# Department of Electrical Engineering

**CS212**

**Object Oriented Programming**



# Lab 11: Template Functions

**Class**: BEE - 12C

**Date**: December 13th, 2021

**Time**: Monday (1400 – 1700)

**Name**: Muhammad Umer

**CMS ID**: 345834

**Tools**

* Microsoft Visual Studio 2013

**Lab Tasks**

* **Task 1**

**Create a C++ Template Function named store so that it accepts an array of 5 elements. A Template Function created will store an array of five elements of any given primitive data type. Define a function to print the contents of the array.**

## Code

#include <iostream>

#include <string>

#include <typeinfo>

using namespace std;

template <typename T>

T store()

{

    T arr[5];

    cout << "Input Data into the Array:\n";

    for (int i = 0; i < 5; i++) {

        cin >> arr[i];

    }

    cout << "The stored elements of the array are: ";

    for (int i = 0; i < 5; i++) {

        cout << arr[i] << " ";

    }

    return 0;

}

int main()

{

    store<int>();

    cout << endl;

    return 0;

}

**Terminal Output**

PS D:\NUST\Semester 3\Object Oriented Programming\Labs> lab11.exe

Input Data into the Array:

1 2 3 4 5

The stored elements of the array are: 1 2 3 4 5

If the main were to be slightly altered, so that instead of **int typename**, we passed in **char**.

int main()

{

    store<char>();

    cout << endl;

    return 0;

}

**Terminal Output**

Input Data into the Array:

a b c d e

The stored elements of the array are: a b c d e

* **Task 2**

**Create the C++ Template Function named add so that it has four parameters sum, x, and n1 and n2. The first two parameters will have the type represented by the function template type parameter T. n1 and n2 will always be int. The return type is void.**

A Template Function created from Add will compute:

**sum = 1 + x + 2x + 3x + ... + n1x + n2x**

## Code

#include <iostream>

#include <string>

#include <cctype>

using namespace std;

template < typename U, typename T >

U add(T & sum, T x, int n1, int n2) {

    if (isalpha(x)) {

        cout << "Computation of Sum\n";

        cout << "Sum: 1 + ";

        for (int i = 1; i <= n2; i++) {

            if (i == n2) {

                cout << to\_string(i) << "x";

            }

            else {

                cout << to\_string(i) << "x + ";

            }

        }

    }

    else {

        sum += 1;

        cout << "Computation of Sum\n";

        for (int i = 1; i < n2; i++) {

            sum += i \* x;

        }

        cout << "The final sum is: " << sum;

    }

}

int main() {

    int sum = 5, n1 = 10, n2 = 11;

    int x = 2;

    add < void, int >(sum, x, n1, n2);

    cout << endl;

    return 0;

}

**Terminal Output**

PS D:\NUST\Semester 3\Object Oriented Programming\Labs> lab11.exe

Computation of Sum

The final sum is: 116

Again, upon altering the main function, so that instead of x being an integer, it was a character, the output will transform as the following:

int main() {

    int sum = 5, n1 = 10, n2 = 11;

    char x = 'a';

    add < void, int >(sum, x, n1, n2);

    cout << endl;

    return 0;

}

**Terminal Output**

PS D:\NUST\Semester 3\Object Oriented Programming\Labs> lab11.exe

Computation of Sum

Sum: 1 + 1x + 2x + 3x + 4x + 5x + 6x + 7x + 8x + 9x + 10x + 11x

* **Task 3:**

**Write and Implement template functions for Matrix.**

**Each Matrix is two dimensional arrays with a no of columns and rows.**

**Create C++ Template Functions to add, subtract and multiply two matrices.**

## Code

#include <iostream>

#include <string>

#include <cctype>

using namespace std;

template <typename T>

T matrix(int dim)

{

    T\*\* sum = new T\*[dim];

    for (int i = 0; i < dim; i++) {

        sum[i] = new T[dim];

