

1. Write a C++ program to find the number of vowel characters in the given character array using pointer arithmetic

```
#include<iostream>
```

```
int noOfVowels(char *str) {
```

```
    int count = 0;
```

```
    for (int i = 0; *(str + i) != '\0'; i++) {
```

```
        if (*(str + i) == 'a' || *(str + i) == 'A' ||
```

```
            *(str + i) == 'e' || *(str + i) == 'E' ||
```

```
            *(str + i) == 'i' || *(str + i) == 'I' ||
```

```
            *(str + i) == 'o' || *(str + i) == 'O' ||
```

```
            *(str + i) == 'u' || *(str + i) == 'U' )
```

```
            ++count;
```

```
    }
```

```
    return count;
```

```
}
```

```
int main() {
```

```
    char str[100];
```

```
    std::cout << "Enter a string" << std::endl;
```

```
    std::cin.getline(str, 100);
```

```
    std::cout << "No of vowels are " << noOfVowels(str) << std::endl;
```

```
}
```

```
× - □ Terminal File Edit View Search Terminal Help
abdul@abdul-pc:~/Local Disk D/oops lab$ g++ q1.cpp
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out
Enter a string
This is a string
No of vowels are 4
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```

2. Write a program in c++ to print a number in reverse order. Use functions with return type and without return type.

```
#include<iostream>
```

```
int reverseNo (int n) {
```

```
    int reverse = 0;
```

```
    while(n > 0) {
```

```
        int last = n % 10;
```

```
        reverse *= 10;
```

```
        reverse += last;
```

```
        n /= 10;
```

```
    }
```

```
    return reverse;
```

```
}
```

```
void reverseNowRT(int n) {
```

```
    std::cout << "\nFunction without return type called" << std::endl;
```

```
    int reverse = 0;
```

```
    while(n > 0) {
```

```
        int last = n % 10;
```

```
        reverse *= 10;
```

```
        reverse += last;
```

```
        n /= 10;
```

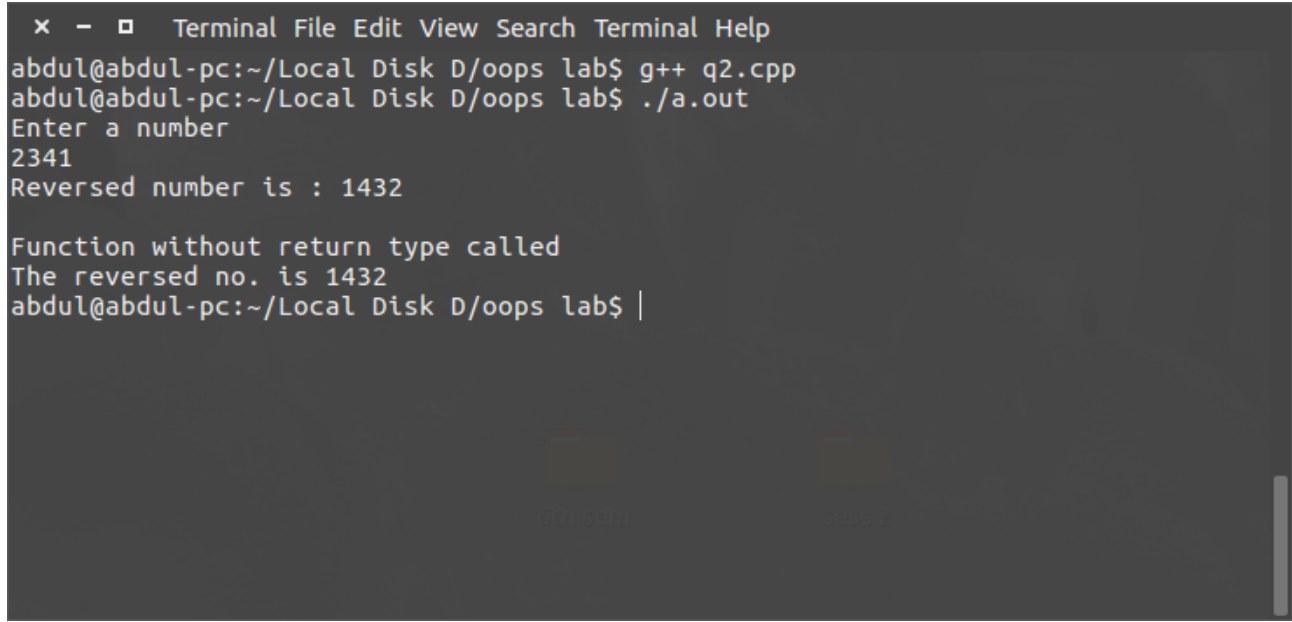
```
    }
```

```
    std::cout<< "The reversed no. is " << reverse << std::endl;
```

```
}
```

```
int main() {
```

```
int n;  
std::cout << "Enter a number" << std::endl;  
std::cin >> n;  
std::cout << "Reversed number is : " << reverseNo(n) << std::endl;  
reverseNoWRT(n);  
}
```



```
x - □ Terminal File Edit View Search Terminal Help  
abdul@abdu1-pc:~/Local Disk D/oops lab$ g++ q2.cpp  
abdul@abdu1-pc:~/Local Disk D/oops lab$ ./a.out  
Enter a number  
2341  
Reversed number is : 1432  
  
Function without return type called  
The reversed no. is 1432  
abdul@abdu1-pc:~/Local Disk D/oops lab$ |
```

3. Write a C++ program to perform various arithmetic operations such as addition, subtraction, multiplication, modulus and division.

