**Lab # 15**

**Week = 20th January – 24th January-2025**

**Topics Covered:**

* Multiplication Operator Function

**Problem Statement:**

You need to write a program in which the following concepts must be implemented.

Write a class *Matrix* and overload multiplication (\*) operator for this class to multiply a matrix by a scalar (2) and display the original and resultant matrix.  
Write following functions in the class.  
1. **read()**: To get input for number of rows and columns of matrix.  
2.**input():** To get elements of matrix from user.   
3.**display():** To display the elements of matrix.

**Solution:**

#include<iostream>

using namespace std;

class Matrix

{

private:

int \*\*A; // pointer to store 2 dimensional array

int x, y;

public:

Matrix();//default constructor

Matrix(int, int);//parametrized constructor

void read();

void input();

void display();

Matrix operator \* ( int d) const;//overloading \* operator

};

Matrix Matrix :: operator \* ( int d) const

{

Matrix temp(\*this);//temp is an object of the class Matrix

for ( int i = 0; i < x; i++)//outer For Loop

{

for (int j = 0; j < y; j++)//Inner For loop

{

temp.A[i][j] \*= d;// Multiplting elements of of temp object with d

}

}

return temp; //returning the temp object

}

Matrix :: Matrix()//initializing default constructor

{ x = 0;

y = 0;

}

Matrix :: Matrix(int a, int b)//initializing parametrized constructor

{ x = a;

y = b;

}

void Matrix :: read()//Defining read function

{

cout << "Enter number of rows " << endl;

cin >> x;

cout << "Enter number of columns " << endl;

cin >> y;

}

void Matrix :: input()//Defining inout function

{

int num1, num2, k;

int i, j;

A = new int \*[x]; //Handling pointer to store 2 dimensional array

for (k = 0; k < x; k++)

A[k] = new int [y];

for (num1=0; num1<x; num1++)//Initializing default values to array

{

for (num2=0; num2<y; num2++)

{

A[num1][num2] = 0;

}

}

for ( i = 0; i < x; i++)//Initializing values given by user

{

for ( j = 0; j < y; j++)

{

cout<<"Enter "<<i<<","<<j<<" element"<<endl;

cin >> A[i][j];

}

}

}

void Matrix :: display()//Display function to display the elements of matrix

{

int i, j;

for (i=0; i<x; i++)//outer For loop

{

for (j=0; j<y; j++)//Inner For loop

{

cout << " " << A[i][j];//Displaying matrix element

}

cout << "\n";

}

}

int main ()

{

Matrix M1, M2;//Making two objects of Matrix cLass

M1.read ();//Taking number of rows and columns for object M1

M1.input ();//Taking values of elements for object M1

cout<<"The entered matrix is"<<endl;

M1.display ();//Displaying M1

M2 = M1\*2;//Multiply M1 with 2 and store the result in M2

cout<<"Matrix after multiplication by 2 is"<<endl;

M2.display();//Displaying M2

return 0;

}

**Lab # 14**

**Week = 13th January – 17th January-2025**

**Topics Covered:**

* Template Function
* Nested Class

**Problem Statement:**

You need to write a program in which the following concepts must be implemented.

Write a template function name “Add” which accepts two arguments of the same type int, float, double from the user then add those arguments, return their sum and display them on the screen.

Write a class named as firstClass, also write another class named as secondClass which should be nested inside the firstClass. Define a function named displayMessage inside the secondClass, this method should print the message “Inside the second class” on the screen. In the main function create the object of the secondClass and invoke the displayMessage method using the secondClass object.

