

**FACULTY OF INFORMATION TECHNOLOGY****Programming 621****1st SEMESTER ASSIGNMENT****Name & Surname:** Tawana Kombora**ICAS/ITS No:** 402202621**Qualification:** BSc I.T**Semester:** 1st**Module Name:** Programming 621**Submission Date:** 31st May 2023

ASSESSMENT CRITERIA	MARK ALLOCATION	EXAMINER MARKS	MODERATOR MARKS
MARKS FOR CONTENT			
QUESTION ONE	30		
QUESTION TWO	30		
QUESTION THREE	30		
TOTAL MARKS	90		
MARKS FOR TECHNICAL ASPECTS			
1. TABLE OF CONTENTS Accurate numbering according to the numbering in text and page numbers.	2		
2. LAYOUT AND SPELLING Font – Calibri 12 Line Spacing – 1.0 Margin should be justified.	3		
3. REFERENCE According to the Harvard Method	5		
TOTAL MARKS	10		
TOTAL MARKS FOR ASSIGNMENT	100		
Examiner's Comments:			
Moderator's Comments:			
Signature of Examiner:		Signature of Moderator:	

Table of Contents

QUESTION ONE.....3

QUESTION TWO.....6

QUESTION THREE.....11

QUESTION ONE

```
1  #include <iostream>
2  #include <set>
3
4  /*
5   * These statements make the specified class available to this program
6   */
7  using std::cin;
8  using std::cout;
9  using std::endl;
10 using std::set;
11
12 int checkConditions(int array[][4], int rowSize, int colSize, set<int> conditions);
13
14 int main(void)
15 {
16     int row, col;
17     set<int> conditions = {1, 2, 3, 6, 7, 8};
18     int arr[3][4] = {{1, 2, 3, 4},
19                     {3, 4, 5, 6},
20                     {5, 6, 7, 8}};
21     cout << checkConditions(arr, 3, 4, conditions) << endl;
22     return 0;
23 }
24
25 /*
26 * The function takes:
27 * - an array of integers (array),
28 * - row size (rowSize),
29 * - column size (colSize),
30 * - a set of integers (conditions)
31 * It iterates over each element of the array using nested loops.
32 * For each element, it assigns the current number to the variable currentNumber.
33 * It then checks if currentNumber is present in the set by iterating through the conditions set using a ranged-based for loop.
34 * If a match is found, it sets the presentInSet flag to true and breaks the loop.
35 * After the inner loop completes, it checks the value of presentInSet.
36 * If it is false, it means the number was not found in the set, so the function returns -1.
37 * If all numbers pass the condition check, the function returns 0, indicating that all numbers in the array are contained in the set.
38 */
39
40 int checkConditions(int array[][4], int rowSize, int colSize, set<int> conditions)
41 {
42     for (int row = 0; row < rowSize; row++)
43     {
44         for (int col = 0; col < colSize; col++)
45         {
46             int currentNumber = array[row][col];
47             bool presentInSet = false;
48
49             for (int num : conditions)
50             { // range-based for loop. for each number in the set conditions do some action
51                 if (num == currentNumber)
52                 {
53                     presentInSet = true;
54                     break;
55                 }
56             }
57
58             if (!presentInSet)
59             {
60                 return -1; // Number not found in the set, return -1
61             }
62         }
63     }
64
65     return 0; // All numbers in the array are contained in the set
66 }
67
```

#include <iostream>

#include <set>

/*

* These statements make the specified class available to this program

*/

```
using std::cin;
using std::cout;
using std::endl;
using std::set;
```

```
int checkConditions(int array[][4], int rowSize, int colSize, set<int> conditions);
```

```
int main(void)
{
    int row, col;
    set<int> conditions = {1, 2, 3, 4, 5, 6, 7, 8};
    int arr[3][4] = {{1, 2, 3, 4},
                     {3, 4, 5, 6},
                     {5, 6, 7, 8}};
    cout << checkConditions(arr, 3, 4, conditions) << endl;
    return 0;
}
```

```
/*
```

* The function takes:

- an array of integers (array),
- row size (rowSize),
- column size (colSize),
- a set of integers (conditions)

* It iterates over each element of the array using nested loops.