```
#include<stdio.h>
#include<iostream>

inline int add(int a, int b) {
    return a + b;
}

inline int sub(int a, int b) {
    return a - b;
}

inline int mult(int a, int b) {
    return a * b;
}

inline double div(int a, int b) {
    return (double) a / b;
}

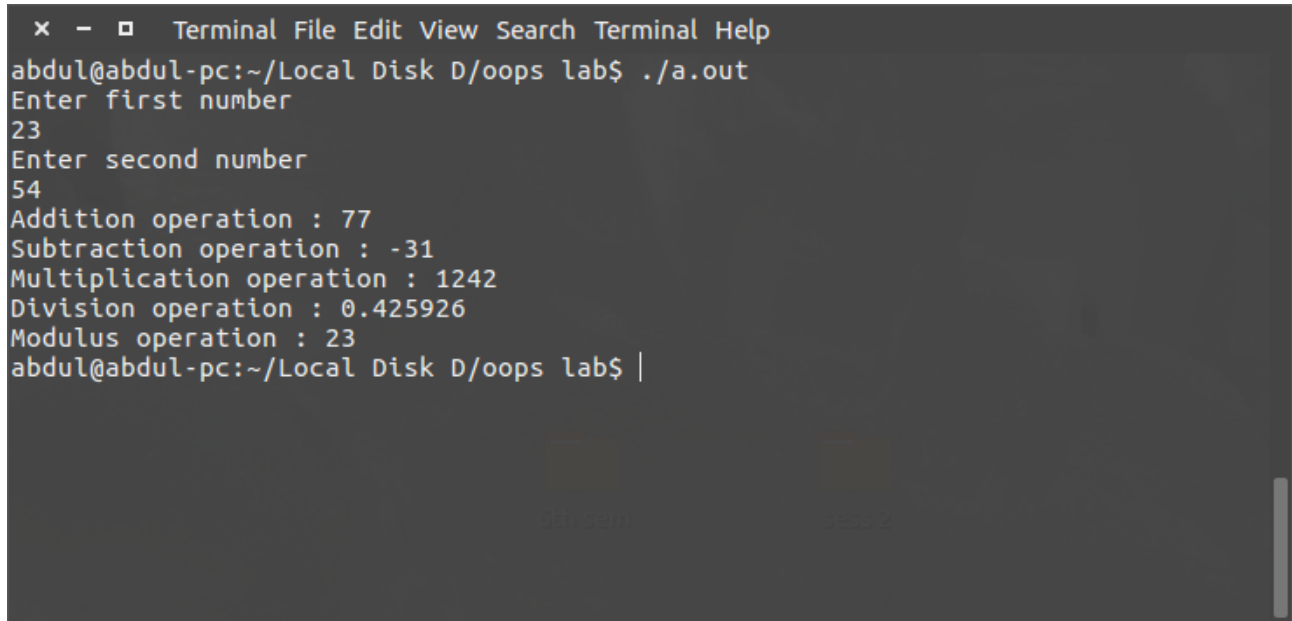
inline int mod(int a, int b) {
    return a % b;
}

int main() {

    int x, y;
    std::cout << "Enter first number" << std::endl;
    std::cin >> x;
    std::cout << "Enter second number" << std::endl;
    std::cin >> y;
```

```
std::cout << "Addition operation : " << add(x, y) << std::endl;
std::cout << "Subtraction operation : " << sub(x, y) << std::endl;
std::cout << "Multiplication operation : " << mult(x, y) << std::endl;
std::cout << "Division operation : " << div(x, y) << std::endl;
std::cout << "Modulus operation : " << mod(x, y) << std::endl;

return 0;
}
```



```
x - □ Terminal File Edit View Search Terminal Help
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out
Enter first number
23
Enter second number
54
Addition operation : 77
Subtraction operation : -31
Multiplication operation : 1242
Division operation : 0.425926
Modulus operation : 23
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```

4. Create a class for counting the objects created and destroyed within various block using constructors and destructors.

```
#include<stdio.h>
#include<iostream>

class Example {

public:
    static int createdObjects;
    static int destroyedObjects;
    static int currentObjects;

    Example () {
        ++createdObjects;
        ++currentObjects;
    }

    ~Example(){
        ++destroyedObjects;
        --currentObjects;
    }

};

int Example::createdObjects = 0;
int Example::destroyedObjects = 0;
int Example::currentObjects = 0;

int main() {

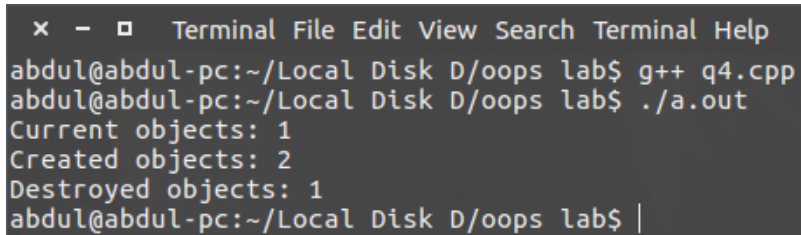
    Example ex;
```

```
{  
    Example ex1;  
}
```

```
std::cout << "Current objects: " << ex.currentObjects << std::endl;  
std::cout << "Created objects: " << ex.createdObjects << std::endl;  
std::cout << "Destroyed objects: " << ex.currentObjects << std::endl;
```

```
return 0;
```

```
}
```



```
x - □ Terminal File Edit View Search Terminal Help  
abdul@abdul-pc:~/Local Disk D/oops lab$ g++ q4.cpp  
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out  
Current objects: 1  
Created objects: 2  
Destroyed objects: 1  
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```


5. Write a C++ program to create three objects for a class named `pntr_obj` with data members such as `roll_no` & `name`. Create a member function `set_data()` for setting the data values and `print()` member function to print which object has invoked it using this pointer.

