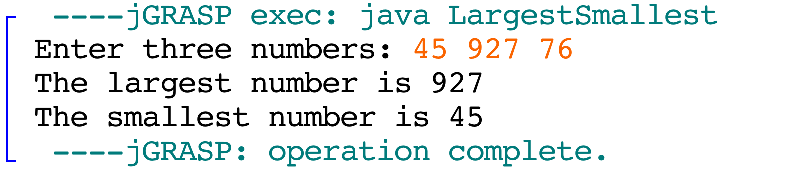
****

**Exercise 2:** Write a program that will ask the user to enter three numbers. Your program will then call the method *largest* where you will pass in all three integers. The method *largest* will return the largest integer (note do not use the Math.max method in your code – instead you will create your own method that will determine the largest number of three numbers). Your program will then call another method called smallest that takes in three integers and returns the smallest integer (again do not use the Math.min method in your answer). Assume that all three numbers are different. See the sample below. You should have 3 test cases (all different from the sample).

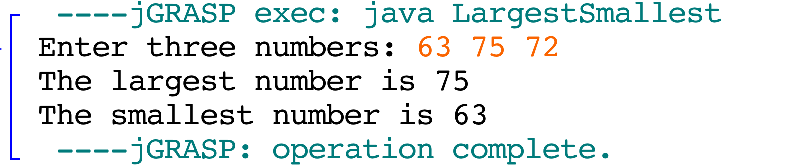
**Program:**

import java.util.Scanner;  
  
public class LargestSmallest{  
 public static void main(String[] args){  
 int num1, num2, num3, max, min;  
 Scanner input = new Scanner(System.in);  
 //prompts user and assigns 3 numbers  
 System.out.print("Enter three numbers: ");  
 num1 = input.nextInt();  
 num2 = input.nextInt();  
 num3 = input.nextInt();  
   
 max = largest(num1, num2, num3); //calls largest() method  
   
 min = smallest(num1, num2, num3);//calls smallest() method  
   
 System.out.println("The largest number is " + max);  
 System.out.print("The smallest number is " + min);  
 }  
   
 public static int largest(int num1, int num2, int num3){  
 /\*method tests to see which number is the largest, and then  
 returns that number \*/  
 if(num1 > num2 && num1 > num3)  
 return num1;  
 else if(num2 > num1 && num2 > num3)  
 return num2;  
 else if(num3 > num1 && num3 > num2)  
 return num3;  
 else  
 return num1;  
 }  
   
 public static int smallest(int num1, int num2, int num3){  
 /\*method tests to see which number is the smallest, and then  
 returns that number \*/  
 if(num1 < num2 && num1 < num3)  
 return num1;  
 else if(num2 < num1 && num2 < num3)  
 return num2;  
 else  
 return num3;  
 }  
}

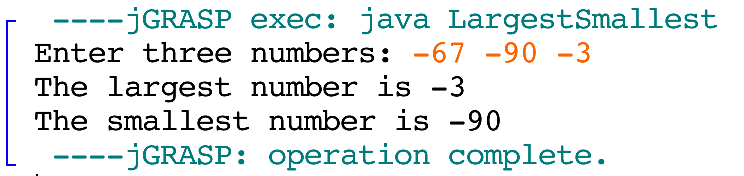
**Test 1.**

****

**Test 2.**

****

**Test 3.**

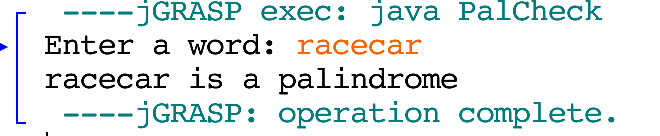


**Exercise 3.** Write a program that asks a user to enter a word. The program then calls a method called *palindrome* that takes the entered word as a parameter. The method *palindrome* tests to see if the word is the same spelled forward as backward. It returns true if the word is spelled the same backwards as forward. See the sample output below. You should have 3 test cases (all different from the sample).

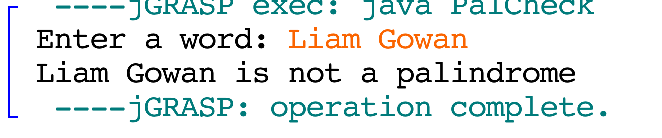
**Program:**

import java.util.Scanner;   
  
public class PalCheck{  
 public static void main(String[] args){  
 String word = "";  
 Scanner input = new Scanner(System.in);  
 System.out.print("Enter a word: ");  
 word = input.nextLine();  
 boolean palState = palindrome(word); //calls palindrome()  
 /\*if palState is true, program will say word is a   
 palindrome, otherwise it will say it is not \*/  
 if(palState)  
 System.out.print(word + " is a palindrome");  
 else  
 System.out.print(word + " is not a palindrome");  
 }  
   
 public static boolean palindrome(String word){  
 /\*The following code converts 'word' to a char array,  
 and then sets wordBackwards to the char array, only in reverse.  
 \*/  
 char lettersInWord[] = word.toCharArray();   
 String wordBackwards = "";  
 for(int i = 0; i<word.length(); i++)  
 wordBackwards += "" + lettersInWord[word.length()-1-i];   
 /\*if word is equal to wordBackwards (meaning it is a palindrome),  
 it will return true, otherwise false \*/   
 if(word.equals(wordBackwards))  
 return true;  
 else  
 return false;  
 }  
}

