UTM UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF SOCIAL SCIENCES AND HUMANITIES

SECJ1013 - 05 (PROGRAMMING TECHNIQUE I)

SEMESTER 1, 2020/2021

ASSIGNMENT 4

GROUP MEMBER:

NAME	MATRIC NO.
1. Lee Ming Qi	A20EC0064
2. Ong HanWah	A20EC0129
3. Vico King	A20EC0338

TEACHER-IN-CHARGE

Dr. GOH EG SU

DATE SUBMISSION

2/2/2021

PART A [70 MARKS]

QUESTION 1 [4 Marks]

Program 1 is the program that is used by the parking attendant to calculate the fee for parking based on the hours and rates. The parking building consists of employee and non-employee car with different type of rates:

(a) Employee of the company: RM 1.50 per hour

(b) Non-employee: RM 3.00 per hour

Based on the comments given in **Program 1**, fill in the blank lines with appropriate C++ statements. The program should generate the output as shown in **Figure 1**. *Note:* The **bold texts** in **Figure 1** indicate input from the user. (4 marks)

```
The number of cars: 2

Hours parked for car 1: 3
Hourly pay rate for car 1: 1.50

Hours parked for car 2: 5
Hourly pay rate for car 2: 3.00

Here is the total fee for each car:
Car [1] = RM4.50
Car [2] = RM15.00
```

Figure 1: Example runs of the Program 1

```
1   // Program 1
2   #include<iostream>
3   #include<iomanip>
4   using namespace std;
5
6   int main()
7   {
8     const int no_cars = 2; //Constant for the array size
9
10    //Declare an array named hours and rate to allocate 2
11    //elements of type integer and double respectively.
12    //Use constant declared in line 8.
```

```
   int hours[no_cars];
   double rate[no_cars];

  cout << "The number of cars: " << no_cars;
  cout << endl;

  for (int index = 0; index < no_cars; index++)
  {
    cout << "\nHours parked for car " << (index+1) << ": ";
    //Read a value and assign it into array named hours
    cin >> hours[index];
    cout << "Hourly pay rate for car " << (index+1) << ": ";
    //Read a value and assign it into array named rate
    cin >> rate[index];
  }
  cout << "\nHere is the total fee for each car:\n";
  cout << fixed << showpoint << setprecision(2);

  for (int index = 0; index < no_cars; index++)
  {
     //Calculate a value of totalPay (fee for parking)
     double totalPay = hours[index] * rate[index];
     cout << "Car [" << (index + 1);
     cout << "] = RM" << totalPay << endl;
  }
   return 0;
}
```

QUESTION 2 **[16 Marks]**

Given the declaration and initialization of some parallel arrays as in the following code segment:

```
string patients[] = {"Wendy", "Kumar", "Ros", "Mael"};
int age[] = {25, 43, 32, 54};
double averageSugarLevel[4];
```

Based on the concept of arrays and using loops, answer the following questions:

(a) Define a two-dimensional array named **`sugarLevel`** which has large enough space to hold the data in the following table. Initialize the array with values from the table. (2 marks)

4.5	5.7	6.8
6.2	7.1	7.3
5.1	6.3	8.0
7.5	8.5	7.3

Answer:

```
double sugarLevel[4][3] = {{4.5, 5.7, 6.8},
                           {6.2, 7.1, 7.3},
                           {5.1, 6.3, 8.0},
                           {7.5, 8.5, 7.3}};
```

(b) Write **a code segment** that prints the patient names along with their ages. The output should be as follows: (3 marks)

```
Age of Patients
---------------
Wendy     25
Kumar     43
Ros       32
Mael      54
```

Answer:

```
cout << "Age of Patients\n";
cout << "-----------------\n";

for (int i=0; i<4; i++){
      cout << setw(10) << left << patients[i];
      cout << age[i] << endl;
}
```

(c) Write **a code segment** that produces the following output from the array named **`sugarLevel`** that you defined in (a). (4 marks)

```
Name            Breakfast  Lunch  Dinner
Wendy              4.5      5.7     6.8
Kumar              6.2      7.1     7.3
Rose               5.1      6.3     8.0
Mael               7.5      8.5     7.3
```

Answer:

```
cout << left << setw(13) << "Name" << setw(11) << "Breakfast"
     << setw(9) << "Lunch" << "Dinner" << endl;
for(int i=0; i<4; i++){
     cout << setw(16) << patients[i];
     for(int j=0; j<3; j++){
          cout << fixed << setprecision(1) << showpoint;
          cout << setw(9) << sugarLevel[i][j];
     }
     cout << endl;
}
```