Solution:

#include <iostream>

using namespace std;

class firstClass { //Declaration of outer class “First Class”

public:

class secondClass { //Declaration of inner/nested class “Second Class”

public:

void displayMessage(){ //Declaration and definition of member function of inner/nested class “Second Class”

cout<< "Function inside the inner class" <<endl;

}

}; //End of inner/nested class “First Class”

}; //End of outer class “Second Class”

template<class T> //Declaration of “Template Function” header

T Add (T x, T y) { //Declaration and definition of template function “Add()”

return x + y;

}

int main() {

firstClass :: secondClass object; // creating object of the second class

object.displayMessage(); //call to member function of inner/nested class “Second Class”

int integerOne = 3, integerTwo = 5; //int variables as argument to template function Add()

float floatOne = 12.34, floatTwo = 894.4; //float variables as argument to template function Add()

double doubleOne = 1236.58, doubleTwo = 8945.685; //double variables as argument to template function Add()

//call to template function Add() with integer arguments.

cout<< "Addition of Two integer Number is = " << Add(integerOne,integerTwo) <<endl;

//call to template function Add() with float arguments.

cout<< "Addition of Two floating point Number is = " << Add(floatOne,floatTwo) <<endl;

//call to template function Add() with float arguments.

cout<< "Addition of Two double Number is = " << Add(doubleOne , doubleTwo) <<endl;

return 0;

}

# **Lab # 13**

**Week = 6th January – 10th January-2025**

**Topics Covered**

* Overloading of stream extraction and insertion operator
* Overloading of ‘+’ operator for strings
* File Handling

Write a program in C++ to create a class name “**String**”. The String class should have a single data member as a character array and a default constructor to initialize the character array with default values.

The String class should overload the stream extraction >> and insertion operator **<<**.

The String class should also overload the **“+”**operator to concatenate the String.

Inside, the main function, create two objects of the String class. Take input from the user in those objects as First Name and Last Name and output the Full Name by merging the two inputs using the **“+”** operator. When taking the first and second input, the program should store the input in a “Sample.txt” file and similarly store the output in the same file.

**Sample Output:**

Enter the First Name: Ali  
Enter the last Name : Ahmed  
Full Name : Ali Ahmed

The contents inside the file should be like this

Ali  
Ahmed  
AliAhmed

**Solution:**

#include <iostream> //header file containing info (like function prototypes etc.) about cout and cin objects.

#include <stdlib.h> //a general purpose standard library in C.

#include <cstring> //header file containing info about string manipulation functions.

#include <fstream> //header file containing info about functions relating creating, reading and writing files.

using namespace std;

ofstream outfile; //Since outfile is global, it can be used in all functions in this program.

class String {

private:

char txt [20]; //an array of 20 characters.

public:

String () { //a default constructor.

strcpy(txt,""); // In **strcpy** () is function, the data member **txt** is initialized to an empty string.

}

friend istream &operator >> (istream &input, String &s); // insertion and extraction operators are overloaded.

friend ostream &operator << (ostream &output, const String &s); //Since these are not member functions of this class, so declared as friend to this class.

//the + operator is overloaded to concatenate two strings.

String operator + (const String &s1) {

outfile.open ("Sample.txt", ios :: app); //a file Sample.txt is opened in **append** mode so that previous contents of the file shouldn’t be overwritten.

String temp; //an object of class String is created.

strcat (txt, s1.txt); //Two strings are concatenated.

strcpy (temp.txt, txt); //string in txt is copied in txt belonging to object temp.

outfile << temp.txt; //string in txt is written in the file Sample.txt.

outfile << "\n"; //a new line is inserted in the file.

outfile.close (); //file Sample.txt is closed.

return temp; //the object temp is returned.

}

};

//extraction operator >> is overloaded in such a way that it takes istream object as a parameter, uses it to get a string from the user and then write this string in a file.

istream & operator >> (istream & in, String &s) {

outfile.open ("Sample.txt", ios :: app); //a file Sample.txt is opened in **append** mode.

in. getline (s.txt,20); //getline() is one of the functions belonging to istream object **input**.

outfile << s.txt; //string in txt is written in the file Sample.txt.

outfile << "\n"; //a new line is inserted in the file.

outfile.close (); //file Sample.txt is closed.

return in; //the object temp is returned.

}

//insertion operator << is overloaded in such a way that it takes ostream object as a parameter and display it on the screen.

ostream & operator << (ostream &out, const String &s) {

out << s.txt; //string in txt is displayed on the screen.

return out; //the object out is returned.