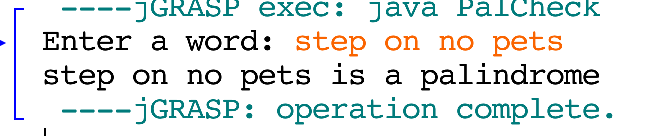
**Test 1.**

****

**Test 2.**

****

**Test 3.**

****

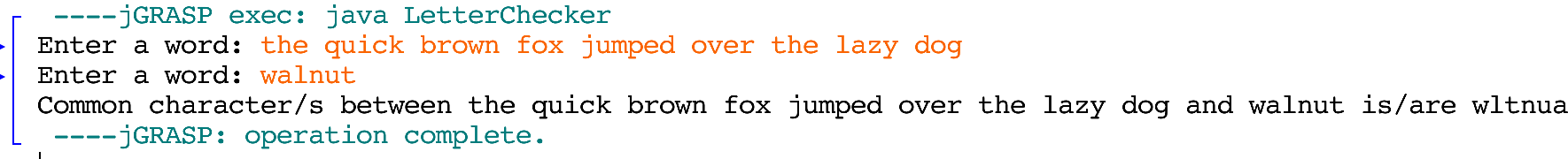
**Exercise 4.** Write a program that asks a user to enter in two words. The program then calls a method called *join* that takes two strings as parameters and returns a string consisting of the characters that are in both of the strings. [You should look up the String method lastIndexOf (char) to help you.] Repetitions must be excluded but the order is not important. For example if the Strings are abccd and ceccaa the value of the String returned by the method would be ac (order of characters not significant). See Sample output below. You should have at least 3 test cases (different from the sample).

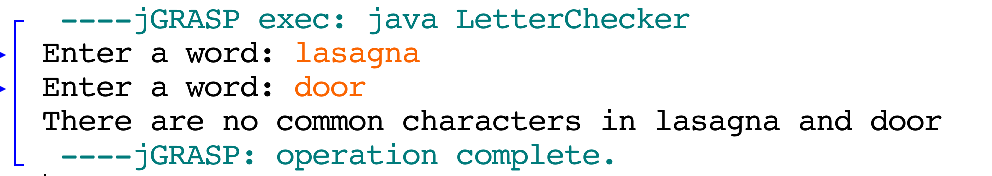
**Program:**

import java.util.Scanner;  
  
public class LetterChecker{  
 public static void main(String[] args){   
 String word1, word2;  
 Scanner input = new Scanner(System.in);  
 //Prompts user for two words, saves them to String variables  
 System.out.print("Enter a word: ");  
 word1 = input.nextLine();  
 System.out.print("Enter a word: ");  
 word2 = input.nextLine();   
 String lettersInBoth = join(word1, word2); //calls join()  
   
 /\*If there are common characters, it will display them,   
 otherwise it will display that there are no common  
 characters \*/  
 if(!lettersInBoth.equals("")){  
 System.out.print("Common character/s between " + word1  
 + " and " + word2 + " is/are " + lettersInBoth);  
 }  
 else{  
 System.out.print("There are no common characters in " + word1 +  
 " and " + word2);

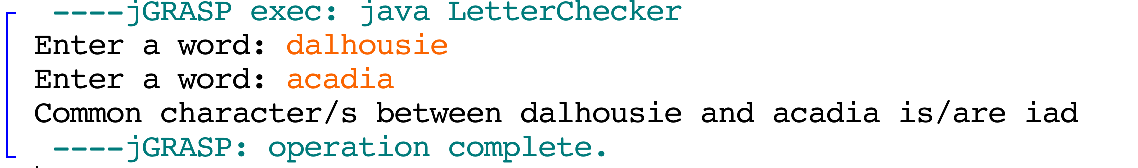
}  
 }  
   
 public static String join(String word1, String word2){  
 int longerWord, shorterWord;  
 String longWordString, shortWordString;  
   
 /\*Assigns int and String values to prepare for  
 rest of program \*/  
 if(word1.length()>=word2.length()){  
 longerWord=word1.length();  
 shorterWord=word2.length();  
 longWordString = word1;  
 shortWordString = word2;  
 }  
 else{  
 longerWord=word2.length();  
 shorterWord=word1.length();  
 longWordString = word2;  
 shortWordString = word1;  
 }  
 String commonLetters = "";  
   