(d) Write a **code segment** to calculate the average sugar level of the patient. The code also store the average sugar level value in the array **`averageSugarLevel`** and print the list of the patients' average sugar level value as follows. (7 marks)

```
Sugar Level of Patients
-----------------------
Wendy     25
Kumar     43
Ros       32
Mael      54
```

Answer:

```
for (int row=0; row<4; row++){
     double sum = 0;
     for (int col=0; col<3; col++){
          sum += sugarLevel[row][col];
     }
     averageSugarLevel[row] = sum / 3;
}

cout << "Sugar Level of Patients\n";
cout << "-----------------------\n";
cout << fixed << setprecision(2) << showpoint;

for (int i=0; i<4; i++){
     cout << setw(10) << left << patients[i];
     cout << averageSugarLevel[i] << endl;
}
```

QUESTION 3 **[11 Marks]**

(a) Name at least two member functions associated with **`iostream`** object. (2 marks)

Answer:

```
cin.getline, cin.ignore
```

(b) Suppose two file stream objects named **`fin`** and **`fout`** have been declared in the code as follows:

```
ifstream fin;
ofstream fout;
```

Complete the code so that the program can copy 10 numbers from a file named **`file1.dat`** and save the numbers to another file named **`file2.dat`** (copy 10 numbers between files). The program should also include an error checking for opening the files. (9 marks)

Answer:

```
fin.open("file1.dat");     //open file1.dat
fout.open("file2.dat");    //open file2.dat
int num;                   //variable to store the numbers
//error checking for opening file
if(fin){
     for(int i=0; i<10; i++){
          fin >> num;           //read number from file1.dat
          fout << num <<endl;   //write number to file2.dat
     }
     fin.close();    //close file1.dat
     fout.close();   //close file2.dat
}
else{
     //Display error message
     cout << "Error opening file1.dat\n";
}
```

QUESTION 4 **[4 Marks]**

Based on the **Program 2**, write the output into **Table 1**. *Note:* one cell represents one space per digit or character. (4 marks)

```
// Program 2
//This program write three rows of numbers to a file
#include <iostream>
#include <fstream>
#include <iomanip>
using namespace std;

int main ()
{
    const int ROWS = 3;
    const int COLS = 3;
    double nums [ROWS][COLS] = {1501, 3.579, 984,
                                5.3, 91, 1979,
                                24, 606, 11.0287};
    fstream outFile ("Jadual.txt”, ios::out);

    for (int row = 0; row < ROWS; row++)
    {
        for (int col = 0; col < COLS; col++)
        {
            outFile << setprecision (1) << fixed << showpoint;
            outFile << setw(6) << nums [row][col];
        }
        outFile << endl;
    }
    outFile.close();
    return 0;
}
```

Table 1: Output file named **`Jadual.txt`**

1	5	0	1	.	0				3	.	6		9	8	4	.	0
			5	.	3			9	1	.	0	1	9	7	9	.	0
		2	4	.	0		6	0	6	.	0			1	1	.	0

QUESTION 5 **[4 Marks]**

As you are travelling to the United States, it is important to know the temperature for you to be able to prepare what to wear for the day. Apparently in the United States, they use Fahrenheit and not Celsius. A conversion program that will ask for a temperature in Fahrenheit, convert that value to Celsius, and display the result, would definitely become handy. Complete the **Program 3**. You need to write the function **`convert`**. *Note:* $C = 5 / 9 \times (F - 32)$. Do not modify the function **`main`**. (4 marks)

```
// Program 3
#include <iostream>
#include <iomanip>
using namespace std;

//Write your function prototype here
void convert (double&);
int main ()
{
    double temperature;

    cout << "Enter a temperature in Fahrenheit, and I will
         << convert it to Celsius: ";
    cin >> temperature;
    convert (&temperature);
    cout << fixed << setprecision(4);
    cout << "Value in Celsius: " << temperature << endl;

    return 0;
}

//Write the function convert here
void convert (double& temperature)
{
   temperature = 5.0/9*(temperature-32);
}

```

QUESTION 6 **[10 Marks]**

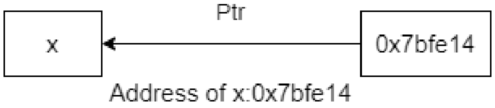
(a) Determine the output and draw a memory layout (or memory allocation) of the pointers and variables for code segment below: (7 marks)

Note: Draw a memory layout that represents C++ statement line by line.

