1. Write a C++ program to create a class called MATRIX using a two-dimensional array of integers. Implement the following operations by overloading the operator = = which checks the compatibility of two matrices m1 and m2 to be added and subtracted. Perform the addition and subtraction by overloading the operators + and – respectively. Display the results (sum matrix m3 and difference matrix m4).

#include <iostream>

using namespace std;

class matrix

{

private:

int m,n;

int \*\*a;

public:

matrix();

matrix(int m\_, int n\_);//constructor for nameless temporyary object, here both the arguments should be passed or else the normal constructor will be called and asked to input values m, n from user, that's y i didn't put default arguments here\*\*\*\*\*\*\*\*\*

~matrix();

void read();

bool operator==(const matrix&);

matrix operator+(const matrix&);

matrix operator-(const matrix&);

void display();

};

matrix::matrix()

{

cout << "Enter the order of a matrix m x n : " ;

cin >> m >> n;

a = (int\*\*)calloc(m, sizeof(int\*));

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

a[i] = (int\*)calloc(n, sizeof(int));

}

matrix::matrix(int m\_, int n\_) : m(m\_), n(n\_)

{

a = (int\*\*)calloc(m, sizeof(int\*));

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

a[i] = (int\*)calloc(n, sizeof(int));

}

matrix::~matrix()

{

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

free(a[i]);

free(a);

}

void matrix::read()

{

cout<<"Enter the elements of a matrix of order "<<m<<" x "<<n<<" : "<<endl;

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

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

cin >> a[i][j];

}

bool matrix::operator==(const matrix &m2)

{

if(m!=m2.m || n!=m2.n)

{

cout << "Order of matrix 1 is not equal to order of matrix 2 " <<endl;

exit(0);

}

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

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

if(a[i][j]!=m2.a[i][j])

return false;

return true;

}

matrix matrix::operator+(const matrix &m2)

{

matrix m3(m,n);

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

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

m3.a[i][j] = a[i][j] + m2.a[i][j];

return m3;

}

matrix matrix::operator-(const matrix &m2)

{

matrix m3(m,n);

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

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

m3.a[i][j] = a[i][j] - m2.a[i][j];

return m3;

}

void matrix::display()

{

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

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

{

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

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

cout << endl;

}

cout << endl;

}

int main()

{

cout << "Matrix 1 : " << endl;

matrix m1;

m1.read();

m1.display();

cout << "Matrix 2 : " << endl;

matrix m2;

m2.read();

m2.display();

if(m1==m2)

cout << "Matrix 1 is equal to Matrix 2" << endl << endl;

else

cout << "Matrix 1 is not equal to Matrix 2" << endl << endl;

matrix m3 = m1 + m2;

cout << "Matrix 1 + Matrix 2 = " << endl;

m3.display();

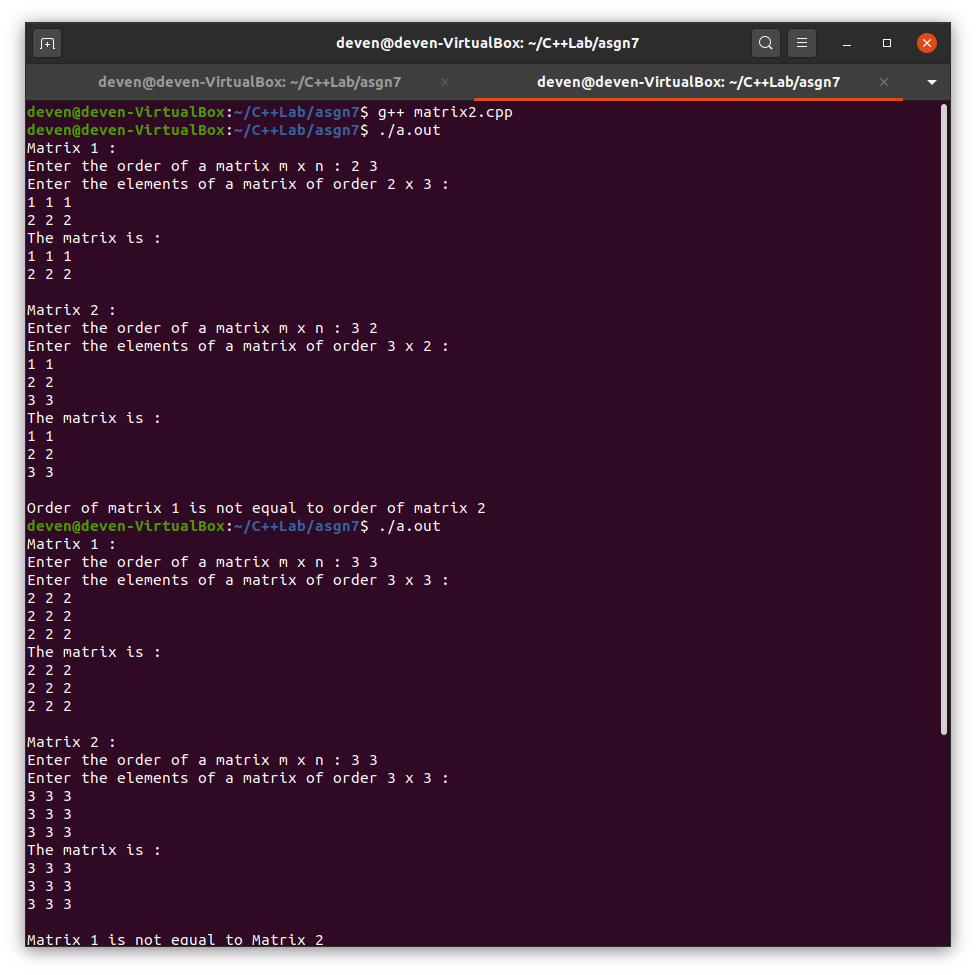
matrix m4 = m1 - m2;