```
#include<iostream>
#include<string.h>

class pntr_obj {

    int roll_no;
    char name[20];
public:
    void set_data (int r, const char *n) {
        roll_no = r;
        strncpy(name, n, sizeof(name)-1);
        name[sizeof(name)-1] = '\0';
    }

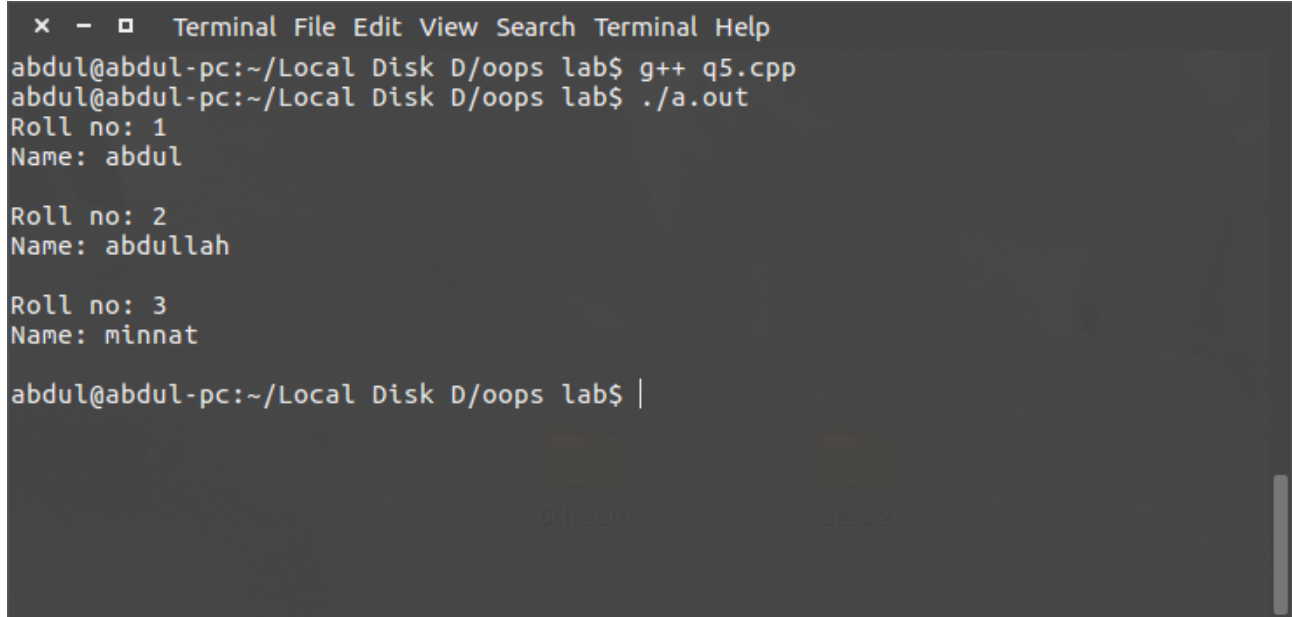
    void print() {
        std::cout << "Roll no: " << this->roll_no << std::endl;
        std::cout << "Name: " << this->name << std::endl;
    }

};

int main() {

    pntr_obj p1;
    p1.set_data(1,"abdul");
    p1.print();
    std::cout << "\n";
```

```
pntr_obj p2;  
p2.set_data(2,"abdullah");  
p2.print();  
std::cout << "\n";  
  
pntr_obj p3;  
p3.set_data(3,"minnat");  
p3.print();  
std::cout << "\n";  
  
return 0;  
}
```



```
x - □ Terminal File Edit View Search Terminal Help  
abdul@abdul-pc:~/Local Disk D/oops lab$ g++ q5.cpp  
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out  
Roll no: 1  
Name: abdul  
  
Roll no: 2  
Name: abdullah  
  
Roll no: 3  
Name: minnat  
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```

6. Write a C++ program to implement virtual function (Polymorphism) by creating a base class c_polygon which has virtual function area(). Two classes c_rectangle and c_triangle derived from c_polygon and they have area() to calculate and return the area of rectangle and triangle respectively.

```
#include<stdio.h>
#include<iostream>

class c_polygon {

public:
    virtual double area() {
        std::cout << "Not implemented";
        return 0;
    }
};

class c_rectangle: public c_polygon {

    double length;
    double breadth;

public:
    c_rectangle(double l, double b) {
        length = l;
        breadth = b;
    }

    double area() {
        return length * breadth;
    }
}
```

```
};
```

```
class c_triangle: public c_polygon {
```

```
    double base;
```

```
    double height;
```

```
public:
```

```
    c_triangle(double b, double h) {
```

```
        base = b;
```

```
        height = h;
```

```
    }
```

```
    double area(){
```

```
        return 0.5 * base * height;
```

```
    }
```

```
};
```

```
int main() {
```

```
    c_polygon *pol = new c_rectangle(2,5);
```

```
    std::cout << "Area of rectangle is: " << pol->area() << std::endl;
```

```
    pol = new c_triangle(3,7);
```

```
    std::cout << "Area of traigle is: " << pol->area() << std::endl;
```

```
    return 0;
```

```
}
```

```
× - □ Terminal File Edit View Search Terminal Help
abdul@abdul-pc:~/Local Disk D/oops lab$ g++ q6.cpp
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out
Area of rectangle is: 10
Area of traigle is: 10.5
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```

7. Write a C++ program to count the number of persons inside a bank, by increasing count whenever a person enters a bank, using an increment (++) operator overloading function, and decrease the count whenever a person leaves the bank, using a decrement (--) operator overloading function inside the class.

