| SHADED logo | INSTITUTE OF TECHNOLOGY TRALEE  AUTUMN EXAMINATIONS AY 2013-2014  **Introduction to Programming**  **PROG61001**  **CRN 43824** |
| --- | --- |

**External Examiner**:

**Internal Examiner**: Mr. J. Brosnan

**Duration**: 2 Hours

**Instructions to Candidates:** You may attempt *any* 3 questions. All questions carry equal marks. Show all workings clearly as you may lose marks otherwise.

**Q1.**

**(a)**

Write a Java program that begins with a single-line comment which displays the name of the program, in this case **Ireland.java**. It should then have a multi-line comment which explains very briefly what the purpose of the program is.

Next the program should **create four variables** to store the following details for a particular country:

Country Name: Ireland

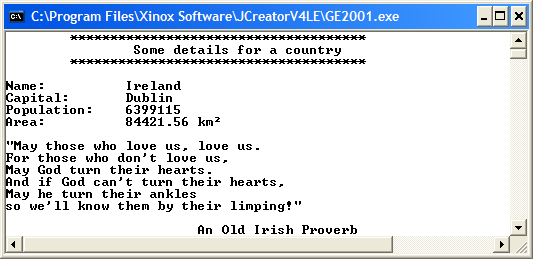
Capital: Dublin

Population: 6,399,115

Area: 84,421.56 km2

Note that you should use numeric variables wherever possible when storing the information above and give the variables the **most meaningful names** you can. Note that you are just setting (initializing) variables here so **no user input** whatsoever is required in this program.

Once the details above have been set, your program should produce **exactly** the output indicated on the screenshot below, including all spaces, tabs, quotes and blank lines. Your program should use a **single println()**to display the information shown. Naturally, the variables referred to earlier should be used in the println() statement in order to display the desired information.



The 2 symbol has ASCII code value 253

These 4 pieces of text take up exactly 15 spaces each in the output window

This is in 2 tabs

This is in 1 tab

This is in 3 tabs

**(21 marks)**

**(b)**

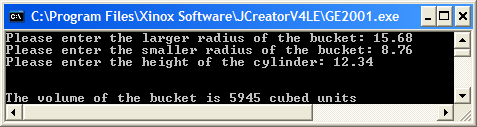
Write a Java program that determines the volume of a bucket.

The volume, V, of the bucket is given by the formula

where R is the larger radius at the top of the bucket and r is the smaller radius at the base of the bucket. h is the height of the bucket. π is the number pi whose value can be set to 3.142 for the purposes of this exam. You should **create pi as a constant** in your program and set it accordingly.

The program should request the user to supply values for both the larger and smaller radius and height of the bucket and then use the formula above to calculate the corresponding volume. The volume should then be displayed to the **nearest whole number**.

Your program should execute exactly as indicated in the sample run below:



**(12.33 marks)**

**Q2.**

Write the main() for a Java application which determines whether or not a person is eligible to join the Reserve Defence Forces (RDF).

In order to join the RDF in Ireland an applicant must satisfy the following conditions:

•be over 17 and under 35 years of age

•be a minimum of 1.55m in height if male or a minimum of 1.3m in height if female

•must pass a medical examination

•must not have a criminal record

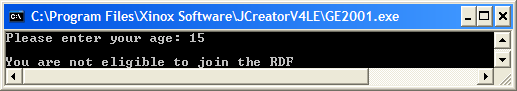
•must not be a serving member of the Gardai

Your program will ask the user to enter the details as indicated in the sample screenshots below and use the answers provided to determine whether the user is eligible to join the RDF.

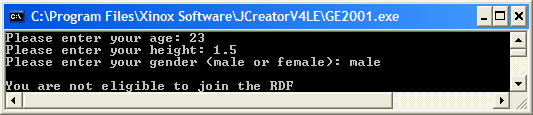
Note here that asking for further details completely depends on the user entering a “positive” reply for the current question. So, for example, the user will only get asked to enter their height and gender provided that the age criteria to become an RDF member have been satisfied, otherwise the program immediately terminates, issuing the “not eligible” message (as in **Run 1** below). As the gender and height are dependent on each other, both of these questions must be asked should the user enter a proper age value (as in **Run 2** below).

You should use **if/if-else** structures for any decision making your program requires.