}

int main () {

String s1, s2; //two instances of class String are created using default constructor.

cout << "Enter First Name:"; //here overloaded << operator is not called because passed parameter is a string not an object of class String.

cin >> s1; //overloaded >> operator is called: an istream object **cin** and an object **s1** of class String is passed.

cout << "Enter Last Name:";

cin >> s2; //overloaded >> operator is called: an istream object **cin** and an object **s2** of class String is passed.

String s3 = s1+s2; //overloaded + operator is called which concatenates the two strings belonging to objects s1 & s2 and return a concatenated string which is stored in object s3.

cout << "FullName is " << s3; //here the first insertion operator << simply displays the string on the screen while the 2nd << operator is overloaded because an object s3 is passed.

}

# **Lab # 12**

**Week = 30th-December – 3rd-January-2025**

**Topics Covered:**

* Array of objects

Write a program in C++ to create two classes named “**Rectangle**” and “**String**” with the following information:

For **Rectangle** class, create two data members **length**, **breadth** of type double, a default constructor and a parameterized constructor. When a default constructor gets called, it should print “Default constructor of Rectangle class” and similarly when parameterized constructor is called, it should print “Parametrized constructor of Rectangle class”. Also, the body of the parametrized constructor should initialize the data members with the passed values.

For **String** class, create a character pointer **text** and a default constructor. When a default constructor gets called it should print “Default constructor of String class”. Also create a destructor for the string class and it should print “Destructor of String class”.

In the main function, create an array of five objects for class “Rectangle” and initialize first two array objects with random values. Similarly, in the main function dynamically create an array of five objects for class “String” using ***new*** operator. In the end, use the **delete** operator to deallocate the dynamically created objects for the String class.

**Solution:**

#include<iostream>

using namespace std;

class Rectangle {// declaring and defining class Rectangle

private:

double length; // Private members

double breadth;

public:

Rectangle(){ // default constructor of class Rectangle

cout << "Default Constructor of Rectangle class is called" << endl;

}

Rectangle(double l, double b){ //parameterize constructor of class Rectangle

length = l;

breadth = b;

cout << "Parameterized Constructor of Rectangle class is called" << endl;

}

};

class String { //declaring and defining class String

private:

char \* text; //character pointer text

public:

String(){//constructor of class String

cout << "Default Constructor of String class is called" << endl;

}

~String(){//destructor of class String

cout << "Destructor of String class is called" << endl;

}

};

int main(){

// creating array of objects of Rectangle class

Rectangle rectangle[5] = { Rectangle(4.6,2.3) , Rectangle(7.6,4.3) };

// dynamically creating an array of String class

String \* str;

str = new String [5];

// delete operator to deallocate memory

delete []str;

return 0;

}

**Lab # 11**

**Week = 23rd-December – 27th-December-2024**

**Topics Covered:**

* Operator Overloading

**Problem Statement:**

Write a program in C++ that add two class objects by overloading “plus (+)” operator. You are required to create a class named “MathClass“ and declare the class member and member functions. Also declare a class data member named as “number” and a parameterized constructor which take one argument and initializes the number. Define the ‘+’ operator overloaded function to add two object’s numbers. Also define another member function named as “Display ()” that shows the calculation result. Create three class objects for example: obj1, obj2 and result. Values are passed by calling the parameterized constructor in main () function. Add two object values by calling the ‘+’ operator overloaded function and then call display () function using obj1, obj2 and result.

**Solution:**

#include <iostream>

using namespace std;

class MathClass //Class defination

{

private:

int number; //private member of the MathClass

public:

MathClass() //Constructor of the MathClass

{

number = 0 ; //Public member of MathClass

}

MathClass(int x) // Parametrized Constructor

{

number = x ;

}

MathClass operator +(MathClass m) //Operator overloading

{

MathClass temp;

temp.number= number+m.number;

return temp;

}

void Display() { cout<<"Result: "<<number<<endl; } // Function to display the result

};

int main()

{

MathClass first(4), second(2), result; //value passing

result = first + second; // Addition of two numbers

result.Display();

system("pause");

}

**Lab # 10**

**Week = 16th-December – 20th-December-2024**

**Topics Covered:**

* Classes
* Constructors and their types

**Problem Statement:**

Write a C++ program which implement a class named “**Employee**”. This class has the following data members:

* Char string name
* Double id
* Character string gender
* Integer age

You have to implement the default and a parameterized constructor for this class.

