UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Engineering, Built Environment and IT
Department of Computer Science

Imperative Programming
COS 132

Semester Test 2 (ST2)

13 June 2020 (11:30 to 13:30)

Examiners

Internal: Dr Linda Marshall, Dr Patricia Lutu, Mr Emilio Singh, Ms Ntombi Banda, Mr Pula Rammoko, Ms Tari Mautsa and Mr Theo Naidu

Instructions

1. Read the question paper carefully and answer all the questions.
2. The assessment opportunity comprises of **7** questions on **7** pages.
3. **2** hours have been allocated for you to complete this paper. An additional 30 min have been allocated for you to download the paper and upload your answer document. The paper is divided into 2 sections, a practical section (Section A) and a theoretical section (Section B).
    - You should not spend more than 50 min on **Section A**
    - **Section B** is scheduled for 70 min of the 2 hours.
4. This paper is **take home** and is subject to the University of Pretoria Integrity statement provided below.
    - You are allowed to consult any literature.
    - You are not allowed to discuss the questions with anyone.
5. If you have any queries when writing the paper, post them in good time on the ClickUP Discussion Board under the appropriate thread, by replying to the thread, in the **Semester Test 2** Forum. Note, this forum is moderated and therefore your post will not be available for viewing to the COS132 class until is has been moderated.
6. An upload slot, per Activity, will be open on `assignments.cs.up.ac.za` for questions in **Section A**, that is from Question 1 to 3, from 11:30 to 13:30.
    - Note that file names are case sensitive. Failure to name the files as specified will result in no marks being awarded.
    - Each question (activity) is self contained and all of the code you will require to complete it will be provided on the appropriate question slot.
    - Each question will require you submit a separate archive to an individual upload slot. That is, each separate question will require its own answer archive upload.
    - Make sure your **name and student number is clearly visible** in the comments of the programs (.cpp files) you upload for Section A

7. Write your answers for Section B in a separate document (eg. using a word processor) and submit the document in **PDF format**. An upload slot will be open on the module ClickUP page under the **Semester Test 2 - Question paper and upload** menu option for questions from **Section B** for the duration of the examination (11:30 to 13:30) and then for an additional 30 min to give enough time to download this paper and upload your PDF. **No late submissions will be accepted.**
   - Please make sure your **name and student number is clearly visible** in the document you upload.
   - Additionally, provide an e-mail address and a phone number on your paper where you can be contacted, should there be any problems, in your Section B document.
8. **Marks per question**

Question:	1	2	3	4	5	6	7	Total
Marks:	5	10	15	15	10	12	20	87

Full marks is: 80

Integrity statement:
The University of Pretoria commits itself to produce academic work of integrity. I affirm that I am aware of and have read the Rules and Policies of the University, more specifically the Disciplinary Procedure and the Tests and Examinations Rules, which prohibit any unethical, dishonest or improper conduct during tests, assignments, examinations and/or any other forms of assessment. I am aware that no student or any other person may assist or attempt to assist another student, or obtain help, or attempt to obtain help from another student or any other person during tests, assessments, assignments, examinations and/or any other forms of assessment.

Section A - Practical

1. Activity 1 (5)

For this practical activity of the semester test, you are required to write a function that computes Euclidean distance. The function takes an array of inputs, of size 4, where the first two elements of that array, are the first x and y coordinates of the first point and similarly for the last two elements. The output of the function is a double value, representing the distance between the two points.

Your Euclidean function should be implemented a file called **distance.cpp**. In addition to containing your function, this file should contain the main function for your code. When run, the program should read from a file called **d1.txt**. The file will have just one line consisting of 4 double values separated by commas, such as

```
1.1,2.2,3.3,4.4
```

The first two values are for the first point, and the second two values are for the second point.

Once calculated, your code should just output the distance with a newline at the end.

Requirements
You will have a maximum of 5 uploads for this activity. You should include a blank **d1.txt**, **distance.cpp** and are encouraged to use the math, iostream, sstream, string and fstream libraries and to make use of functions. Remember to include your makefile.

2. Activity 2 (10)

For this practical activity of the semester test, you are required to write code, **map.cpp**, that reads in a file and stores it in a 2D array. The file is named **m1.txt** and consists of 10 rows. Each row comprises of 10 columns containing a single character. Your code needs to read this file into the 2D array and then print this information to the screen. Each row of output should have a newline at the end of the row. Note, there are no delimiters that separating the characters in each row.

Requirements
You will have a maximum of 5 uploads for this activity. You should include a blank **m1.txt**, **map.cpp** and are encouraged to use the math, iostream, sstream, string and fstream libraries and to make use of functions. Remember to include your makefile.

