University of Burgundy

Software Engineering

Tutorial 1

Lab Report-1

Author:
Hardiksinh Parmar

Supervisor:
Dr. Yohan Fougerolle

October 14, 2018

1 Exercises

Input and Output in the console

2 Pass parameters to a function

2.0.1 Parameters by value::Swap program

```
//————————Param by values main.cpp

#include <iostream>
#include <test23.h>
using namespace std;

int main()
{
  int a = 9;
   int b = 3;
   int c;

   //swap1
    swap_1(a, b);
}
```

```
//————————Param by values test23.cpp

#include <iostream>
#include <test23.h>
using namespace std;

void swap_1(int a, int b) {
    cout<< "value␣for␣a:␣";
    cin>> a;
    cout<< "value␣for␣b:␣";
    cin>> b;
    //before swapping
    int x = 0;
    x = a;
    a = b;
    b = x;
    //after swapping
    cout << "New␣␣a␣is:␣" <<a<<endl;
    cout << "New␣␣b␣is:␣" <<b<<endl;
}
```

```
//—————param by values test23.h————
void swap_1(int, int);

                    ****
```

2.0.2 Param by reference

```
//—————Param by reference main.cpp

#include <iostream>
#include <test23.h>
using namespace std;

int main()
{
  int a = 9;
   int b = 3;
   int c;
//———————swap by refernece
   //swap2
    swap_2(a, b);

    //————————swap by pointers
    //swap3
    swap_3(x, y);
}
```

```
#include <iostream>
#include <test23.h>
using namespace std;

//—————Param by reference  test23.cpp
void swap_2(int & a, int & b) {

    cout<< "value␣for␣a:␣";
    cin>> a;
    cout<< "value␣for␣b:␣";
    cin>> b;
       //before swapping
    int x = 0;
    x = a;
    a = b;
    b = x;
       //after swapping
    cout << "New value of a is: "<<a<<endl;
    cout << "New value of b is: "<<b<<endl;
}

//---------Param by pointers  test23.cpp
void swap_3(int *a, int *b)
{
    int x=*a;
    *a=*b;
    *b=x;
    cout <<"a="<< *a <<endl;
    cout<<"b="<< *b <<endl;
}
```

```
//----------param by reference  test23.h-------
void swap_2(int&, int&);

//----------param by  pointers test23.h-------
void swap_3(int*, int*);
                         ****
```

2.1 Multiple Return values

```
//------ cartiesian to polar main.cpp

#include <iostream>
#include <test23.h>
using namespace std;


int main()
{ int a = 9;
   int b = 3;
   int c;

    //cartesian program
    double phi, theta;
    CartesianToPolar(a,b,phi,theta);
    cout << "Phi is: " <<phi<<endl;
    cout << "theta is: "<<theta<<endl;
}
```

```
//----------cartiesian to polar test23.cpp
void CartesianToPolar
    (const int a, const int b, double & phi, double & theta)
{
    phi = sqrt(a*a + b*b);
    theta = atan2(b, a);
}
```

```
//——————————cartiesian to polar test23.h
  void CartesianToPolar(const int, const int, double &, double &);

                  ***********
```

2.2 Default Values

```
//———————————is Multiple  main.cpp
#include <iostream>
#include <test23.h>
using namespace std;

int main()
{
   int a = 9;
   int b = 3;
   int c;
 // IsMultipleOf(a);

    IsMultipleOf(a, b);
}
```

```
//———————————is Multiple  test23.cpp
#include <iostream>
#include <test23.h>
using namespace std;

void IsMultipleOf(const int & p, const int & q ) {
    if (p % q == 0) {
        cout << p << " is a multiple of "<< q <<endl;
    }
    else cout << p << " is not a multiple of "<< q <<endl;
}
```

```
//———————————is Multiple  test23.h

void IsMultipleOf(const int&, const int& );

                       *********
```