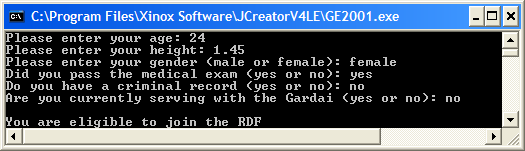
**Run 1**



**Run 2**



**Run 3**



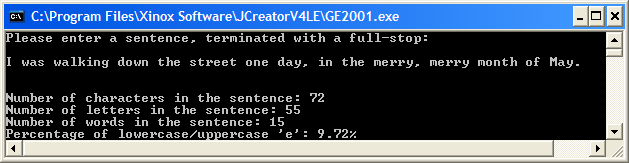
**(33.33 marks)**

**Q3.**

Write the main() for a Java program that uses a data-sentinel controlled **while** loop to read in a sentence from the user, terminated by a full-stop. The program should determine

* The number of characters in the sentence
* The number of letters in the sentence (uppercase or lowercase)
* The number of words in the sentence (we will take it that there is exactly one space character between each word here)
* The percentage of lowercase or uppercase ‘e’ in the sentence

Your program should run as indicated in the following sample screenshot, displaying the percentage to **2 decimal places**:



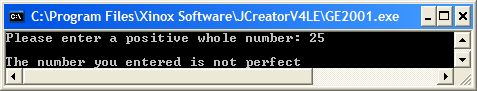
**(20.33 marks)**

**(b)**

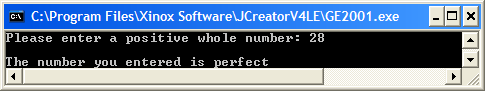
In mathematics, a *perfect* *number* is said to be a positive whole number whose divisors (excluding itself) add up to that number so, for example, 6 is a perfect number because its divisors are 1, 2 and 3 and they add up to 6, but 8 is not a perfect number because its divisors are 1, 2 and 4, which add up to 7.

Write the main() for a Java program that reads in a user-supplied positive whole number and determines whether or not it is a perfect number. You should use a task-controlled **while** loop for the iteration process here and your program should run as indicated in the following sample screenshots:

**Run 1**



**Run 2**



**(13 marks)**

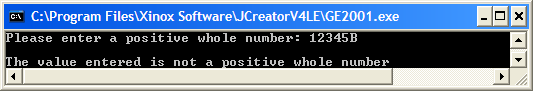
**Q4.**

Write the main() for a Java program that begins by asking the user to enter a positive whole number, which will be read in as a **String**. It then uses a counter-controlled **do-while** loop to process each character in the value entered to test whether it is a digit or not. If a non-digit is encountered, then the loop should immediately exit.

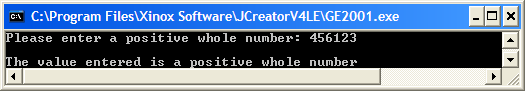
A test will then follow the loop to determine whether the value entered is indeed a positive whole number – it is as long as the number of characters processed by the loop before exiting equals the number of characters contained in the original value entered.

Your program should run as indicated in the sample screenshots below:

**Run 1**



**Run 2**



**(20.33 marks)**

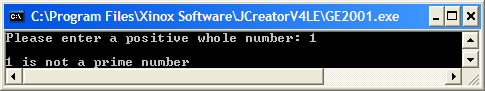
**(b)**

In mathematics, a *prime number* is any positive whole number greater than 1 that has no positive divisors other than itself and 1. So, for example, 2, 3 and 5 are prime numbers but 6 is not as it can be divided by 1, 2 , 3 and 6. Also, 1 is an exceptional case in that it is not a prime number.

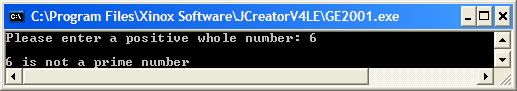
Write the main() of a Java program which asks the user to enter a positive whole number and uses a **for** loop to determine whether or not it is a prime number. Make sure your program takes the exceptional case of 1 into account also.

Your program should run as indicated in the sample screenshots below:

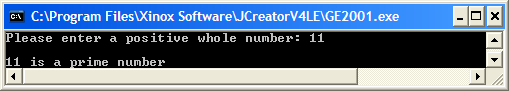
**Run 1**



**Run 2**



**Run 3**



**(13 marks)**