 //Adds common characters to commonLetters String  
 for(int i = 0; i<longerWord; i++){  
 for(int j = 0; j<shorterWord; j++){  
 if(longWordString.charAt(i) == shortWordString.charAt(j) && word1.length()>=word2.length())  
 commonLetters += word1.charAt(i);  
   
 else if(longWordString.charAt(i) == shortWordString.charAt(j) && word1.length()<word2.length())  
 commonLetters += word1.charAt(j);  
 }  
 }  
   
 //Ensures that there is no repetition of characters  
 for(int i=0; i<commonLetters.length(); i++){  
 for(int j=0; j<commonLetters.length(); j++){  
 if(commonLetters.charAt(i) == commonLetters.charAt(j)){  
 String holder = "" + commonLetters.charAt(i);  
 commonLetters = commonLetters.replaceAll(holder, "");  
 commonLetters += holder;   
 }  
 }  
 }  
 return commonLetters; //returns commonLetters  
 }   
}

**Test 1:**

****

**Test 2: **

**Test 3:**

****

**Exercise 5.** Write a program that has two arrays of the same size filled with integers. The program prints out the sum of the two arrays. You do not need to use methods for this program (just code in the main method). See the below output to see how the program works. Test this with two different sets of numbers.

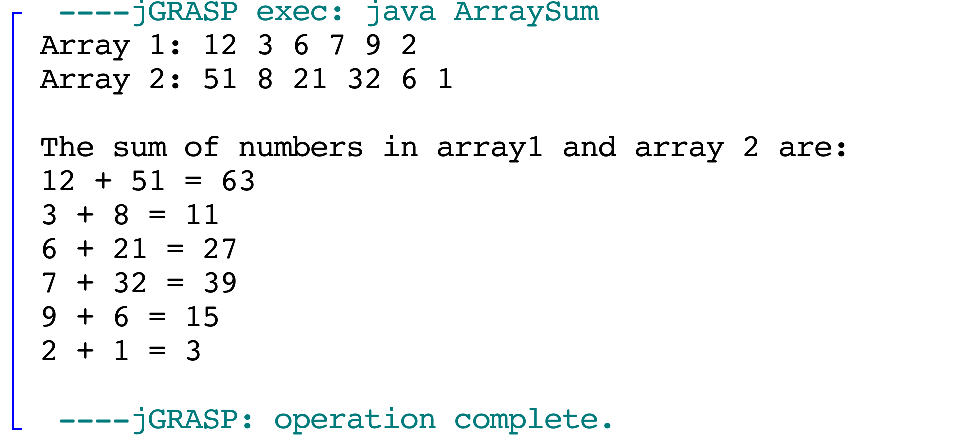
**Program:**

public class ArraySum{  
 public static void main(String[] args){  
 /\*As specified in instructions, two arrays have been made   
 without asking for user input \*/  
 int[] array1 = {12, 3, 6, 7, 9, 2};  
 int[] array2 = {51, 8, 21, 32, 6, 1};  
 System.out.print("Array 1: ");  
 for(int i=0; i<array1.length; i++) //lists numbers in array1  
 System.out.print(array1[i] + " ");  
 System.out.println();  
 System.out.print("Array 2: ");  
 for(int i=0; i<array2.length; i++) //lists numbers in array2  
 System.out.print(array2[i] + " ");  
 System.out.println("\n\nThe sum of numbers in array1 and array 2 are: ");  
 for(int i=0; i<array1.length; i++){ //adds numbers, shows result  
 System.out.println(array1[i] + " + " + array2[i] + " = " +  
 (array1[i] + array2[i]));  
 }  
 }  
}

**Test 1.**

**Arrays have been set to the following:**

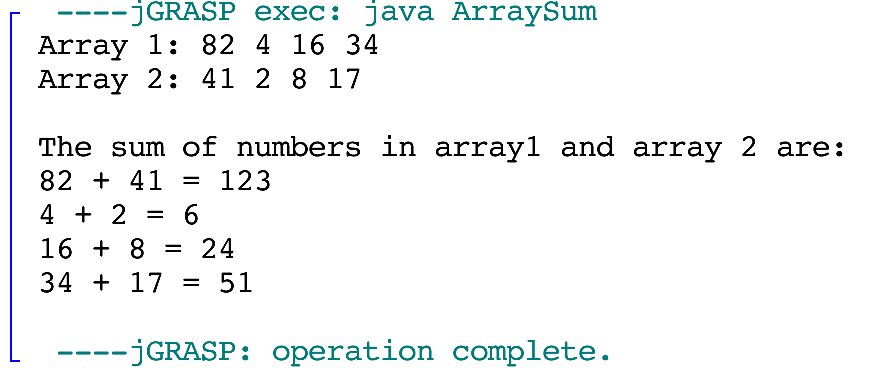
int[] array1 = {12, 3, 6, 7, 9, 2};  
int[] array2 = {51, 8, 21, 32, 6, 1};

****

**Test 2.**

**Arrays have been set to the following:**

int[] array1 = {82, 4, 16, 34};  
int[] array2 = {41, 2, 8, 17};

****