Write getters and setters for each data member of the class and also implement a member function named “**display**” that will output the values of these data members for the calling object.

In the main () function, you need to create two objects of class “Employee”. Initialize one object with default constructor and other with parameterized constructor. Also show the default values of both objects with the display function.

In the end, update the values of both objects using setter functions and then show the updated values of data members of both objects using getter functions.

**Solution:**

#include <iostream> //Including iostream header file for input/output operations

#include <cstring> // Including cstring header file for string functions

using namespace std;

class Employee { //Class is defined

private:

char name[30];

char gender[10];

double id;  
 int age;

public:

Employee(); // Default Constructor

Employee(char[], char[], int, double); // Parametrized Constructor // Class public member functions   
 void setName(char[]); //setter functions of the class

void setGender(char[]);

void setAge(int);

void setId(double);

//getter functions of the class

char\* getName();

char\* getGender();

int getAge();

double getId();

void display(); //Display function

};

Employee::Employee() // Default constructor to initialize variables

{

strcpy(name, "Empty");

strcpy(gender, "Empty");

age = 0;

id = 0.0;

}

Employee::Employee(char Name[], char Gender[], int Age, double Id) // Parameterized constructor to initialize variables

{

strcpy(name, Name);

strcpy(gender, Gender);

age = Age;

id = Id;

}

void Employee::setName(char Name[]) // Setter function to set name

{

strcpy (name, Name);

}

void Employee::setGender(char Gender[]) // Setter function to set gender

{

strcpy (gender, Gender);

}

void Employee::setAge(int Age) // Setter function to set age

{

age = Age;

}

void Employee::setId(double Id) // Setter function to set ID

{

id = Id;

}

char\* Employee::getName() //Getter function to get Name

{  
return name;  
}

char\* Employee::getGender() //Getter function to get Gender{  
 return gender;

}

int Employee::getAge() //Getter function to get Age

{

return age;

}

double Employee::getId() //Getter function to get ID

{

return id;

}

void Employee::display() // Displaying data

{

cout<<endl<< "The name of the Employee is " << name <<endl;

cout<< "The gender of the Employee is " << gender <<endl;

cout<< "The age of the Employee is " << age <<endl;

cout<< "The Id of the Employee is " << id <<endl;

}

int main () {

Employee e1; // Declare an object of class Employee

Employee e2("Amir", "Male", 35, 166); // calling parameterized constructor

e1.display(); // Accessing member function

e2.display(); // Accessing member function

e1.setName("Asif"); //Setting name for e1 class object

e1.setGender("Male"); //Setting gender for e1 class object

e1.setAge(30); //Setting age for e1 class object

e1.setId(162); //Setting ID for e1 class object

cout<<endl;

cout<< "The name of the Employee is " << e1.getName() <<endl; // displaying Name of employee

cout<< "The gender of the Employee is " << e1.getGender() <<endl; // displaying gender of employee  
cout<< "The age of the Employee is " << e1.getAge() <<endl; // displaying age of employee

cout<< "The Id of the Employee is " << e1.getId() <<endl; // displaying id of employee

system("pause");

return 0;

}

**Lab # 9**

**Week = 9th-December – 13th-December-2024**

**Topics Covered:**

* Function Overloading

**Problem Statement:**

Write a program that overloads function named “myfunc” for integer, double and character data types. For example, if an integer value is passed to this function, the message "**using integer myfunc"** should be printed on screen, on passing a double type value, the message "**using double myfunc"** and on passing character value, the message "**using character myfunc"** should get printed.