```
#include<iostream>
```

```
class PersonInsideBank {
```

```
    int personsInsideBank;
```

```
public:
```

```
    PersonInsideBank() {  
        personsInsideBank = 0;  
    }
```

```
    PersonInsideBank(int n) {  
        personsInsideBank = n;  
    }
```

```
    void operator ++ () {  
        ++personsInsideBank;  
    }
```

```
    void operator -- () {  
        --personsInsideBank;  
    }
```

```
    int status() {  
        return personsInsideBank;  
    }
```

```
};
```

```
int main() {
```

```
    PersonInsideBank pib(0);
```

```
    ++pib;
```

```
    ++pib;
```

```
    ++pib;
```

```
    std::cout << "Perons inside bank are: " << pib.status() << std::endl;
```

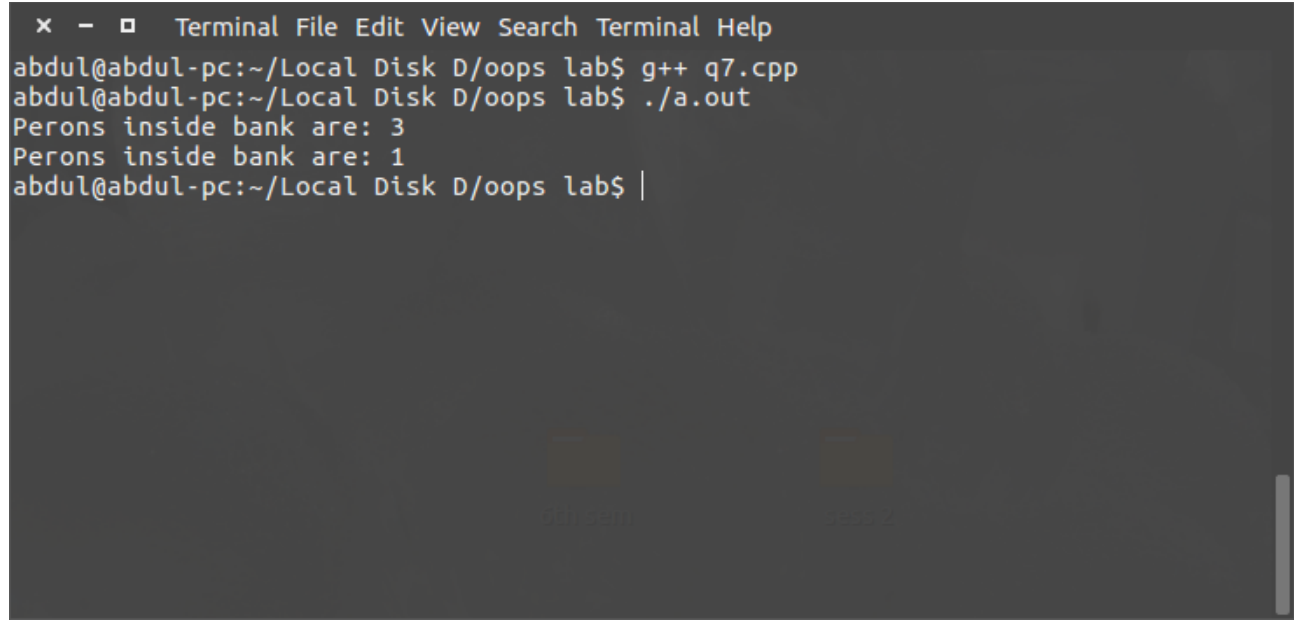
```
    --pib;
```

```
    --pib;
```

```
    std::cout << "Perons inside bank are: " << pib.status() << std::endl;
```

```
    return 0;
```

```
}
```



```
x - □ Terminal File Edit View Search Terminal Help
abdul@abdul-pc:~/Local Disk D/oops lab$ g++ q7.cpp
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out
Perons inside bank are: 3
Perons inside bank are: 1
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```

8. Write a C++ program to create two objects of a class called company and add their data members using an operator overloaded function for '+' operator and '-' operator.

```
#include<iostream>
```

```
class Company {
```

```
    int employees;
```

```
public:
```

```
    Company() {
```

```
        employees = 0;
```

```
    }
```

```
    Company(int n) {
```

```
        employees = n;
```

```
    }
```

```
    Company operator + (Company &c) {
```

```
        int n = this-> employees + c.noOfEmployees();
```

```
        return Company(n);
```

```
    }
```

```
    Company operator - (Company &c) {
```

```
        int n = this-> employees - c.noOfEmployees();
```

```
        return Company(n);
```

```
    }
```

```
    int noOfEmployees() {
```

```
        return employees;
```

```
    }
```



```
};
```

```
int main() {
```

```
    Company c1(7), c2(3);
```

```
    Company c3 = c1 + c2;
```

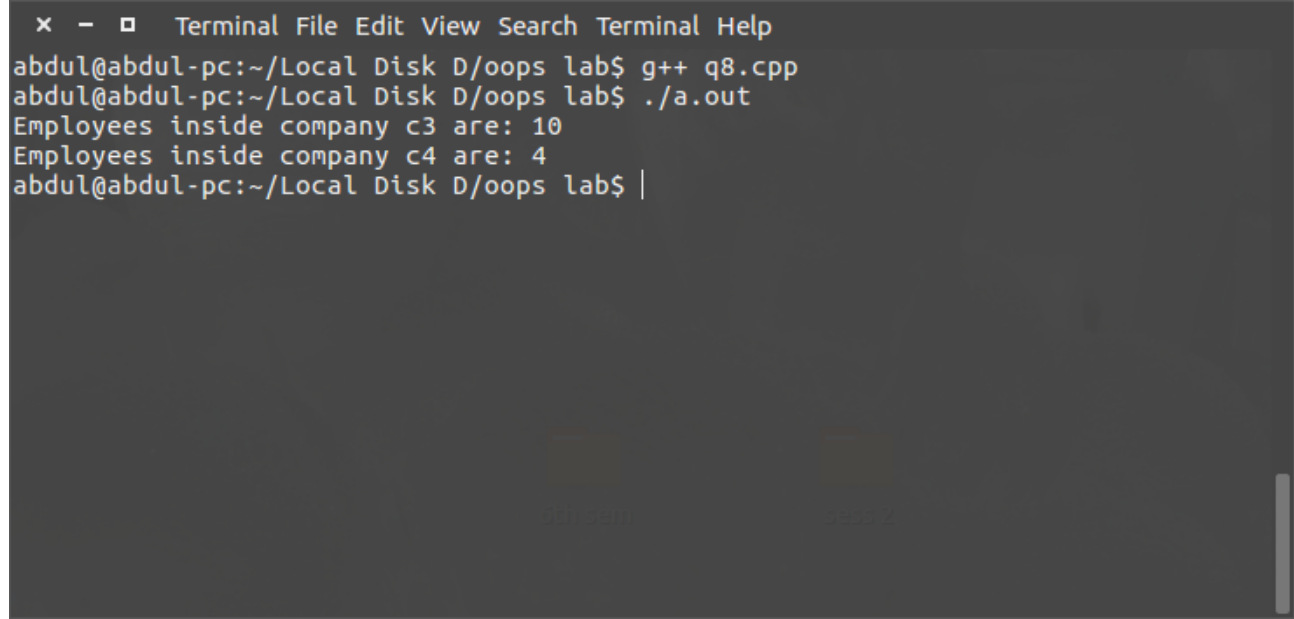
```
    std::cout << "Employees inside company c3 are: " << c3.noOfEmployees()  
<< std::endl;
```

```
    Company c4 = c1 - c2;
```

```
    std::cout << "Employees inside company c4 are: " << c4.noOfEmployees()  
<< std::endl;
```

```
    return 0;
```

```
}
```



```
x - □ Terminal File Edit View Search Terminal Help  
abdul@abdul-pc:~/Local Disk D/oops lab$ g++ q8.cpp  
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out  
Employees inside company c3 are: 10  
Employees inside company c4 are: 4  
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```

9. Program to create a class matrix and overload +,- operators to perform matrix addition and subtraction.