    }

    T\*\* diff = new T\*[dim];

    for (int i = 0; i < dim; i++) {

        diff[i] = new T[dim];

    }

    T\*\* mult = new T\*[dim];

    for (int i = 0; i < dim; i++) {

        mult[i] = new T[dim];

    }

    int i, j;

    int r = dim, c = dim;

    T\*\* ptr1 = new T\*[dim];

    for (int i = 0; i < dim; i++) {

        ptr1[i] = new T[dim];

    }

    T\*\* ptr2 = new T\*[dim];

    for (int i = 0; i < dim; i++) {

        ptr2[i] = new T[dim];

    }

    // Storing elements of first matrix entered by user.

    cout << endl

         << "Enter elements of 1st matrix: " << endl;

    for (i = 0; i < r; ++i) {

        for (j = 0; j < c; ++j) {

            cout << "Enter element A" << i + 1 << j + 1 << ": ";

            cin >> ptr1[i][j];

        }

    }

    // Storing elements of second matrix entered by user.

    cout << endl

         << "Enter elements of 2nd matrix: " << endl;

    for (i = 0; i < r; ++i) {

        for (j = 0; j < c; ++j) {

            cout << "Enter element B" << i + 1 << j + 1 << ": ";

            cin >> ptr2[i][j];

        }

    }

    // Displaying the resultant sum matrix.

    // Adding two matrices

    for (i = 0; i < r; ++i) {

        for (j = 0; j < c; ++j) {

            sum[i][j] = ptr1[i][j] + ptr2[i][j];

        }

    }

    cout << endl

         << "Sum of two matrices yields: " << endl;

    for (i = 0; i < r; ++i) {

        for (j = 0; j < c; ++j) {

            cout << sum[i][j] << "  ";

            if (j == c - 1)

                cout << endl;

        }

    }

    // Displaying the resultant diff matrix.

    // Subtracting two matrices

    for (i = 0; i < r; ++i) {

        for (j = 0; j < c; ++j) {

            diff[i][j] = ptr1[i][j] - ptr2[i][j];

        }

    }

    cout << endl

         << "Difference of two matrices yields: " << endl;

    for (i = 0; i < r; ++i) {

        for (j = 0; j < c; ++j) {

            cout << diff[i][j] << "  ";

            if (j == c - 1)

                cout << endl;

        }

    }

    // Displaying the multiplication of two matrix.

    // Initializing elements of matrix mult to 0.

    for (i = 0; i < dim; ++i) {

        for (j = 0; j < dim; ++j) {

            mult[i][j] = 0;

        }

    }

    // Multiplying matrix a and b and storing in array mult.

    for (i = 0; i < dim; ++i) {

        for (j = 0; j < dim; ++j)

            for (int k = 0; k < dim; ++k) {

                mult[i][j] += ptr1[i][k] \* ptr2[k][j];

            }

    }

    cout << endl

         << "Multiplication of two matrices yields: " << endl;

    for (i = 0; i < dim; ++i) {

        for (j = 0; j < dim; ++j) {

            cout << " " << mult[i][j];

            if (j == c - 1)

                cout << endl;

        }

    }

    return 0;

}

int main()

{

    int dimension;

    cout << "Input dimension of matrix: ";

    cin >> dimension;

    matrix<float>(dimension);

    cout << endl;

    return 0;

}

**Terminal Output**

PS D:\NUST\Semester 3\Object Oriented Programming\Labs> lab11.exe

Input dimension of matrix: 2

Enter elements of 1st matrix:

Enter element A11: 1.5

Enter element A12: 3.5

Enter element A21: 7

Enter element A22: 2.3

Enter elements of 2nd matrix:

Enter element B11: 7.8

Enter element B12: 3.2

Enter element B21: 2.1

Enter element B22: 2.0

Sum of two matrices yields:

9.3  6.7

9.1  4.3

Difference of two matrices yields:

-6.3  0.3

4.9  0.3

Multiplication of two matrices yields:

 19.05 11.8

 59.43 27