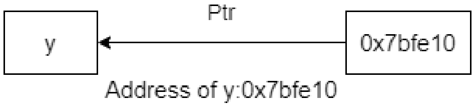
```
int  x = 10, y = 20, z = 30;
int *ptr;

cout << x << “ “  << y << “ “ << z << endl;
ptr = &x;
*ptr *= 10;
ptr = &y;
*ptr *= 4;
ptr = &z;
*ptr *= 2;

cout << x << “ “  << y << “ “ << z << endl;
```

Answer:

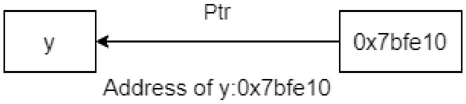
```
int  x = 10, y = 20, z = 30; int *ptr;

cout << x << " "  << y << " " << z << endl;
ptr = &x;
*ptr *= 10;
```



```
ptr = &y;
*ptr *= 4;
```



```
ptr = &z;
*ptr *= 2;
```



```
cout << x << " "  << y << " " << z << endl;
```

Output

```
10 20 30
100 80 60
```

(b) Determine the output for code segment below: (3 marks)

```
int numbers[5] = {2,4,6,8,10};
int *numPtr;

numPtr = numbers;
cout << *(numbers + 3) << endl;
cout << *(numbers) << endl;

for (int count=0; count < 2; count++)
{
    cout << *numPtr  << “\t“;
    numPtr++;
}
cout << endl;

for (int count=0; count < 2; count++)
{
    numPtr--;
    cout << *numPtr << “\t“;
}
```

Answer:

```
8
2
2       4
4       2
```

QUESTION 7 **[5 Marks]**

The code segment below has five (5) syntax and/ or logical errors. Identify and describe each of the errors. (5 marks)

```
struct TwoLoc
{
    int x = 10;
    int y = 15;
}

int main()
{
    TwoLoc point[10];
    point.x[0] = 1;
    point.y[0] = 1;
    cout << point << endl;
    return 0;
}
```

Answer:

```
struct TwoLoc
{
  int x = 10;
  int y = 15;
};//semicolon

int main()
{
  TwoLoc point[10];
  point[0].x = 1; //wrong class type
  point[0].y = 1; //wrong class type

  cout << point[0].x << point[0].y << endl;
  //must put specific expression

  return 0;
}
```

QUESTION 8 **[16 Marks]**

Meteorological Department is responsible to keep the information about weather forecast in Malaysia and display the information through the website. Write the code segment for each of the following tasks:

(a) i) Declare a structure type named **`TempScale`**, with the following members:

`fahrenheit` : a double value

`celsius` : a double value

ii) Declare a structure type named **`Reading`**, with the following members:

`windSpeed` : an integer value

`humidity` : a double value

`temperature`: a **`TempScale`** structure variable

iii) Declare a variable of structure type **`Reading`** named **`today`**. (6 marks)

Answer:

```
i)
   struct TempScale{
     double fahrenheit;
     double celcius;
   };


ii)
   struct Reading{
     int windSpeed;
     double humidity;
     TempScale temperature;
   };


iii)

   Reading today;
```

(b) Write statements that will store the following data into the variable you declared in (a).

Wind speed: 8 km/h

Humidity: 53%

Fahrenheit temperature: 80.2 degrees

Celsius temperature: 26.8 degrees

(3 marks)

Answer:

```
//storing windspeed value
today.windSpeed = 8;

//storing humidity value
today.humidity = 53;

//storing fahrenheit value from temperature
today.temperature.fahrenheit = 80.2;

//storing celcius value from temperature
today.temperature.celcius = 26.8;
```

(c) Define a function named **showReading**. It should accept a **Reading** structure variable as its argument. The function should display the contents of the variable onto the screen. (7 marks)

Answer:

```
void showReading(Reading today)
{
  //print windspeed value
  cout << "Wind speed: ";
  cout << today.windSpeed << " km/h" << endl;

  //print humidity value
  cout << "Humidity: ";
  cout << today.humidity << " %" << endl;

  //print fahrenheit value from temperature
  cout << "Fahrenheit temperature: ";
  cout << today.temperature.fahrenheit << " degrees" << endl;

  //print celcius value from temperature
  cout << "Celcius temperature: ";
  cout << today.temperature.celcius << " degrees" << endl;
}
```

PART B [30 MARKS]

Ministry of Transport Malaysia is required to prepare a report of the total road accidents by states in Malaysia from the year, 2006 to 2015.