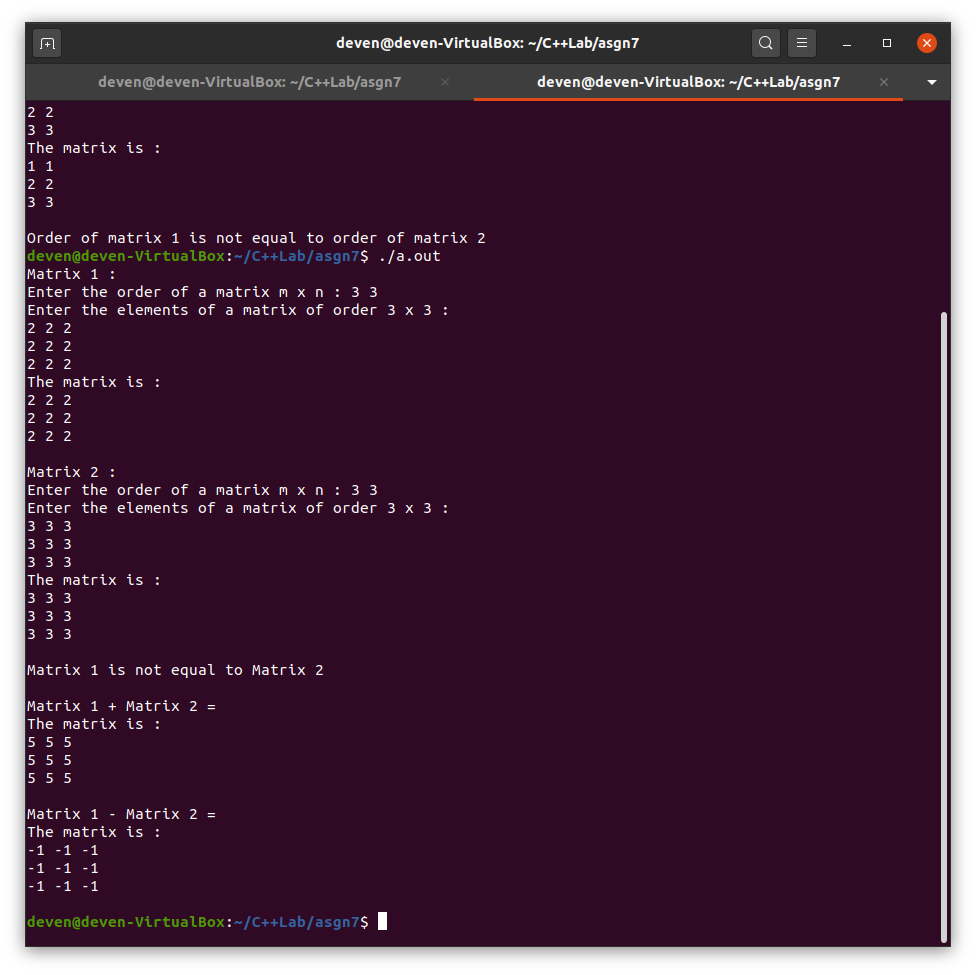
cout << "Matrix 1 - Matrix 2 = " << endl;

m4.display();

return 0;

}





2. Consider a class Date with year, month and day. Perform prefix and postfix increment operators for the Date suitably to show the new values.