3. Activity 3 (15)

For this practical activity of the semester test, you are going to implement the delivery system as specified in the scenario document. You will create a file called **system.cpp**. This file will contain code that does the following:

1. Read in and store the map in **m1.txt** into a 2D array
2. Process each of the deliveries specified in **delivery.txt**.
   (a) For each delivery, calculate the profit potential of that delivery
   (b) Output each delivery's profit potential, and viability, with a newline at the end in the format specified in the scenario document

Requirements
You will have a maximum of 5 uploads for this activity. You should include a blank, **m1.txt** and **delivery.txt**, your **system.cpp** and are encouraged to use the math, iostream, sstream, string and fstream libraries and to make use of functions. Remember your makefile.

Section B - Theory

4. Short questions

(a) For each of the questions that follow, choose the most appropriate option. (8)

i. Which one of the following is a correct function header for a function that computes and returns the square root of a float value?
   A. void squareRoot(float value)
   B. float squareRoot(int value)
   C. int squareRoot(float value)
   D. float squareRoot(float value)

ii. Which of the following statements about arrays are true? (1) An array may contain elements of different data types. (2) A constant array must have an initialisation list. (3) When an array is passed as an argument to a function, the function has access to the original array.
   A. (1), (2) and (3) are true.
   B. (2) and (3) are true.
   C. (1) and (2) are true.
   D. Only (3) is true.

iii. Given the following array declaration:
const int SIZE = 5; int numbers[SIZE] = 10, 15, 20, 25, 30 ;
Which of the following statements will display all the values in the array?

A.
```
for (int index = 1; index <= SIZE; index++)
                    cout << numbers[index] << '\t';
```

B.
```
for (int index = 0; index < SIZE; index++)
                    cout << numbers[index] << '\t';
```

C.
```
for (int index = 1; index < SIZE; index++)
                    cout << numbers[index] << '\t';
```

iv. Which of the following is a correct declaration for a 2-dimensional array with 5 rows and 10 columns?
   A. int table[5][10];
   B. int table [10][5];
   C. int table[5], [10];
   D. int table[10, 5];

v. Given the following declarations:

```
float num1 = 10.5 , num2 = 5.2, temp;
```

Which of the following statements will correctly swap the values of num1 and num2;
   A.

```
num1 = num2;
num2 = num1;
```

B.

```
num2 = num1;
num1 = num2;
```

C.

```
temp  = num1;
num1 = num2;
num2 = temp;
```

D.

```
temp  = num2;
num2 = temp;
num1 = num2;
```

vi. Study the program statements below and answer the question that follows.

```
int num = 125;
int *numPtr;

numPtr = &num;
cout <<  numPtr  << '\t'  << *numPtr;
```

What will be displayed on the screen?

A. memory address of variable numPtr, followed by a tab, followed by 125
B. memory address of variable num, followed by a tab, followed by 125
C. value of num in hexadecimal followed by a tab followed by 125
D. memory address of variable numPtr followed by a tab followed by NULL

vii. Given the following statements:

```
const int SIZE = 5;
int values[SIZE] = { 10, 20, 30, 40, 50 };
for (index = 0; index < SIZE; index++)
    cout << values[index] << '\t';
```

Which of the following cout statements will produce the same output as the `cout` statement above?

A. `cout << *(values + index)  << '\t';`

B. `cout << &(values + index) << '\t';`

C. `cout << *values[index]  << '\t';`

D. `cout << &values[index]  << '\t';`

viii. Given the following statements:

```
const int SIZE = 10;
float *arrayPtr =  new float[SIZE];

for(int index = 0; index < SIZE; index++)
       *(arrayPtr + index) = index * 10.0;      //store value in array
```

Which of the following is equivalent to the statement with the comment "store value in array"?

A. `arrayPtr[index]  = index * 10.0;`

B.
```
&arrayPtr[index]    = index * 10.0;
```

C.
```
*arrayPtr[index]     = index * 10.0;
```

D.
```
*arrayPtr + index   = index * 10.0;
```

(b) What is the reason for undefined behaviour/errors that will occur when the following functions are called.

i. (2)
```
int* multiply(int n)
{
        int Value = n * 3;
          int *q = &Value;

       return q;
}          (2marks)
```

ii. (2)
```
void setToZero(int * const dptr)
{
          dptr = 0;
}
```

(c) Indicate if the following overloaded functions are valid or invalid. (3)

i.
```
int getMinimumValue(int, int);
int getMinimumValue(int, int, int);
```

ii.
```
int getMinimumValue(int, int);
int getMinimumValue(double, int);
```

iii.
```
int getMinimumValue(int, int);
void getMinimumValue(int&, int&);
```

5. Arrays

Use the code below to answer the questions that follow.

```
string street_map[10] = {"O","-","-","*","*","*"};
```

(a) Describe the contents of the `street_map` array. (1)

(b) Use a range-based for loop to initialise the last four (4) elements of the `street_map` array to the character '-'; (2)

(c) i. Write a statement that deallocates dynamically created memory in the program below. (1)

```
int* allocate(int nSize)
{
          int *ptr = new int[nSize];

          return ptr;
}


int main()
{
    int *array = allocate(25);

// do stuff with array
```

```
    //
    //

// deallocate memory here !!!

    return 0;
}
```

ii. The 10-by-10 maps very soon become too limiting for the delivery service company. They decide to include two integers at the beginning of each map file to indicate the number of rows and columns used to define the map in the map file. The typical 10-by-10 map file (`d1.txt`) from the scenario will change to the following: (6)

```
10 10
#O########
#**------#
#**--**--#
#----**--#
O--------#
#***--**-#
#***--**-O
#***--**-#
#--------#
########O#
```

Write the main program that reads the map into an appropriately sized, `rows`-by-`columns` array and then frees the memory assigned to the array.