2.3 Recursive function:: Prime number

```
//-----------prime num recursion main.cpp
#include <iostream>
#include <test23.h>
using namespace std;

int main()
{
   int a = 9;
   int b = 3;
   int c;

  //recursive prime

    cout<< "Enter a value to check if prime ";
    cin>>c;

    int p = Prime(c, c-1);
    if (p == 1){
        cout<< c <<" is a prime number"<<endl;
    }
    else {
        cout<< c <<" is not a prime number "<<endl;
    }
}
```

```
//-----------prime num recursion test23.cpp
#include <iostream>
#include <test23.h>
using namespace std;

bool Prime(const int& p, const int& q) {
    if(p<2)
        return 0;
    else if(q == 1)
        return true;
    else if(p % q == 0)
            return false;
    else
        return Prime(p, q - 1);
    }
```

```
//----------------prime num recursion test23.h
bool Prime(const int&, const int&);

                        *******
```

2.4 Monodimensional array

```
//————————monodimensional array main23.cpp
#include <iostream>
#include "test23.h"
using namespace std;
int main()
{
   int a = 9;
   int b = 3;
   int c;
  //monodimensional array

int index_of = 5;
cout <<"The "<<index_of<<"th"<<" Index of the Arrays are: "<<endl;
    ArraysEx1(index_of);
}
```

```
//——————————monodimensional array test23.h
int ArraysEx1(int index)
{
int arr[10] = {2, 8, 15, 10, 3, 7, 9, 4, 20, 77}
int *arr2 = new int[10] {24, 11, 12,20, 29, 34, 32, 17, 16, 15};

    cout<< "The index of static array : "<<arr[index]<<endl;
    cout<< "The index of dynamic array : "<<arr2[index]<<endl;

    delete[] arr2;

    return index;
}

                        ************
```

```
//————————————monodimensional array test23.h
int ArraysEx1(int);

                   ************
```

2.5 Bidimensional array-Pascal triangle revisted

```
//——————static bidimen pascal main.cpp

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

```
    int main()
    {
        const int ROW = 6, COL = 6;
        int k[ROW][COL] ={} ;
        k[0][0] = 0;
        cout << k[0][0] << endl;

        for ( int i = 1; i < ROW; i++ )
        {   k[i][0] = 1;
            k[i][i] = 1;
            cout << k[i][0] << "   ";

            for ( int j = 1; j < COL; j++ )
            {   k[i][j] = k[i-1][j] + k[i-1][j-1];
                cout << k[i][j] << "    ";
            }
            cout << k[i][i] << endl;
        }
     }
```

2.6 Matrix Multiplication

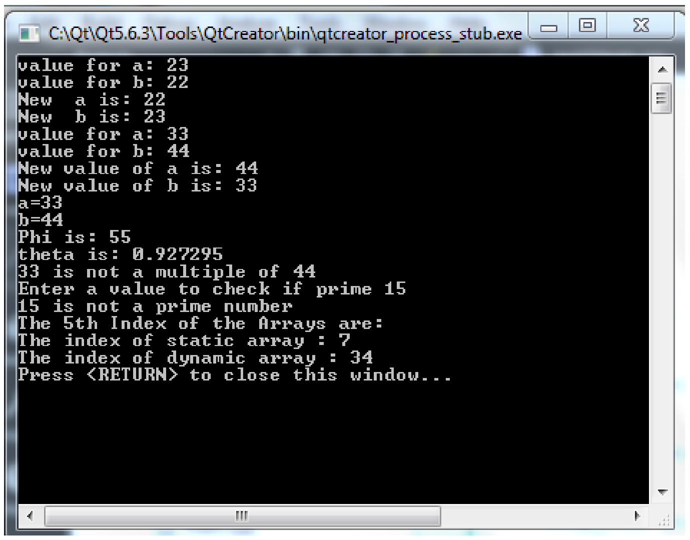
```
//---------------------------------------matMul main.cpp
#include <iostream>
using namespace std;

void enterData(int firMat[][10], int secMat[][10], int rowFir,
    int colFir, int rowSec, int colSec);
void multiplyMatrices(int firtMat[][10], int secMat[][10],
 int multRes[][10], int rowFir, int columnFir, int rowSec, int colSec);
void display(int mult[][10], int rowFir, int colSec);

int main()
{
    int firMat[10][10], secMat[10][10], mult[10][10], rowFirst,
     colFir, rowSec, colSec, i, j, k;

    cout << "Enter rows and column for first matrix: ";
    cin >> rowFirst >> colFir;

    cout << "Enter rows and column for second matrix: ";
    cin >> rowSec >> colSec;