#include <iostream>

using namespace std;

class Date

{

private :

int day, month, year;

public :

Date(int d = 0, int m = 0, int y = 0) : day(d), month(m), year(y) {}

void read();

Date operator++();

Date operator++(int);

void display();

};

void Date::read()

{

cout << "Enter the day, month and year : ";

cin >> day >> month >> year;

}

Date Date::operator++()

{

++day;

month += (day/30);

year += (month/12);

month %= 12;

day %= 30;

//return Date(day, month, year);

return \*this;

}

Date Date::operator++(int)

{

int d = day++, m = month, y = year;

month += (day/30);

year += (month/12);

month %= 12;

day %= 30;

return Date(d, m, y);

}

void Date::display()

{

cout << endl;

cout << "Day = " << day << endl;

cout << "Month = " << month << endl;

cout << "Year = " << year << endl;

cout << endl;

}

int main()

{

Date d;

d.read();

d.display();

Date e = ++d;

cout << "After prefix operation : " << endl;

d.display();

cout << "Returned object : " << endl;

e.display();

e = d++;

cout << "After postfix operation : " << endl;

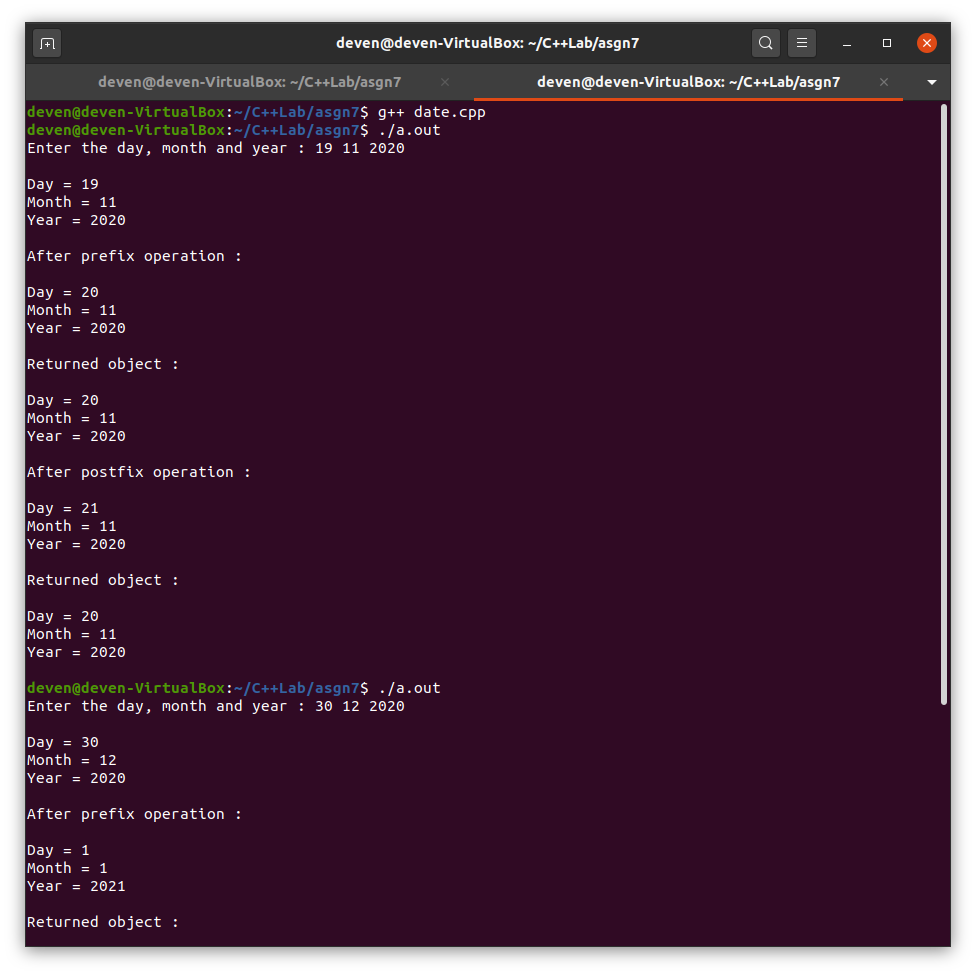
d.display();