6. Functions

(a) The delivery company needs to consider the cost of petrol used in delivering packages. The value of the petrol used is determined by the price of petrol (per litre), delivery car fuel consumption (in litres/100km), and travel distance (in km). The company has taken a great liking to a particular car brand. Therefore, the great majority of its fleet consists of that car brand which has a fuel consumption of 8.2 litres/100km.

i. Write the prototype for the function `computePetrolCost` that estimates and returns the value of the petrol used in delivering a package. The function prototype should appropriately reflect any default values. (2)

ii. Implement the `computePetrolCost` function such that it returns the value of the petrol used in a delivery. The petrol cost is the product of the three input parameters. Make sure to apply any necessary value conversions. (2)

(b) Suppose the delivery system has a function that sends an email to a driver to instruct them to fulfil a delivery order, as defined below. (6)

```
void sendOrderToDriver(string order, string driverEmail)
{
        sendEmail(driverEmail, orderDetails);
}
```

Modify the `sendOrderToDriver` function such that it keeps count of the number of order instructions sent. Add a parameter called `shouldReset` that makes it possible to reset the count to 0 (for example, at the start of a new day). The function should return the count.

(c) Overload the Euclidean distance function you implemented in Question 1 to accept four integer values as input. The first two integer parameters should correspond to the x and y coordinates of the first point, and the last two integer values should correspond to the x and y coordinates of the second point, respectively. (2)

The formula for the Euclidean distance (Equation (3) in the Scenario and repeated below) is:

$$d(p, q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}$$

7. Searching and Sorting

(a) Given a two-dimensional delivery map from the Semester Test scenario, which is the most appropriate search algorithm to use for locating the health check points (O) in each row? Give reasons for your answer. (2)

(b) Suppose that after running your delivery program, you end up with a list of possible profit margins corresponding to house deliveries that need to be made. (2)

What is the most efficient way of sorting the profit margins list such that the viable deliveries are listed first? Which sorting algorithm would you use? Motivate your choice of sorting algorithm.

(c) Consider the following array definition which shows an array of profit margins for some of the different deliveries.

```
double profitmargins[]={0.4, 2.5, 5.0, 10.0, 15.0, 20.5}
```

i. Write a function `getLargest(double numbers[], int size)`, which accepts the array `profitmargins` and `size` as its arguments. The function `getLargest` should return the largest number stored in the array. (5)

ii. In the function `getLargest`, the contents of the array can be easily modified. Write a function header for the function `getLargest` such that the contents of the array passed as an argument cannot be modified. (1)

iii. Suppose the delivery company wants to order the profit margins for the different delivery destinations starting with deliveries with the lowest profit margin. The incomplete code below implements an efficient bubble sort algorithm that can be used to sort an array of profit margins in ascending order where `size` is a parameter that holds the size of the array that can be used to store the profit margins and the parameter `arr` holds the address of the array passed as an argument. The missing expressions and/or instructions are numbered [*i*] to [*iv*]. (4)

In a bubble sort, after the first pass, the largest number is guaranteed, to be the highest-indexed element of the array; after the second pass, the two highest numbers are "in place", and so on. Instead of making five comparisons on every pass (as will be the case with the array `profitmargins[]`), the bubble sort implemented here should make four comparisons on the second pass, three on the third pass, and so on.

```
void bubbleSort(double arr[], int size)
{
        for (int k = 0;     --[i]--;      k++)
        {
                for (int j = 0;   --[ii]-- ;  --[iii]--)
                {
                        if   --[iv]--
                        {
                                double temp = arr[j + 1];
                                arr[j + 1] = arr[j];
                                arr[j] = temp;
                        }
                }
        }
}
```

Write **ONLY** the missing code that should be placed where each of the numbers appear in the above listing

(d) Write a function `findcharacter(const char array[][COLS], int n, char item)` that locates a given character, for example a health check point, in the map. The function should search only the n^{th} row (e.g. for the second row, a value of 2 will be passed) of the 2D array that is stored in the delivery map. The function locates the character (`item`) the delivery system wishes to find and if the searched character appears in the n^{th} row the function should return the column index of where the character was found else return `-1` if searched item is not found. (6)