CONFIDENTIAL

UTM UNIVERSITI TEKNOLOGI MALAYSIA SCHOOL OF COMPUTING Faculty of Engineering

FINAL EXAMINATION
SEMESTER I, SESSION 2020/2021

PART I: DEBUGGING

COURSE CODE	**: SECJ/SCSJ 1013**
COURSE NAME	**: PROGRAMMING TECHNIQUE I**
SECTION	**: 1 – 10 & 15**
DATE/ DAY	**: 9 FEBRUARY 2021 (TUESDAY)**
START	**: 10.10 AM**

INSTRUCTIONS TO THE STUDENTS:

- Read the problem and instructions carefully.
- You are given **FIFTY MINUTES** to complete the test inclusive of the submission of your program (**40 minutes to answer** the question and **10 minutes to submit** the answer).
- Write your particular (**Name, Matrics_No, Section and Lecturer Name**) in your program as a comment.
- Your program must follow the input and output as required in the text and shown in the examples. You must test the programs with (but not limited to) all the input given in the examples.
- A candidate who is suspected of cheating in examinations is liable to disciplinary action including (but not limited to) suspension or expulsion from the University. All materials and or devices which are found in violation of any examination rules and regulation will be confiscated.

IMPORTANT NOTES:

- All the **COMMENT STATEMENTS** in the submitted program **WILL NOT BE EVALUATED**.

SUBMISSION PROCEDURE:

- Only the source code is required for the submission and the source code's file shall be named as follows: ***Name_ matricsNo_section.cpp*** (i.e. *AinaAli_A20EC018_01.cpp*).
- You do not need to compress the file.
- Submit the source code file via the **UTM's e-learning system**.

Question [35 Marks]

You are given a C++ program (**`FinalQ1.cpp`**) with errors (syntax errors and/ or logical errors, if any). The program has two (2) user-defined functions as listed in Table 1.

Table 1: Description of functions

Function Name	**Description**
`getSpec`	The function accepts two arguments. The first argument is the caption text to guide users to enter a correct type of car's attributes. The second argument is the car's attribute itself which was represented by a parallel array of pointers type for variables: **`models`**, **`horsepower`**, and **`weight`**. The **`getSpec`** function assigns data entered by the users to the corresponding item of these array variables with their index is based on variable used to control the loop.
`fastestModel`	The function accepts three arguments that are the **`models`**, **`horsepower`**, and **`weight`** parallel array pointer variables. It calculates the weight/ horsepower ratio of each car's model and then returns the index number of array item which representing the fastest car (car's model which has the lowest weight/ horsepower ratio).

The **`main`** function of the program has a series of calls to **`getSpec`** function inside a loop control structure. The **`models`**, **`horsepower`**, and **`weight`** are parallel pointer type arrays with references to their item's index was made based on variable used to control the loop (loop which used to make a series of call to **`getSpec`** function). Some of the output was produced by a call made to **`fastestModel`** function. The last part of the output produced after the index number of parallel array items representing the fastest car was returned by the **`fastestModel`** function.

You are required to debug the errors, compile, and run the program. You are **NOT ALLOWED** to **remove** any statements in the program. You are only allowed to **update** the statements provided in the program and add a new statement(s) if absolutely necessary.

The program should produce the output as in **Figure 1**. ***Note:*** The values in **bold** are input by the user.

```cpp
//FinalQ1.cpp
#include <iostream>
#include <string>
#define SIZE 3

using namespace std;

void getSpec(string caption, string *p_data) {
    cout << caption;
    cin >> p_data;
}

void getSpec(string caption, int p_data) {
    cout << caption;
    cin >> *p_data;
}

int fastestModel(string *m[], int *hp, int *w) {
    float whp_ratio, whp_ratio_lowest = 0;
    int idx;

    cout << "Check weight and horsepower ratio: \n";

    for (int i = 0; i < SIZE; i++) {
        whp_ratio = (float)*w[i] / *hp[i];
        cout << m[i] << " - " << whp_ratio << "\n";

        if (whp_ratio_lowest > whp_ratio) {
            whp_ratio = whp_ratio_lowest;
            idx = i;
        }
    }

    cout << "\n";

    return idx;
}

// Start main function
int main() {
    string *models [SIZE];
    int *horsepower [SIZE];
    int weight [SIZE];

    // Examples of car's models, horse power and weight
    // Almera - 76 hp - 1035 Kg
    // City - 88 hp - 1110 Kg
    // Persona - 80 hp - 1210 Kg
    // Vios - 78 hp - 1112 Kg
    for (int i = 0; i < SIZE; i++) {
        models[i] = new string;
        getSpec("Car's model: ", models[i]);

        horsepower[i] = new string;
        getSpec("Horsepower (hp): ", horsepower[i]);

        getSpec("Weight (Kg): ", weight[i]);

        cout << "\n";
    }

    int idx_fastest = fastestModel(models, horsepower, *weight);

    cout << "Fastest car's model is " << *models[idx_fastest]
         << " with " << *horsepower[idx_fastest] << " horsepower and "
         << *weight[idx_fastest] << " kg of weight\n\n";

    // delete array data from memory
    for (int i = 0; i < SIZE; i++) {
        delete models[i];
        delete horsepower[i];
        delete weight[i];
    }

    return 0;
}
```

```
Car's model: Almera
Horsepower (hp): 76
Weight (Kg): 1035

Car's model: City
Horsepower (hp): 88
Weight (Kg): 1110

Car's model: Vios
Horsepower (hp): 78
Weight (Kg): 1112

Check weight and horsepower ratio:
Almera - 13.6184
City - 12.6136
Vios - 14.2564

Fastest car's model is City with 88 horsepower and 1110 kg of weight
```

Figure 1: Expected output