**Solution:**

#include <iostream> //Including iostream header files

using namespace std;

int myfunc (int x); // function for integer data type parameter

double myfunc (double y); // function for double data type parameter

char myfunc(char z); // function for character data type parameter

main()

{

cout<<myfunc(5) << "\n"; //calling function for integer type

cout<<myfunc(5.5) << "\n"; //calling function for float type

cout<<myfunc('x') << "\n"; //calling function for character type

system("pause");

}

int myfunc(int x) //function definition for integer type

{

cout<< "Using integer myfunc()\n"; //printing integer value

return x;

}

double myfunc(double y) //function definition for float type

{

cout<< "Using float myfunc()\n"; //printing float value

return y;

}

char myfunc(char z) //function definition for character type

{

cout<< "Using char myfunc()\n"; //printing character value

return z;

}

**Lab # 8**

**Week = 2nd-December – 6th-December-2024**

**Topics Covered:**

* AND, OR operators
* Bitwise manipulation

**Problem Statement:**

Write a program which declare two variables of integer type and take their values as input from user. Also, perform the following operations on them and print their values.

1. Logical AND
2. Bitwise AND
3. Logical OR
4. Bitwise OR

**Solution:**

#include <iostream> //Including iostream header files

using namespace std;

int main(){

int var1 = 0, var2 = 0, var3 = 0; //Declare and intialize 3 integer variables

cout<< "Enter First Value: "; //Prompt to get first variable value

cin>> var1; // Save value in first variable

cout<< "Enter Second Value: "; //Prompt to get second variable value

cin>> var2; // Save value in second variable

var3 = var1 && var2; // Performing logical AND using && operators

cout<<"Logical AND is "<< var3 <<endl; // Printing result of Logical AND

var3 = var1 & var2; // Performing Bitwise AND using & operators

cout<<"Bitwise AND is "<< var3<<endl; // Printing result of Bitwise AND

var3 = var1 || var2; // Performing logical OR using || operator

cout<<"Logical OR is "<< var3 <<endl; // Printing result of Logical OR

var3 = var1 | var2; // Performing Bitwise OR using | operator

cout<<"Bitwise OR is "<< var3<<endl; // Printing result of Bitwise OR

}

**Lab # 7**

**Week = 25th-November – 29th-November-2024**

**Topics Covered:**

* Structures

- Declaration of a Structure

- Initializing Structures

- Functions and structures

**Problem Statement:**

Write a program which has:

1. A structure with only two member variables of integer and float type.
2. Initialize two data members by assigning 0 values in different ways.
3. Take input from user to assign different values to both structure variables.
4. Write a function that takes two structure variables as parameters.
5. This function must return a variable of structure type.
6. Function must add the data members of two passed structures variables and store the values in a new structure variable and print it on the screen.

**Solution:**

#include <iostream>//Including header files

using namespace std;

//Structure with two Data Members

struct MyStruct{

int i;

float f;

}

ms3 = {0,0.0}, // To show different methods of initializing structure

ms4 = {0,0.0};

//Function to add two structure variables

MyStruct add(MyStruct ms1, MyStruct ms2){

MyStruct ms3; //declaring an object of structure

ms3.i = ms1.i + ms2.i; //adding integer variables

ms3.f = ms1.f + ms2.f; //adding float variables

return ms3; //returning the resultant obejct

}

main(){

//Initialize structure variable values

MyStruct ms1 = {0,0.0};

struct MyStruct ms2 = {0,0.0};

//Taking input from user for first object

cout <<"Enter integer value of First structure variable ";

cin>> ms1.i;

cout <<"Enter float value of First structure variable ";

cin>> ms1.f;

//Taking input from user for second object

cout <<"\n Enter integer value of Second structure variable ";

cin>> ms2.i;

cout <<"Enter float value of Second structure variable ";

cin>> ms2.f;

//Display the values of first and second structure

cout<< "\n values of data members of first structure variable\n\n";

cout << ms1.i << "\t" << ms1.f << endl;

cout<< "values of data members of Second structure variable\n\n";

cout << ms2.i << "\t" << ms2.f <<endl;

MyStruct ms3= add(ms1,ms2);//Adding structure 1 and structure 2

cout<< "values of data members of structure variable returned from function\n\n";

cout << ms3.i << "\t" << ms3.f <<endl; //Display the values of resultant structure

system("pause");

}

**Lab # 6**

**Week = 18th-November – 22nd-November-2024**

**Topics Covered:**

* Multi-dimensional Arrays

**Problem Statement:**

Write a program in which you need to declare a multidimensional integer type array of size 4\*4. In this program:

* You should take input values from the users and store it in 4\*4 matrix.
* Display this matrix on the screen.
* Also, display the transpose of this matrix by converting rows into cols.