INSTRUCTIONS:

Write a C++ program to calculate the average number of road accidents for each state and find the highest number of road accidents from 2006 to 2015. Your program should be able to do the following tasks:

(a) The program should use the following **`struct`** definition:

```
struct dataAcc
{
   int numAcc[10]; //number of road accidents from 2006-2015
   string state; //states in Malaysia
   float avg; //average number of road accidents for 2006-2015
}
```

(b) The program should use an array of **`struct`** defined in (a) to store the total road accidents by states in Malaysia from 2006 to 2015.

(c) The program will read input data from an input file named “input.txt” into the array of **`struct`** declared in (b). Example of input data in “input.txt” is shown in **Figure 2**.

(d) The program should have two (2) global constants and one (1) global variable as follows:

Global contants:

NUM_STATE = 14 ⇨ Number of states

NUM_YEAR = 10 ⇨ Number of years

Global variable:

out ⇨ Variable to point to output file named “output.txt”

(e) Besides the function **`main()`**, the program has three (3) other functions as described in **Table 2**. One of the functions is given below:

```
void displayLine()
{
   for (int i = 0; i < 98; i++)
```

```
        out << "-";
    out << endl;
}
```

Based on the description given in **Table 2**, the program needs to define two (2) more functions. You should use appropriate argument(s) (if necessary) for each function.

Table 2: Description of functions

Function	Description
`displayLine()`	To display lines using 98 characters of '-' in the output file using a loop.
`cal_Avg()`	To calculate the average number of road accidents for each state. The function should accept a 1D array of the number of road accidents for 10 years for each state as its argument. The function should return the average number of road accidents for each state.
`find_HighLow()`	To find and display the highest number of road accidents from 2006 to 2015. The function should accept a 1D array of structures as its argument.

(f) The program needs to display the following information. **Figure 3** shows the output of the successful program.

- Name of state.
- Number of road accidents for 2006-2015 in each state.
- Average number of road accidents for 10 years in each state.
- The highest number of road accidents for 10 years with the name of state and year.

The assessment criteria are shown in **Table 3**.

```
1160 1364 1417 1633 1548 1791 1881 1895 1888 1861 PERLIS
15505 16172 16520 17701 17966 19699 19935 20228 20159 22016 KEDAH
32573 33881 34049 33719 34306 37158 37851 39361 38747 39856 PULAU PINANG
27432 29203 30539 32327 32072 33506 34714 35408 35131 36736 PERAK
92632 99157 100380 107429 115565 128876 129106 135024 137809 140957 SELANGOR
46254 49454 48671 51942 53493 58795 61872 64527 63535 64664 KUALA LUMPUR
15197 16079 17362 18369 19407 21157 22146 23066 23748 22939 NEGERI SEMBILAN
10707 11720 12105 13275 14110 14720 15195 16083 16375 17069 MELAKA
43757 46584 48667 51747 55381 59501 62316 64600 64473 67112 JOHOR
13242 13982 15629 17068 17315 19001 20554 20130 19071 19635 PAHANG
7337 8116 8842 9549 9707 9603 9968 9748 10326 9960 KELANTAN
7098 8155 8814 10118 10106 10684 10861 10996 9383 10381 TERENGGANU
13550 14256 14588 15798 16192 16585 17446 17438 17858 17290 SABAH
14808 15196 15488 16655 17253 17964 18578 18700 17693 19130 SARAWAK
```