```
#include<stdio.h>
#include<iostream>
#include<stdexcept>

class matrix
{
    int row, col;
    int **mat;

    void assignMatrix(){

        mat = new int *[col];

        for (int j = 0; j < col; ++j) {
            mat[j] = new int [row];
        }
    }

    void checkForOrder(matrix &m) {
        if (this->row != m.getRow() || this->col != m.getCol())
            throw new std::domain_error("Order does not match of both
matrices");
    }

public:
    matrix(int n) {
        row = n;
        col = n;
        assignMatrix();
    }
```

```

matrix(int r, int c) {
    row = r;
    col = c;
    assignMatrix();
}

int getRow() {
    return row;
}

int getCol() {
    return col;
}

int **getMatrix() {
    return mat;
}

matrix operator +(matrix &m) {

    checkForOrder(m);

    matrix m1(m.getRow(), m.getCol());

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

        for (int j = 0; j < col; ++j) {
            m1.getMatrix()[i][j] = this->mat[i][j] + m.getMatrix()[i]
[j];
        }
    }

    return m1;
}

```

```
}
```

```
matrix operator -(matrix &m) {
```

```
    checkForOrder(m);
```

```
    matrix m1(m.getRow(), m.getCol());
```

```
    for (int i = 0; i < row; ++i) {
```

```
        for (int j = 0; j < col; ++j) {
```

```
            m1.getMatrix()[i][j] = this->mat[i][j] - m.getMatrix()[i]  
[j];
```

```
        }
```

```
    }
```

```
    return m1;
```

```
}
```

```
void input(std::istream &in) {
```

```
    for (int i = 0; i < row; ++i) {
```

```
        for (int j = 0; j < col; ++j) {
```

```
            in >> mat[i][j];
```

```
        }
```

```
    }
```

```
}
```

```
void output(std::ostream &out) {
```

```
    for (int i = 0; i < row; ++i) {
```

```
        out << std::endl;
```

```

        for (int j = 0; j < col; ++j) {
            out << mat[i][j] << " ";
        }
    }
}

};

int main() {

    matrix m1(2), m2(2);

    std::cout << "Enter first matrix" << std::endl;
    m1.input(std::cin);
    std::cout << "Enter second matrix" << std::endl;
    m2.input(std::cin);

    matrix m3 = m1 + m2;
    std::cout << "Their sum is: " << std::endl;
    m3.output(std::cout);

    matrix m4 = m1 - m2;
    std::cout << "\n\nTheir difference is: " << std::endl;
    m4.output(std::cout);

}

```

× - □ Terminal File Edit View Search Terminal Help

Enter first matrix

1

2

3

4

Enter second matrix

5

6

7

8

Their sum is:

6 8

10 12

Their difference is:

-4 -4

-4 -4 abdul@abdul-pc:~/Local Disk D/oops lab\$ |

10. Program to accept five different numbers by creating a class called friendfunc1 and friendfunc2 taking 2 and 3 arg respectively and calculate the average of these numbers by passing object of class to friend function.