    // If colum of first matrix in not equal to row of second
    matrix, asking user to enter the size of matrix again.
    while (colFir != rowSec)
    {
        cout << "Error! column of first matrix not equal to row of
        second." << endl;
        cout << "Enter rows and column for first matrix: ";
        cin >> rowFirst >> colFir;
        cout << "Enter rows and column for second matrix: ";
        cin >> rowSec >> colSec;
    }

    // Function to take matrices data
        enterData(firMat, secMat, rowFirst, colFir, rowSec, colSec);

   // Function to multiply two matrices.
        multiplyMatrices(firMat, secMat, mult, rowFirst, colFir,
        rowSec, colSec);

  // Function to display resultant matrix after multiplication.
        display(mult, rowFirst, colSec);

    return 0;
}
```

```
//——————————————————————matMul test23.cpp
#include <iostream>
using namespace std;

void enterData(int firMat[][10], int secMat[][10],
  int rowFir, int colFir, int rowSec, int colSec);
void multiplyMatrices(int firtMat[][10], int secMat[][10],
int multRes[][10], int rowFir, int columnFir, int rowSec, int colSec);
void display(int mult[][10], int rowFir, int colSec);

void enterData(int firMat[][10], int secMat[][10],
      int rowFir, int colFir, int rowSec, int colSec)
{     int i, j;
    cout << endl << "Enter elements of matrix 1:" << endl;
    for(i = 0; i < rowFir; ++i)
    {     for(j = 0; j < colFir; ++j)
        {         cout << "Enter elements a"<< i + 1 << j + 1 << ": ";
            cin >> firMat[i][j];
        }     }
    cout << endl << "Enter elements of matrix 2:" << endl;
    for(i = 0; i < rowSec; ++i)
    {         for(j = 0; j < colSec; ++j)
```

```
        {      cout << "Enter␣elements␣b" << i + 1 << j + 1 << ":␣";
            cin >> secMat[i][j];
        }     } }

void multiplyMatrices(int firMat[][10], int secMat[][10],
  int mult[][10], int rowFir, int colFir, int rowSec, int colSec)
{     int i, j, k;
    // Initializing elements of matrix mult to 0.
    for(i = 0; i < rowFir; ++i)
    {         for(j = 0; j < colSec; ++j)
        {             mult[i][j] = 0;
        }
    }

    // Multiplying matrix firstMatrix and secondMatrix and storing
      in array mult.
    for(i = 0; i < rowFir; ++i)
    {         for(j = 0; j < colSec; ++j)
        {             for(k=0; k<colFir; ++k)
            {             mult[i][j] += firMat[i][k] * secMat[k][j];
            }}}}

void display(int mult[][10], int rowFirst, int columnSecond){
    int i, j;
    cout << "Output␣Matrix:" << endl;
    for(i = 0; i < rowFirst; ++i){
        for(j = 0; j < columnSecond; ++j)
        {cout << mult[i][j] << "␣";
            if(j == columnSecond - 1)
                cout << endl << endl;
        }}}
```

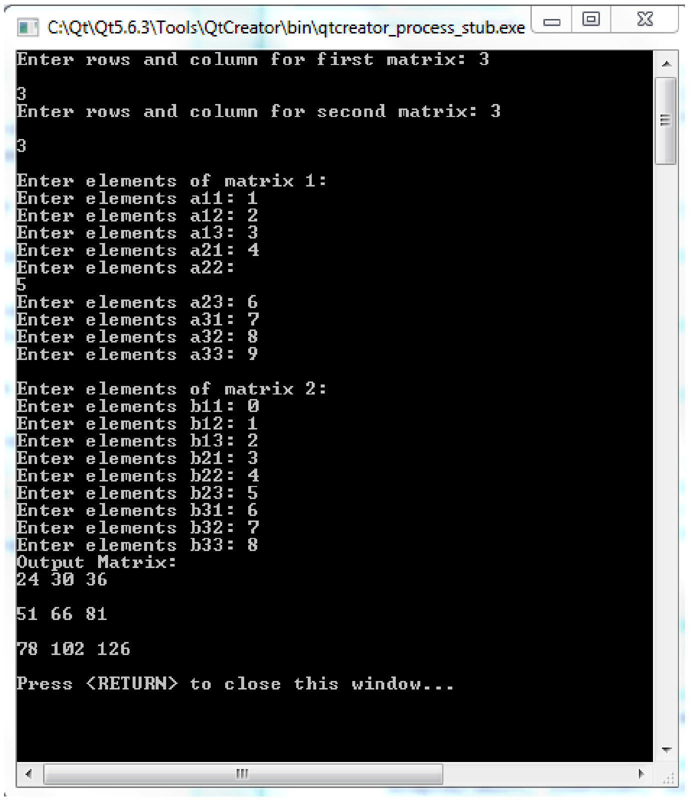
```
//————————————————————————————mulMat test23.h