* For each element, it assigns the current number to the variable currentNumber.

* It then checks if currentNumber is present in the set by iterating through the conditions set using a ranged-based for loop.

* If a match is found, it sets the presentInSet flag to true and breaks the loop.

* After the inner loop completes, it checks the value of presentInSet.

* If it is false, it means the number was not found in the set, so the function returns -1.

* If all numbers pass the condition check, the function returns 0, indicating that all numbers in the array are contained in the set.

*

*/

```
int checkConditions(int array[][4], int rowSize, int colSize, set<int> conditions)
```

```
{
```

```
    for (int row = 0; row < rowSize; row++)
```

```
    {
```

```
        for (int col = 0; col < colSize; col++)
```

```
        {
```

```
            int currentNumber = array[row][col];
```

```
            bool presentInSet = false;
```

```
            for (int num : conditions)
```

```
            { // range-based for loop. for each number in the set conditions do some action
```

```
                if (num == currentNumber)
```

```
                {
```

```
                    presentInSet = true;
```

```
                    break;
```

```
                }
```

```
            }
```

```
            if (!presentInSet)
```

```
            {
```

```
                return -1; // Number not found in the set, return -1
```

```
            }
```

```
        }
```

```
    }
```

```
    return 0; // All numbers in the array are contained in the set
```

```
}
```

QUESTION TWO

```

1  #include <iostream>
2
3  /*
4   * These statements make the specified class available to this program
5   */
6  using std::cin;
7  using std::cout;
8  using std::endl;
9
10 class Customer
11 {
12 public:
13     float base_charge;
14     float item_charge;
15     int items;
16
17     // initializes the base_charge, item_charge, and items member variables of the Customer class to 0
18     Customer() : base_charge(0), item_charge(0), items(0){};
19
20     bool membersEmpty()
21     {
22         // Check if any member variable is not equal to 0
23         return (base_charge == 0 || item_charge == 0 || items == 0);
24     }
25
26     float computeCharge()
27     {
28         // Calculate the total charge for the customer
29         float final_charge = base_charge + (items * item_charge);
30         return final_charge;
31     }
32 };
33
34 int main()
35 {
36     Customer person1, person2, person3;
37
38     // Get input and compute charge for person1
39     while (person1.membersEmpty())
40     {
41         cout << "[1]: Enter the base charge: ";
42         cin >> person1.base_charge;
43         cout << "[1]: Enter the number of items: ";
44         cin >> person1.items;
45         cout << "[1]: Enter the item charge: ";
46         cin >> person1.item_charge;
47         cout << "[1]: Charge: R" << person1.computeCharge() << "\n"
48             << endl;
49     }
50
51     cin.ignore(); // Clear the input buffer
52
53     // Get input and compute charge for person2
54     while (person2.membersEmpty())
55     {
56         cout << "[2]: Enter the base charge: ";
57         cin >> person2.base_charge;
58         cout << "[2]: Enter the number of items: ";
59         cin >> person2.items;
60         cout << "[2]: Enter the item charge: ";
61         cin >> person2.item_charge;
62         cout << "[2]: Charge: R" << person2.computeCharge() << "\n"
63             << endl;
64     }
65
66     cin.ignore(); // Clear the input buffer
67
68     // Get input and compute charge for person3
69     while (person3.membersEmpty())
70     {
71         cout << "[3]: Enter the base charge: ";
72         cin >> person3.base_charge;
73         cout << "[3]: Enter the number of items: ";
74         cin >> person3.items;
75         cout << "[3]: Enter the item charge: ";
76         cin >> person3.item_charge;
77         cout << "[3]: Charge: R" << person3.computeCharge() << endl;
78     }
79
80     return 0;
81 }
82