**Note**: Use different functions for above three point’s functionalities.

**Solution:**

#include <iostream>

using namespace std;

const int arraySize = 4; // intializing constant variable with value 4

void readMatrix(int arr[][arraySize]); // prototype of a function that takes input values from the user and will store it into the array.

void displayMatrix(int a[][arraySize]); // prototype/signature of the function to display array values

void transposeMatrix(int a[][arraySize]); // protype of a function that will change rows into columns and columns into rows.

main(){

// Declaring a multi-dimensional array

int a[arraySize][arraySize];

// Taking input from user

readMatrix(a); // Call by value of a function to store input values in the array.

// Display the matrix

cout << "\n\n" << "The original matrix is: " << '\n';

displayMatrix(a); // Calling a function that is written to display array values from a multi-dimensional array.

//Taking Transpose of a matrix

transposeMatrix(a); //Calling a function that will return swapped value at the end.

//Display the transposed matrix

cout << "\n\n" << "The transposed matrix is: " << '\n';

displayMatrix(a); // Calling a function that is written to display array values from a multi-dimensional array. This time it will show swapped values.

}

// Defining a readMatrix() function, in which we are using nested for loop to build a multi-dimensional matrix.

void readMatrix(int arr[arraySize][arraySize]){

int row, col;

for (row = 0; row < arraySize; row ++){

for(col=0; col < arraySize; col ++){

cout << "\n" << "Enter " << row << ", " << col << " element: ";

cin >> arr[row][col];

}

cout << '\n';

}

}

// Defining displayMatrix() function, that will traverse through the whole array and will display array elements.

void displayMatrix(int a[][arraySize]){

int row, col;

for (row = 0; row < arraySize; row ++){

for(col = 0; col < arraySize; col ++){

cout << a[row][col] << '\t';

}

cout << '\n';

}

}

// A function that is swapping whole rows into columns and columns into arrays.

void transposeMatrix(int a[][arraySize]){

int row, col;

int temp;

for (row = 0; row < arraySize; row ++){

for (col = row; col < arraySize; col ++){

/\* Interchange the values here using the swapping mechanism \*/

temp = a[row][col]; // Save the original value in the temp variable

a[row][col] = a[col][row];

a[col][row] = temp; //Take out the original value

}

}

}//end of lab=6

**Lab # 5**

**Week = 11th-November – 15th-November-2024**

**Topics Covered:**

* Array Manipulation
* Pointers

**Problem Statement:**

1. Write a program that will take two strings as input from the user in the form of character arrays namely as “firstArray” and “secondArray” respectively.
2. Both arrays along with the size will be passed to the function ***compareString***.
3. ***compareString*** function will use pointer to receive arrays in function and then start comparing both arrays using ***while*** loop.
4. If all the characters of both these arrays are same then the message “Both strings are same” should be displayed on the screen.

**Note:** For comparing both these arrays, the size should be same.

#include <iostream> // header file for input/output streams

#include <cstring> // header file for manipulating strings

using namespace std;

//function definition

void compareString(char \*array1, char \*array2, int array\_size){

//Initializing variables

int flag = 1;

int i = 0;

//Start of while loop

while(i<array\_size){

if(array1[i] != array2[i]) //Comparing each character of arrays

{

flag = 0; //Set Variable value to 0

break; // Terminate from loop

}

i++; //Incrementing variable value by 1

}

if(flag == 1) //Check whether variable value is 1

cout<< "Both strings are same!" <<endl;

else

cout<< "Both strings are not same!" <<endl;

}

int main() {

//Declaring character array of size 20

char firstArray[20];

char secondArray[20];

cout<< "Enter the First Name: ";

cin.getline(firstArray, sizeof(firstArray)); //First String Input from user

cout<< "Enter the Second Name: ";

cin.getline(secondArray, sizeof(secondArray));//Second String Input from user

if(strlen(firstArray) == strlen(secondArray)) //Comparing size of both arrays

{

int array\_size = strlen(firstArray); //Getting the size of the string

compareString(firstArray, secondArray, array\_size); //Calling compareString function