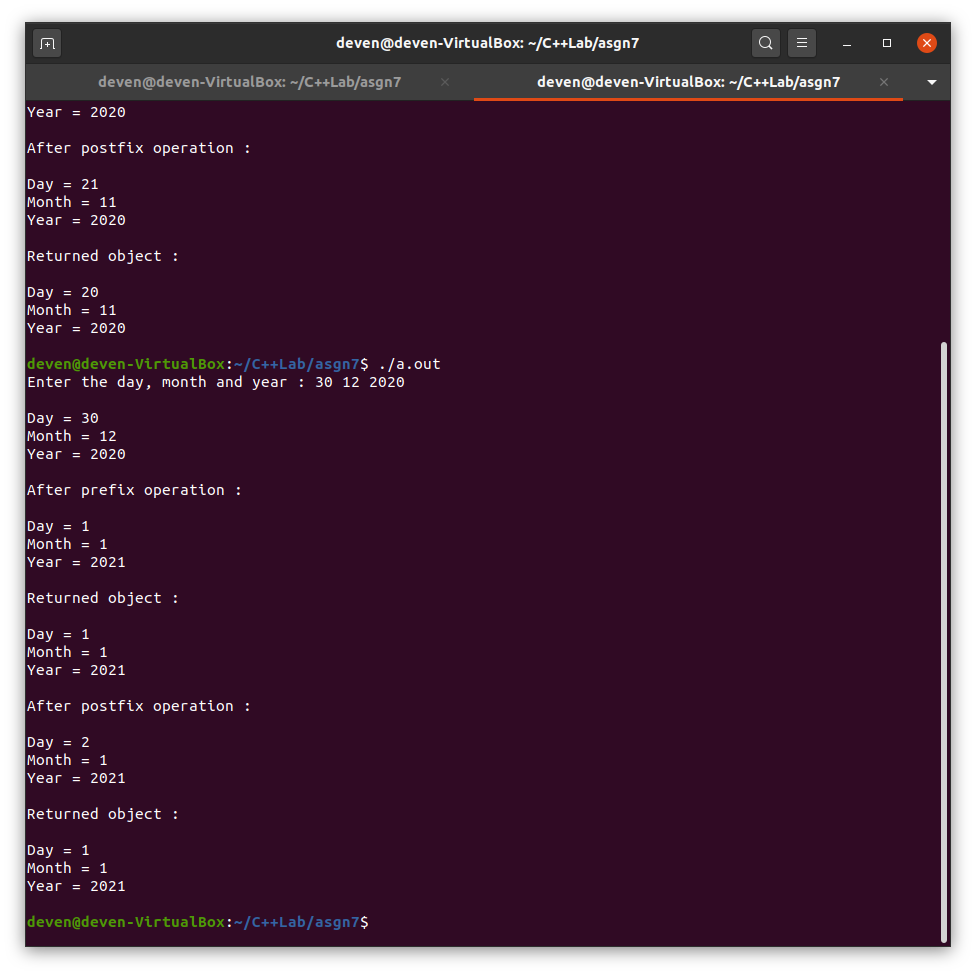
cout << "Returned object : " << endl;

e.display();

return 0;

}





3. Consider a class Student with marks. Compare marks of two students using comparison operators.

#include <iostream>

using namespace std;

class student

{

private:

int id;

string name;

int marks[4];

public:

void read();

void display();

bool operator==(student);

bool operator>(student);

};

void student::read()

{

cout << "Enter the student's id and name : " << endl;

cin >> id >> name;

cout << "Enter the 4 courses marks : " << endl;

for(int i = 0; i < 4; ++i)

cin >> marks[i];

}

void student::display()

{

cout << "Student details : " << endl;

cout << "Id : " << id << endl;

cout << "Name : " << name << endl;

cout << "Marks in the 4 courses : " << endl;

for(int i = 0; i < 4; ++i)

cout << marks[i] <<" ";

cout << endl;

}

bool student::operator==(student s2)

{

int sum1 = 0, sum2 = 0;

for(int i = 0; i < 4; ++i)

{

sum1 += marks[i];

sum2 += s2.marks[i];

}

if(sum1==sum2)

return true;

return false;

}

bool student::operator>(student s2)

{

int sum1 = 0, sum2 = 0;

for(int i = 0; i < 4; ++i)

{

sum1 += marks[i];

sum2 += s2.marks[i];

}

if(sum1>sum2)

return true;

return false;

}

int main()

{

student s1, s2;

cout << "Student 1 :" << endl;

s1.read();

s1.display();

cout << "Student 2 :" << endl;

s2.read();

s2.display();

if(s1 == s2)

cout << "Total marks of student 1 is equal to total marks student 2" << endl;

else if(s1 > s2)

cout << "Total marks of student 1 is greater than total marks student 2" << endl;

else

cout << "Total marks of student 2 is greater than total marks student 1" << endl;

return 0;

}