```

```
#include <iostream>
```

```
/*
```

```
 * These statements make the specified class available to this program
```

```
*/
```

```
using std::cin;
```

```
using std::cout;
```

```
using std::endl;
```

```
class Customer
```

```
{
```

```
public:
```

```
    float base_charge;
```

```
    float item_charge;
```

```
    int items;
```

```
    // initializes the base_charge, item_charge, and items member variables of the Customer class to  
    0
```

```
    Customer() : base_charge(0), item_charge(0), items(0){};
```

```
    bool membersEmpty()
```

```
{
```

```
    // Check if any member variable is not equal to 0
```

```
    return (base_charge == 0 || item_charge == 0 || items == 0);
```

```
}
```

```
    float computeCharge()
```

```
{
```

```
    // Calculate the total charge for the customer
```

```
    float final_charge = base_charge + (items * item_charge);
```

```
    return final_charge;
```



```
}  
};
```

```
int main()
```

```
{
```

```
    Customer person1, person2, person3;
```

```
    // Get input and compute charge for person1
```

```
    while (person1.membersEmpty())
```

```
    {
```

```
        cout << "[1]: Enter the base charge: ";
```

```
        cin >> person1.base_charge;
```

```
        cout << "[1]: Enter the number of items: ";
```

```
        cin >> person1.items;
```

```
        cout << "[1]: Enter the item charge: ";
```

```
        cin >> person1.item_charge;
```

```
        cout << "[1]: Charge: R" << person1.computeCharge() << "\n"
```

```
            << endl;
```

```
    }
```

```
    cin.ignore(); // Clear the input buffer
```

```
    // Get input and compute charge for person2
```

```
    while (person2.membersEmpty())
```

```
    {
```

```
        cout << "[2]: Enter the base charge: ";
```

```
        cin >> person2.base_charge;
```

```
        cout << "[2]: Enter the number of items: ";
```

```
        cin >> person2.items;
```

```
        cout << "[2]: Enter the item charge: ";
```

```
        cin >> person2.item_charge;
```

```
    cout << "[2]: Charge: R" << person2.computeCharge() << "\n"
        << endl;
}

cin.ignore(); // Clear the input buffer

// Get input and compute charge for person3
while (person3.membersEmpty())
{
    cout << "[3]: Enter the base charge: ";
    cin >> person3.base_charge;
    cout << "[3]: Enter the number of items: ";
    cin >> person3.items;
    cout << "[3]: Enter the item charge: ";
    cin >> person3.item_charge;
    cout << "[3]: Charge: R" << person3.computeCharge() << endl;
}

return 0;
}
```

QUESTION THREE


```

1  #include <iostream>
2
3  /*
4   * These statements make the specified class available to this program
5   */
6  using std::cin;
7  using std::cout;
8  using std::endl;
9
10 /*
11  * Stores a temperature value
12  * Modifies the temperature value
13  * Manages access to the temperature value
14  */
15 class Temperature
16 {
17 protected:
18     double temperature;
19
20 public:
21     Temperature() : temperature(0.0){};
22
23     void setTemperature(double temp)
24     {
25         temperature = temp;
26     }
27
28     double getTemperature()
29     {
30         return temperature;
31     }
32 };
33
34 /*
35  * Converts the stored temperature value.
36  * Supported temp conversions:
37  * | to celsius
38  * | to fahrenheit
39  */
40 class TempConverter : public Temperature
41 {
42 public:
43     void toCelsius()
44     {
45         double inputFahrenheit;
46         cout << "Enter a temperature in Fahrenheit: ";
47         cin >> inputFahrenheit;
48         setTemperature((inputFahrenheit - 32.0) * 5.0 / 9.0); // Convert Fahrenheit to Celsius using the formula
49     }
50
51     void toFahrenheit()
52     {
53         double inputCelsius;
54         cout << "Enter a temperature in Celsius: ";
55         cin >> inputCelsius;
56         setTemperature((inputCelsius * 9.0 / 5.0) + 32.0); // Convert Celsius to Fahrenheit using the formula
57     }
58 };
59
60 int main()
61 {
62     int numConversions, choice;
63
64     cout << "Enter the number of temperature conversions to perform: ";
65     cin >> numConversions;
66
67     cout << "Select the conversion type:" << endl;
68     cout << "Enter 1: Converts Celsius to Fahrenheit" << endl;
69     cout << "Enter 2: Converts Fahrenheit to Celsius" << endl;
70     cout << "\nEnter your choice: ";
71     cin >> choice;
72
73     for (int i = 1; i <= numConversions; i++)
74     {
75         TempConverter converter;
76
77         cout << "\nConversion " << i << ":" << endl;
78
79         if (choice == 1)
80         {
81             converter.toFahrenheit();
82             cout << "Fahrenheit temperature: " << converter.getTemperature() << endl;
83         }
84         else if (choice == 2)
85         {
86             converter.toCelsius();
87             cout << "Celsius temperature: " << converter.getTemperature() << endl;
88         }
89         else
90         {
91             cout << "Invalid choice -- Skipping conversion." << endl;
92             continue;
93         }
94     }
95
96     return 0;
97 }
98