}

else {

cout<< "Size of both arrays are not same" <<endl;

}

}

**Lab # 4**

**Week = 4th-November – 8th-November-2024**

**Topics Covered:**

* Arrays
* Functions

**Problem Statement:**

Write a program in which you have to declare an integer array of size 10 and initializes it with numbers of your choice. Find the maximum and minimum number from the array and output the numbers on the screen.

For finding the maximum and minimum numbers from the array you need to declare two functions **findMax** and **findMin** which accept an array and size of array (an int variable) as arguments and find the max min numbers, and return those values.

**Solution:**

#include <iostream>

using namespace std;

//functions declaration

int findMin(int [],int);

int findMax(int [],int);

int main() {

const int SIZE = 10; //size of array

//Array initialization

int number[10] = {

21,25,89,83,67,81,52,100,147,10

};

//Displaying minimum and maximum number

cout<< "Maximum number in the array is :" <<findMax(number, SIZE) <<endl;

cout<< "Minimum number in the array is :" <<findMin(number, SIZE) <<endl;

return 0;

}

//Definition of findMin function

int findMin(int array[],int size){

int min = 0;

min = array[0];//Storing the value of the first element of array in 'min' variable

for (int i = 0; i<size; i++){ //loop for traversing array

if(min > array[i])//Testing if the value of 'min' variable is greater than the current element of array

min = array[i];//Storing the value of current element of array in 'min' variable

}

return min; //returning the minimum value of the array

}

//Definition of findMax function

int findMax(int array[],int size){

int max = 0;

max = array[0];//Storing the value of the first element of array in 'max' variable

for (int i = 0; i<size; i++){//loop for traversing array

if(max < array[i]) //Testing if the value of 'max' variable is less than the current element of array

max = array[i];//Storing the value of current element of array in 'max' variable

}

return max; //returning the maximum value of the array

}

**Lab # 3**

**Week = 28th-October – 01st-November-2024**

**Topics Covered:**

* Functions
* Repetition Structure (Loop)

**Problem Statement:**

Write a program in which you have to define a function displayDiagnol which will have two integer arguments named rows and cols. In the main function, take the values of rows and columns from the users. If the number of rows is same as numbers of columns then call the displayDiagnol function else show a message on screen that number of rows and columns is not same.

The following logic will be implemented inside the displayDiagnol function:

The function will take the value of rows and cols which are passed as argument and print the output in matrix form. To print the values in the matrix form, nested loops should be used. For each loop, you have to use a counter variable as counter. When the value of counters for each loop equals, then it prints the value of row at that location and prints hard coded zero at any other location.

Example if the user enters rows and cols as 3, then the output should be like this

1 0 0

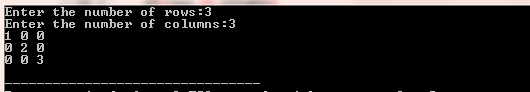
0 2 0

0 0 3

Example: when rows and columns are not same.



Example: when rows and columns are same.



**Solution:**

#include <iostream>

using namespace std;

void displayDiagonal(int,int); // function declaration

int main(){

int rows, columns;//variable declaration and initialization

rows = 0;

columns = 0;

cout<< "Enter the number of rows:";//Taking number of rows as input

cin>> rows;

cout<< "Enter the number of columns:";//Taking columns of rows as input

cin>> columns;

if(rows == columns)//conditional check for square matrix

displayDiagonal(rows,columns); // call function

else

cout<< "Wrong input! Num of rows should be equal to num of columns";

return 0;

}

// function definition

void displayDiagonal(int rows, int columns){

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

for (int j = 1; j<=columns; j++){

if(i==j)//diagonal check

cout<<i<< " "; //displaying element

else

cout<< 0 << " ";

}

cout<< "\n";

}

}

**Lab # 2**

**Week = 28th-October – 01st-November-2024**

**Topics Covered:**

* Repetition Structure (Loop)

**Problem Statement: While Loop**

“Calculate the average age of a class of ten students using while loop. Prompt the user to enter the age of each student.”