void enterData(int firMat[][10], int secMat[][10], int rowFir,
int colFir, int rowSec, int colSec);

void multiplyMatrices(int firtMat[][10], int secMat[][10],
 int multRes[][10], int rowFir, int columnFir, int rowSec, int colSec);

void display(int mult[][10], int rowFir, int colSec);

                  ************
```

2.7 Outputs


C:\Qt\Qt5.6.3\Tools\QtCreator\bin\qtcreator_process_stub.exe

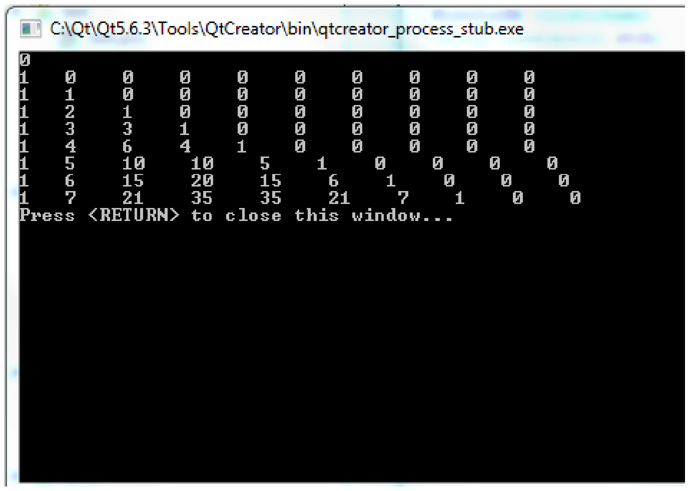
```
value for a: 23
value for b: 22
New  a is: 22
New  b is: 23
value for a: 33
value for b: 44
New value of a is: 44
New value of b is: 33
a=33
b=44
Phi is: 55
theta is: 0.927295
33 is not a multiple of 44
Enter a value to check if prime 15
15 is not a prime number
The 5th Index of the Arrays are:
The index of static array : 7
The index of dynamic array : 34
Press <RETURN> to close this window...
```

Figure 1: All outputs



```
C:\Qt\Qt5.6.3\Tools\QtCreator\bin\qtcreator_process_stub.exe
Enter rows and column for first matrix: 3

3
Enter rows and column for second matrix: 3

3

Enter elements of matrix 1:
Enter elements a11: 1
Enter elements a12: 2
Enter elements a13: 3
Enter elements a21: 4
Enter elements a22:
5
Enter elements a23: 6
Enter elements a31: 7
Enter elements a32: 8
Enter elements a33: 9

Enter elements of matrix 2:
Enter elements b11: 0
Enter elements b12: 1
Enter elements b13: 2
Enter elements b21: 3
Enter elements b22: 4
Enter elements b23: 5
Enter elements b31: 6
Enter elements b32: 7
Enter elements b33: 8
Output Matrix:
24 30 36

51 66 81

78 102 126

Press <RETURN> to close this window...
```

Figure 2: output of the MatMul Program



```
C:\Qt\Qt5.6.3\Tools\QtCreator\bin\qtcreator_process_stub.exe
0
1    0    0    0    0    0    0    0    0    0
1    1    0    0    0    0    0    0    0    0
1    2    1    0    0    0    0    0    0    0
1    3    3    1    0    0    0    0    0    0
1    4    6    4    1    0    0    0    0    0
1    5    10    10    5    1    0    0    0    0
1    6    15    20    15    6    1    0    0    0
1    7    21    35    35    21    7    1    0    0
Press <RETURN> to close this window...
```

Figure 3: output of the Pascal Program