```

```
#include <iostream>

/*
 * These statements make the specified class available to this program
 */

using std::cin;
using std::cout;
using std::endl;

/*
 * Stores a temperature value
 * Modifies the temperature value
 * Manages access to the temperature value
 */
class Temperature
{
protected:
    double temperature;

public:
    Temperature() : temperature(0.0){};

    void setTemperature(double temp)
    {
        temperature = temp;
    }

    double getTemperature()
    {
        return temperature;
    }
};
```

```

/*
 * Converts the stored temperature value.
 * Supported temp conversions:
 * | to celsius
 * | to fahrenheit
 */
class TempConverter : public Temperature
{
public:
    void toCelsius()
    {
        double inputFahrenheit;
        cout << "Enter a temperature in Fahrenheit: ";
        cin >> inputFahrenheit;

        setTemperature((inputFahrenheit - 32.0) * 5.0 / 9.0); // Convert Fahrenheit to Celsius using the
        formula
    }

    void toFahrenheit()
    {
        double inputCelsius;
        cout << "Enter a temperature in Celsius: ";
        cin >> inputCelsius;

        setTemperature((inputCelsius * 9.0 / 5.0) + 32.0); // Convert Celsius to Fahrenheit using the
        formula
    }
};

int main()
{

```

```
int numConversions, choice;
```

```
cout << "Enter the number of temperature conversions to perform: ";
```

```
cin >> numConversions;
```

```
cout << "Select the conversion type:" << endl;
```

```
cout << "Enter 1: Converts Celsius to Fahrenheit" << endl;
```

```
cout << "Enter 2: Converts Fahrenheit to Celsius" << endl;
```

```
cout << "\nEnter your choice: ";
```

```
cin >> choice;
```

```
for (int i = 1; i <= numConversions; i++)
```

```
{
```

```
    TempConverter converter;
```

```
    cout << "\nConversion " << i << ":" << endl;
```

```
    if (choice == 1)
```

```
    {
```

```
        converter.toFahrenheit();
```

```
        cout << "Fahrenheit temperature: " << converter.getTemperature() << endl;
```

```
    }
```

```
    else if (choice == 2)
```

```
    {
```

```
        converter.toCelsius();
```

```
        cout << "Celsius temperature: " << converter.getTemperature() << endl;
```

```
    }
```

```
    else
```

```
    {
```

```
        cout << "Invalid choice -- Skipping conversion." << endl;
```

```
        continue;
```



```
}
```

```
}
```

```
return 0;
```

```
}
```

References

Malik, D.S 2018. C++ Programming: Program Design including Data Structures. Eighth Edition.
United Kingdom: Cengage Learning