RICHFIELD

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

```cpp
#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, 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
}

```

#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

```cpp
#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;
}

```

```
#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

```cpp
#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;
}
```

```
#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