**American International University Bangladesh (AIUB)**

****

**Faculty of science & Technology**

**Department of Computer Science**

**LAB MANUAL-01**

CSC 2211: Algorithms

|  |
| --- |
| **TITLE** |

**Algorithmic and Computational Thinking**

|  |
| --- |
| **PREREQUISITE** |

* To be able to write, build and run a C++ program in CodeBlocks.
* To be able to identify and understand the basic components of a C++ program
* Able to write user define function and to use the library function (e.g. math.h, rand()).
* Good knowledge to use Array/List as argument of functions.
* Good knowledge to use pointers as argument of functions and function return a pointer.
* Function call by reference and call by value (swap example)
* Dynamic memory allocation in C++
* Standard Template library(STL in C++)

|  |
| --- |
| **OBJECTIVE** |

* Algorithmic and Computational Thinking

|  |
| --- |
| **Step by step learn C++** |

* Download Code Blocks that is open source, cross platform, free C, C++ and Fortran IDE.
* C++ data types and variables
* Operator and expression
* Conditional structure (if, if else, if else if…. Else, switch)
* Loops(for, while, do while loop)
* **Functions**
* **Array**
* **Pointer**
* **Function, array and pointer**
* String
* Structure and class
* Object Oriented programming

**Basic building blocks of C++ program**

#include<iostream>

using namespace std;

int main()

{

double a, b, c;

cout<<"Enter two number: ";

cin>>a>>b;

c = a+b;

cout<<"the result: "<<c;

return 0;

}

**Data Type or variable:**

**C++ Fundamental Data Types:**

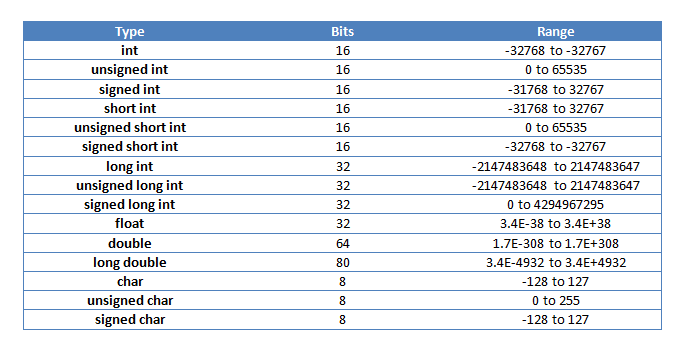
int ------ Integer ------ 2 or 4 bytes

float ------ Floating point ------ 4 bytes

double ------ Double Floating-point ------ 8 bytes

char ------ Character ------ 1 byte

bool ------ Boolean ------ 1 byte

****

#include<iostream>

using namespace std;

int main()

{

int myNum = 5; // Integer (whole number)

float myFloatNum = 5.99; // Floating point number

double myDoubleNum = 9.98; // Floating point number

char myLetter = 'D'; // Character

bool myBoolean = true; // Boolean

string myText = "Hello"; // String

return 0;

}

**C++ Arithmetic Operators**

|  |  |
| --- | --- |
| **Operator** | **Operation** |
| + | Addition |
| - | Subtraction |
| \* | Multiplication |
| / | Division |
| % | Modulo Operation (Remainder after division) |

**C++ Relational Operators**

|  |  |  |
| --- | --- | --- |
| Operator | Meaning | Example |
| = = | Is Equal To | 3 == 5 gives us **false** |
| != | Not Equal To | 3 != 5 gives us **true** |
| > | Greater Than | 3 > 5 gives us **false** |
| < | Less Than | 3 < 5 gives us **true** |
| >= | Greater Than or Equal To | 3 >= 5 give us **false** |
| <= | Less Than or Equal To | 3 <= 5 gives us **true** |

# Function, Array and Pointer

#include<bits/stdc++.h>

using namespace std;

void printList(int \*m, int s){

for(int i=0;i<s;i++){

cout<<m[i]<<" ";

}

}

int sumList(int \*m, int s){

int result = 0;

for(int i=0;i<s;i++){

result+= m[i];

}

return result;

}

int \*doubleList(int \*x, int s){

int \*result = new int[s];

for(int i=0;i<s;i++){

result[i]= x[i]\*2;

}

return result;

}

int main(){

int a[]={10,20,30,40};

printList(a,4);

cout<<"\nSum List= "<<sumList(a,4)<<endl;

cout<<"Double the list: ";

int \*p=doubleList(a,4);

printList(p,4);

return 0;

}

# 2D dynamic memory

#include<bits/stdc++.h>

using namespace std;

int \*\*Matrix(int r, int c)

{

int \*\*m = new int\*[r];

for(int i=0; i<c; i++)

{

m[i]= new int[c];

}

return m;

}

void printMatrix(int \*\*p, int rows, int colums)

{

for(int i=0; i<rows; i++)

{

for(int j=0; j<colums; j++)

{

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

}

cout<<endl;

}

cout<<endl;

}

void addValuse(int \*\*k,int r,int c){

for(int i=0; i<r; i++)

{

for(int j=0; j<c; j++)

{

k[i][j]=rand()%10;

}

}

}

int main()

{

srand(time(0));

int \*\*A = Matrix(5,5);

addValuse(A,5,5);

printMatrix(A,5,5);

int \*\*B = Matrix(5,5);

addValuse(B,5,5);

printMatrix(B,5,5);

return 0;

}

# Call By Reference

return 0;

}

|  |
| --- |
| **LAB WORK** |

Write the following functions in C++