Figure 2: Input file named "**`input.txt`**"

```
------------------------------------------------------------------------------------------------------
     STATE          2006   2007   2008   2009   2010   2011   2012   2013   2014   2015    AVERAGE
------------------------------------------------------------------------------------------------------
 PERLIS             1160   1364   1417   1633   1548   1791   1881   1895   1888   1861     1643.8
 KEDAH             15505  16172  16520  17701  17966  19699  19935  20228  20159  22016    18590.1
 PULAU PINANG      32573  33881  34049  33719  34306  37158  37851  39361  38747  39856    36150.1
 PERAK             27432  29203  30539  32327  32072  33506  34714  35408  35131  36736    32706.8
 SELANGOR          92632  99157 100380 107429 115565 128876 129106 135024 137809 140957   118693.5
 KUALA LUMPUR      46254  49454  48671  51942  53493  58795  61872  64527  63535  64664    56320.7
 NEGERI SEMBILAN   15197  16079  17362  18369  19407  21157  22146  23066  23748  22939    19947.0
 MELAKA            10707  11720  12105  13275  14110  14720  15195  16083  16375  17069    14135.9
 JOHOR             43757  46584  48667  51747  55381  59501  62316  64600  64473  67112    56413.8
 PAHANG            13242  13982  15629  17068  17315  19001  20554  20130  19071  19635    17562.7
 KELANTAN           7337   8116   8842   9549   9707   9603   9968   9748  10326   9960     9315.6
 TERENGGANU         7098   8155   8814  10118  10106  10684  10861  10996   9383  10381     9659.6
 SABAH             13550  14256  14588  15798  16192  16585  17446  17438  17858  17290    16100.1
 SARAWAK           14808  15196  15488  16655  17253  17964  18578  18700  17693  19130    17146.5
------------------------------------------------------------------------------------------------------
The highest number of road accidents = 140957 at SELANGOR on 2015
------------------------------------------------------------------------------------------------------
```

Figure 3: Example output in output file named "**`output.txt`**"

Table 3: Assessment criteria

Item	Criteria	Marks
A	Using an appropriate structure for the program (*e.g*. all required header files are included, the function main is properly written, *etc*.).	3
B	Define a global constants, global variable and array of structures	2
	Reading the input data from input file.	4
C	Definition of function **`cal_Avg()`**.	4
	Definition of function **`find_HighLow()`**.	7
D	Printing the output as requested in (f), Figure 3 and in a proper format.	10
Total		**30**

Answer spaces for Part B

```
#include <iostream>
#include <fstream>
#include <iomanip>
using namespace std;

struct dataAcc{     //declaration of dataAcc
  int numAcc[10];
  string state;
  float avg;
};

void displayLine();  //function prototype
float calc_Avg(int*);
void find_HighLow(dataAcc*);

const int NUM_STATE = 14; //global constant variable declaration
const int NUM_YEAR = 10;
ofstream out;

int main(){

  dataAcc data[NUM_STATE]; //declaration of array of struct dataAcc
  int num;
  string state;
  ifstream in;
  in.open ("input.txt");  //open input.txt file

  for (int i = 0; i < NUM_STATE; i++){
    for (int j = 0; j < NUM_YEAR; j++){
      in >> num;                //read the data from input.txt
      data[i].numAcc[j] = num; //input the data into array of struct
    }
    getline(in, state);
    data[i].state = state;
    data[i].avg = calc_Avg(data[i].numAcc);
  }
  in.close();  //close the file

  out.open ("output.txt"); //open output.txt file

  displayLine();
  out << right << setw(10)  << "STATE" << setw(7) << "";

  for (int y = 2006; y < 2016; y++){
    out << right << setw(7) << y ;
  }

  out << right << setw(10) << "AVERAGE" << endl;
  displayLine();
  //move the data from array of struct into output.txt
```

<u>Answer spaces for Part B</u>

```
  for (int x = 0; x < NUM_STATE; x++){
    out << left << setw(17) << data[x].state;
    //store state name into output.txt
    for (int z = 0; z < NUM_YEAR; z++){
      out << right << setw(7) << data[x].numAcc[z];
    } //store data into output.txt
    out << fixed << setprecision(1);
    out << right << setw(10) << data[x].avg << endl;
  } //store average into output.txt

  displayLine();      //call function
  find_HighLow(data);
  displayLine();

  out.close();   //close the file

  return 0;
}


void displayLine(){  //function definition
  for (int i = 0; i < 98; i++)
    out << "-";
  out << endl;
}

float calc_Avg(int* num){ //function definition
  int total = 0;

  for (int i = 0; i < NUM_YEAR; i++)
    total += num[i]; //calculate total

  return (static_cast<float>(total)/NUM_YEAR); //calculate the average
}

void find_HighLow(dataAcc* d){ //function definition

  int max , max_year;
  string max_state;

  for (int i = 0; i < NUM_STATE; i++){
    for (int j = 0; j < NUM_YEAR; j++){
      if(d[i].numAcc[j] > max){
        max = d[i].numAcc[j];   //the highest number
        max_state = d[i].state; //the highest number state
        max_year = 2006 + j;    //the highest number year
      }
    }
  }
  out << "The highest number of road accidents = ";

  out << max << " at" << max_state << " on "<< max_year << endl;
  //store the highest number, state and year into output.txt
}
```