* We need 10 values of variable age of int type to calculate the average age.

int age;

* “Prompt the user to enter the age of each student” this requires cin>> statement.

For example:

cin>> age;

* Average can be calculated by doing addition of 10 values and dividing sum with 10.

TotalAge = TotalAge + age1;

AverageAge = TotalAge / 10;

**Solution:**

#include<iostream>

using namespace std;

main() {

**// declaring variable age to take input**   
 int age=0;  
   
 **// declaring variables to calculate totalAge and averageAge**  
 int TotalAge = 0, AverageAge = 0;

**// declaring ageCounter variable to check the number of iterations**   
int ageCounter=0;  
  
**// comparing the value of ageCounter**

while (ageCounter <10)

{

cout<<"please enter the age of student "<<++ageCounter<<" :\t";

cin>>age;

**//calculate totalAge by adding age of all students**

TotalAge = TotalAge + age;

}

cout<< "Total Age of 10 students = "<<TotalAge<<endl;  
 **// calculate AverageAge by dividing the totalAge with number of students**  
 AverageAge = TotalAge/10;

**//Display the result (average age)** cout<<"Average of students is: "<<AverageAge;

}

**Lab # 1**

**Week = 21st-October – 25th-October-2024**

**Topics Covered:**

* Variables
* Data Types
* Arithmetic Operators
* Precedence of Operators

**Problem Statement:**

“Calculate the average age of a class of ten students. Prompt the user to enter the age of each student.”

* We need 10 variables of int type to calculate the average age.

int age1, age2, age3, age4, age5, age6, age7, age8, age9, age10;

* “Prompt the user to enter the age of each student” this requires cin>> statement.

For example:

cin>> age1;

* Average can be calculated by doing addition of 10 variables and dividing sum with 10.

TotalAge = age1 + age2 + age3 + age4 + age5 + age6 + age7 + age8 +age9 + age10 ;

AverageAge = TotalAge / 10;

**Solution:**

#include<iostream>

using namespace std;

int main(){  
  
**// declaring 10 integer variables to take input of students age**  
int age1,age2,age3, age4, age5, age6, age7, age8, age9, age10;  
  
**// declaring variables to calculate totalAge and averageAge**  
int TotalAge = 0, AverageAge = 0;

**// taking input of each student age**  
cout<<"please enter the age of student 1 ";

cin>>age1;

cout<<"please enter the age of student 2 ";

cin>>age2;

cout<<"please enter the age of student 3 ";

cin>>age3;

cout<<"please enter the age of student 4 ";

cin>>age4;

cout<<"please enter the age of student 5 ";

cin>>age5;

cout<<"please enter the age of student 6 ";

cin>>age6;

cout<<"please enter the age of student 7 ";

cin>>age7;

cout<<"please enter the age of student 8 ";

cin>>age8;

cout<<"please enter the age of student 9 ";

cin>>age9;

cout<<"please enter the age of student 10 ";

cin>>age10;

**//calculate totalAge by adding age of all students**  
TotalAge = age1 + age2 + age3 + age4 + age5 + age6 + age7 + age8 + age9 + age10;   
 **// calculate AverageAge by dividing the totalAge with number of students**AverageAge = TotalAge/10;

**//Display the result (average age)**

cout<<"Average of students is: "<<AverageAge;

}

**Alternative Solution**

#include<iostream>

using namespace std;

main() {  
  **// declaring variable age to take input** int age=0;  
   
 **// declaring variables to calculate totalAge and averageAge**  
 int TotalAge = 0, AverageAge = 0;

**// using for loop to take input of each students and adding them**  
 for (int i = 1;i<=10;i++){

cout<<"please enter the age of student "<<i<<" :\t";

cin>>age;

TotalAge += age;

}

**// calculate AverageAge by dividing the totalAge with number of students**  
 AverageAge = TotalAge/10;

**//Display the result (average age)**

cout<<"Average of students is: "<<AverageAge;

}