```
#include<iostream>
```

```
class friendfunc2;
```

```
class friendfunc1 {
```

```
    int a, b;
```

```
public:
```

```
    friendfunc1(int x, int y) {
```

```
        a = x;
```

```
        b = y;
```

```
    }
```

```
    friend double avg(friendfunc1 &f1, friendfunc2 &f2);
```

```
};
```

```
class friendfunc2 {
```

```
    int c, d, e;
```

```
public:
```

```
    friendfunc2(int x, int y, int z) {
```

```
        c = x;
```

```
        d = y;
```

```
        e = z;
```

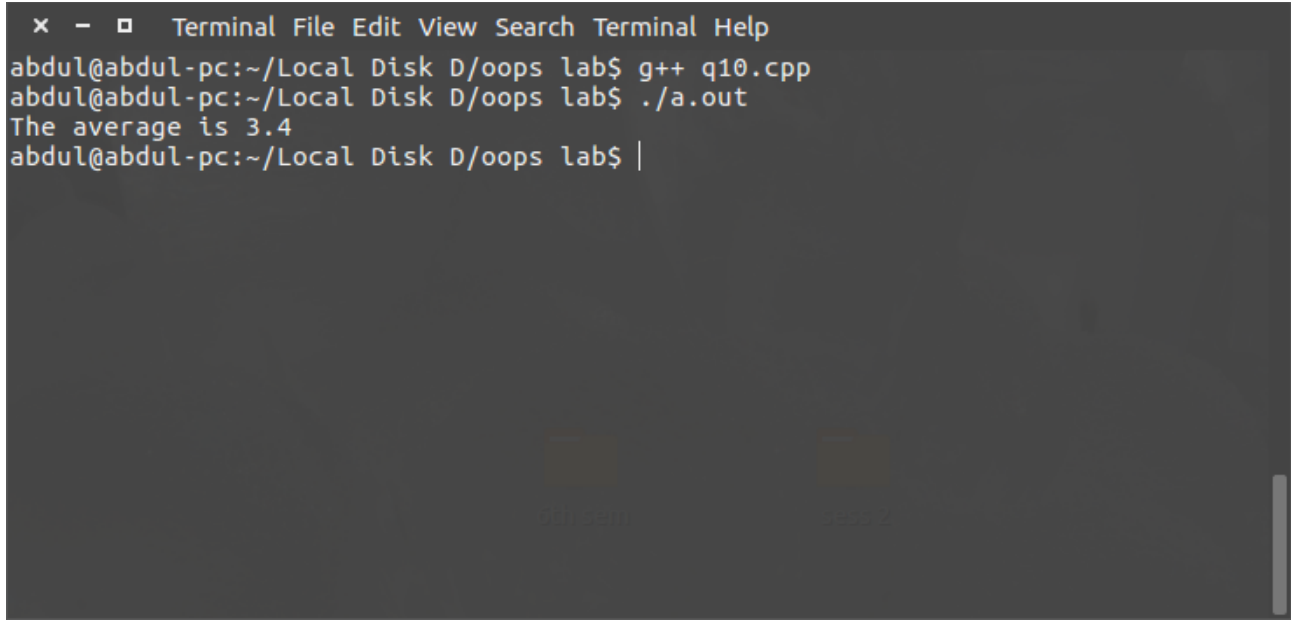
```
    }
```

```
    friend double avg(friendfunc1 &f1, friendfunc2 &f2);
```

```
};
```

```
double avg(friendfunc1 &f1, friendfunc2 &f2) {  
  
    return (double)(f1.a + f1.b + f2.c + f2.d + f2.e) / 5;  
}
```

```
int main() {  
  
    friendfunc1 f1(1, 2);  
    friendfunc2 f2(3, 4, 7);  
  
    std::cout << "The average is " << avg(f1, f2) << std::endl;  
    return 0;  
}
```



The screenshot shows a terminal window with a dark background and light-colored text. The window title is "Terminal File Edit View Search Terminal Help". The user is in a directory "~/Local Disk D/oops lab". The commands entered are `g++ q10.cpp` and `./a.out`. The output of the program is "The average is 3.4". The prompt `abdu1@abdu1-pc:~/Local Disk D/oops lab$` is visible at the end of the line.

```
x - □ Terminal File Edit View Search Terminal Help  
abdu1@abdu1-pc:~/Local Disk D/oops lab$ g++ q10.cpp  
abdu1@abdu1-pc:~/Local Disk D/oops lab$ ./a.out  
The average is 3.4  
abdu1@abdu1-pc:~/Local Disk D/oops lab$ |
```


12. Write a program that uses a function template called min to determine the smaller of two arguments. The program should work for integers, characters and floating point numbers as arguments to this function.

```
#include<iostream>
```

```
template <class T>
```

```
T min(T a, T b) {  
    return a < b ? a : b;  
}
```

```
int main() {
```

```
    std::cout << "Minimum of 3 and 6 is : " << min(3, 6) << std::endl;
```

```
    std::cout << "Minimum of 'a' and 's' is : " << min('a', 's') <<  
std::endl;
```

```
    std::cout << "Minimum of 1.73 and 3.14 is : " << min(1.73, 3.14) <<  
std::endl;
```

```
    return 0;
```

```
}
```

× - □ Terminal File Edit View Search Terminal Help

abdul@abdul-pc:~/Local Disk D/oops lab\$./a.out

Minimum of 3 and 6 is : 3

Minimum of 'a' and 's' is : a

Minimum of 1.73 and 3.14 is : 1.73

abdul@abdul-pc:~/Local Disk D/oops lab\$ |

13. Write a program to explain class template by creating a template T for a class name pair having two data members of type t which are inputted by a constructor and a member function get_max() return the greatest of two number to main.

```
#include<iostream>
```

```
template <class T>
```

```
class pair {
```

```
    T a, b;
```

```
public:
```

```
    pair(T x, T y) {
```

```
        a = x;
```

```
        b = y;
```

```
    }
```

```
    T get_max(){
```

```
        return a > b ? a : b;
```

```
    }
```

```
};
```

```
int main() {
```

```
    pair<int> p1(2,5);
```

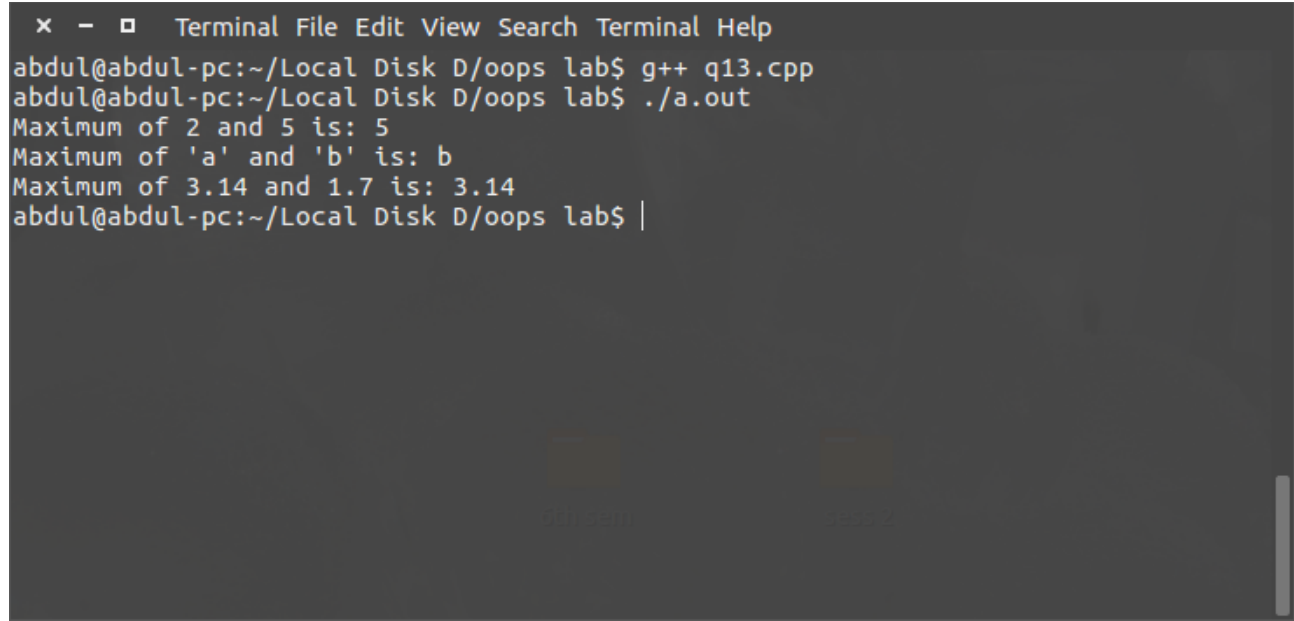
```
    std::cout << "Maximum of 2 and 5 is: " << p1.get_max() << std::endl;
```

```
    pair<char> p2('a','b');
```

```
    std::cout << "Maximum of 'a' and 'b' is: " << p2.get_max() << std::endl;
```

```
    pair<float> p3(3.14, 1.7);
```

```
std::cout << "Maximum of 3.14 and 1.7 is: " << p3.get_max() <<
std::endl;
}
```



A screenshot of a terminal window with a dark background and light-colored text. The window has a title bar with icons for close, maximize, and a menu, followed by the text 'Terminal File Edit View Search Terminal Help'. The terminal shows the following commands and output:

```
abdul@abdul-pc:~/Local Disk D/oops lab$ g++ q13.cpp
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out
Maximum of 2 and 5 is: 5
Maximum of 'a' and 'b' is: b
Maximum of 3.14 and 1.7 is: 3.14
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```

At the bottom of the terminal, there are two small, faint icons: a square icon labeled '0th 3200' and a circular icon labeled '3200 2'.

14. write a program in c++ to overload cin and cout stream Operators to input and display objects of a class student which contains student details like name,class,roll_no etc.

```
#include<iostream>
```

```
class student {
```

```
    int roll;
```

```
    char name[20];
```

```
friend std::istream& operator>> (std::istream &in, student &s) {  
    std::cout << "Enter the roll number of student" << std::endl;  
    in >> s.roll;  
    std::cout << "Enter the name of student" << std::endl;  
    in >> s.name;  
    return in;  
}
```

```
friend std::ostream& operator<< (std::ostream &out, student &s) {  
    std::cout << "The roll number of student: ";  
    out << s.roll;  
    std::cout << "\nThe name of student: ";  
    out << s.name << std::endl;  
    return out;  
}
```

```
};
```

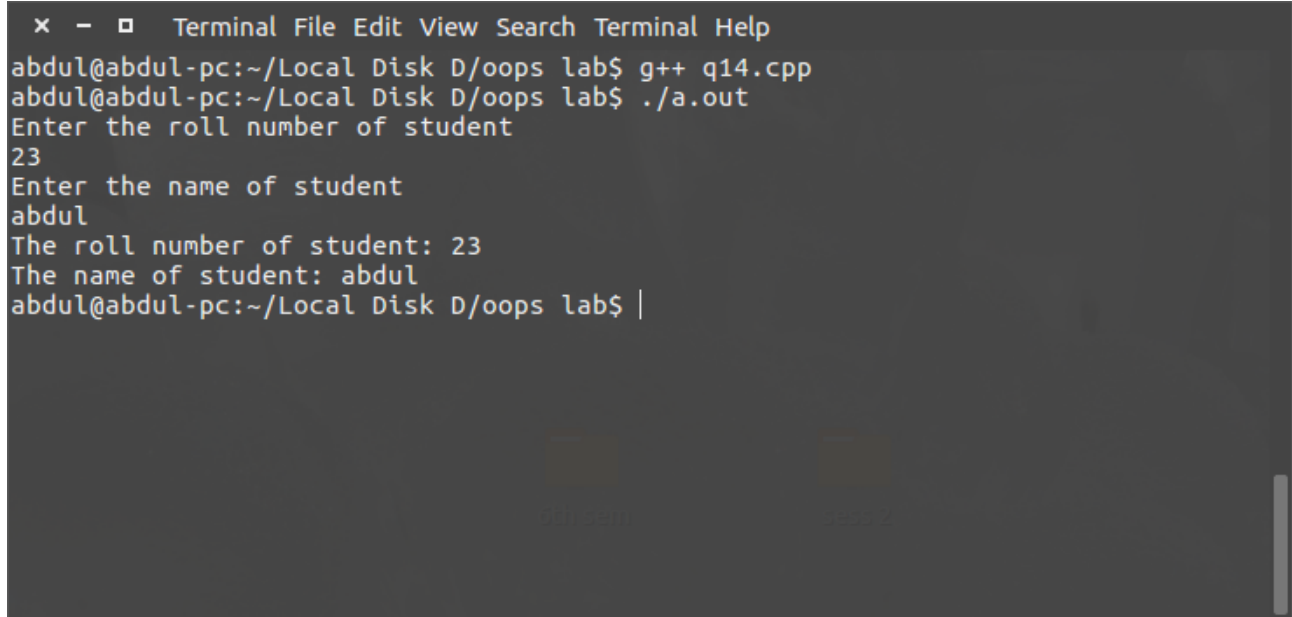
```
int main() {
```

```
student s;
```

```
std::cin >> s;
```

```
std::cout << s;
```

```
}
```



A screenshot of a terminal window with a dark background and light-colored text. The window has a title bar with 'Terminal' and several icons. The command prompt shows the user 'abdul' at 'abdul-pc' in the directory '~/Local Disk D/oops lab'. The user runs 'g++ q14.cpp' and then './a.out'. The program prompts for a roll number, which is '23', and then for a name, which is 'abdul'. The program then outputs 'The roll number of student: 23' and 'The name of student: abdul'. The prompt returns to the user.

```
abdul@abdul-pc:~/Local Disk D/oops lab$ g++ q14.cpp
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out
Enter the roll number of student
23
Enter the name of student
abdul
The roll number of student: 23
The name of student: abdul
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```

15. Write a class 3D which inherits another class 2D ,for 3D calculation using these classes define any two points in 3D space and find the distance between them. The formula for finding distance between any two points in 3D space is :

$$\text{distance} = \text{squareroot}(x_d^2 + y_d^2 + z_d^2);$$

where $x_d = x_2 - x_1$, $y_d = y_2 - y_1$ and $z_d = z_2 - z_1$.

```
#include<iostream>
```

```
#include<math.h>
```

```
class TwoD {
```

```
protected:
```

```
    double x, y;
```

```
public:
```

```
    TwoD(int a, int b) {
```

```
        x = a;
```

```
        y = b;
```

```
    }
```

```
    double getX() {
```

```
        return x;
```

```
    }
```

```
    double getY() {
```

```
        return y;
```

```
    }
```

```
};
```

```
class ThreeD: public TwoD {
```

```
double z;
```

```
public:
```

```
    ThreeD(int a, int b, int c) : TwoD(a, b) {  
        z = c;  
    }
```

```
    double getZ() {  
        return z;  
    }
```

```
    double distanceFrom(ThreeD ob) {  
        double xd = ob.getX() - this->x;  
        double yd = ob.getY() - this->y;  
        double zd = ob.getZ() - this->z;  
  
        return sqrt(pow(xd, 2) + pow(yd, 2) + pow(zd, 2));  
    }
```

```
};
```

```
int main() {
```

```
    ThreeD obj1(1,2,3);
```

```
    ThreeD obj2(4,6,7);
```

```
    std::cout << "The distance between two points are: " <<  
obj1.distanceFrom(obj2) << std::endl;
```

```
}
```


× - □ Terminal File Edit View Search Terminal Help

```
abdul@abdul-pc:~/Local Disk D/oops lab$ g++ q15.cpp
abdul@abdul-pc:~/Local Disk D/oops lab$ ./a.out
The distance between two points are: 6.40312
abdul@abdul-pc:~/Local Disk D/oops lab$ |
```

16. Write a program in Java to implement a calculator having addition and subtraction operations along with the concept of memory i.e. to save the current value on the screen for later use.

```
import java.util.*;

public class q16 {

    public static void main(String[] args) {

        String str = "y";
        double mem = Double.MIN_VALUE;
        int choice = 0;
        Scanner sc = new Scanner(System.in);

        while(!str.equals("n")) {
            System.out.println("\n\nEnter the first number (or type MEM for
memory)");
            str = sc.next();
            double a = getNum(str, mem);

            System.out.println("Enter the second number (or type MEM for
memory)");
            str = sc.next();
            double b = getNum(str, mem);

            while (choice <= 0 || choice >2) {
                System.out.printf("Enter the operation to be
performed\n1.Add.\n2.Subtract");
                choice = sc.nextInt();
            }

            double ans = getResult(choice, a, b);
            System.out.println("The answer is " + ans );
            System.out.println("Do you want to remember it? (y or n)");
```

```

        str = sc.next();

        mem = remember(str, mem, ans);
        System.out.println("Do you want to continue? (y or n)");
        str = sc.next();
    }
}

public static double add (double a, double b) {
    return a + b;
}

public static double sub (double a, double b) {
    return a - b;
}

public static double remember (String str, double mem, double ans) {
    if(str.toLowerCase().equals("y") || str.toLowerCase().equals("yes"))
        return ans;
    else return mem;
}

public static double getResult(int choice, double a, double b) {
    if (choice == 1)
        return add(a, b);
    else if (choice == 2)
        return sub(a, b);
    else {
        throw new IllegalArgumentException("wrong choice");
    }
}

public static double getNum(String str, double mem) {

```

```
try {
    return Double.parseDouble(str);
} catch (IllegalArgumentException e) {
    if(mem == Double.MIN_VALUE) {
        System.out.println("Mem not initialised, will use 0 for the
mem.");
        return 0;
    } else return mem;
}
}
```

17. Write a program in C++ which acts as a contact manager. It saves content of a file, and retrieve them when asked by the user. Modules of the program include: Add, Delete, Search & Save.

```
#include <fstream>
#include <iostream>
#include <stdlib.h>

int search_contact() {

    std::string name, line;
    std::cout << "Enter the name to be searched: " << std::endl;
    std::cin >> name;
    std::ifstream file;
    file.open("abd.txt");

    while (getline(file, line)) {

        if (line == name) {

            std::cout << "Search successful!. The name is found." <<
std::endl;
            return 1;

        }

    }

    std::cout << "Search unsuccessful. Name not found .";
    return 0;

}

void add_contact() {
```

```

std::string name;
std::string mobile_no;
std::cout << "Enter name: " << std::endl;
std::cin >> name;
std::cout << "Enter mobile no: " << std::endl;
std::cin >> mobile_no;
std::ofstream file;
file.open("abd.txt",std::ios::app | std::ios::out);
file << name << std::endl << mobile_no << std::endl;
file.close();
}

void delete_contact()

{

std::string name,line;
std::cout << "Give the name to be deleted: " << std::endl;
std::cin >> name;
std::ifstream file;
std::ofstream temp;
file.open("abd.txt");
temp.open("temp.txt");

while (getline(file,line)) {

    if (line != name){
        temp << line << std::endl;
    } else {

std::cout << "The name was present and deleted successfully.";

```

```

        getline(file,line);
        getline(file,line);

    }
}

file.close();
temp.close();
remove("abd.txt");
rename("temp.txt","abd.txt");

}

int main()

{

    int choice;

    while (1) {

        std::cout << "\nMENU\n" << std::endl;

        std::cout << "\n1. Add \n2. Delete \n3. Search \n4. Exit"<<
std::endl;

        std::cout << "Enter your choice: " << std::endl;

        std::cin >> choice;

        switch(choice) {

            case 1: add_contact();
                    break;

```

```
    case 2: delete_contact();  
            break;  
  
    case 3: search_contact();  
            break;  
  
    case 4: exit(1);  
  
    default:  std::cout<< "Please Enter a correct choice."  
            break;  
  
    }  
    }  
}
```


Terminal File Edit View Search Terminal Help

abdul@abdul-pc:~/Local Disk D/oops lab\$./a.out

MENU

1. Add
2. Delete
3. Search
4. Exit

Enter your choice:

1

Enter name:

abdul

Enter mobile no:

880243453

MENU

1. Add
2. Delete
3. Search
4. Exit

Enter your choice:

3

Enter the name to be searched:

abdul

Search successful!. The name is found.

MENU

1. Add
2. Delete
3. Search
4. Exit

Enter your choice:

2

Give the name to be deleted:

abdul

The name was present and deleted successfully.

MENU

1. Add
2. Delete
3. Search
4. Exit

Enter your choice: