



Progressive Education Society's
Modern College of Engineering, Pune
MCA Department
A.Y.2022-23

(310907) Object Oriented Programming Laboratory

Class: FY-MCA

Shift / Div: A

Roll Number: 51043

Name: Vanessa Reetu Prashant More

Assignment No: 2

Date of Implementation: 21. 11. 22

Q1. Define a class to represent a bank account. Include the following members:

Data Members:

i) Name of the Depositor ii) Account number iii) Type of account iv) Balance amount in the account

Member Functions:

i) To assign initial values ii) To deposit an amount

iii) To withdraw an amount after checking the balance iv) To display name and balance

Program:

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

class account{
    char name[15];
    int acc_num;
    char type[1];
    float balance;

public:
    void getdata();
    void deposit();
    void withdraw();
    void display();
};

void account :: getdata(){
    cout<<"Enter name:";
    cin>>name;
    cout<<"Enter account number: ";
    cin>>acc_num;
    cout<<"Enter account type: ";
    cin>>type;
    cout<<"Enter balance: ";
    cin>>balance;
}

void account ::deposit(){
    float rs;
    cout<<"Enter amount to deposit: ";
    cin>>rs;
```

```

        balance += rs;
        cout<<"New balance = "<<balance;
    }

void account :: withdraw(){
    float amt;
    cout<<"Enter amount to withdraw: ";
    cin>>amt;

    if(amt>balance){
        cout<<"Insufficient amount in account!";
        cout<<"\nCurrent balance: "<<balance;
    }
    else {
        balance -= amt;
        cout<<"New balance = "<<balance;
    }
}

void account :: display(){
    cout<<"\nAccount Holder's Name: "<<name;
    cout<<"\nAccount balance: "<<balance<<endl;
}

int main(){
    account a;
    clrscr();
    a.getdata();
    a.display();
    int choice;
    char decision;

    cout<<"Do you want to perform some operation in your
account: (Press Y to continue)"<<endl;
    cin>>decision;

    while (decision == 'Y' || decision == 'y'){

        cout<<"\nEnter 1 to check balance"<<endl;
        cout<<"Enter 2 to withdraw"<<endl;
        cout<<"Enter 3 to deposit "<<endl;
        cin>>choice;

        switch(choice){
            case 1: a.display();
                    break;
            case 2: a.withdraw();
                    break;
            case 3 : a.deposit();
                    break;
            default :cout<<"Wrong choice";
        }

        cout<<"\nDo you want to perform some operation in your
account: "<<endl;
    }
}

```

```
        cin>>decision;  
    }  
    getch();  
    return 0;  
}
```

Output:

```
[■] Output 3=[■]  
Enter name:Vanessa  
Enter account number: 2400  
Enter account type: saving  
Enter balance: 30000  
  
Account Holder's Name: Vanessa  
Account balance: 30000  
Do you want to perform some operation in your account: (Press Y to continue)  
y  
  
Enter 1 to check balance  
Enter 2 to withdraw  
Enter 3 to deposit  
2  
Enter amount to withdraw: 3000  
New balance = 27000  
Do you want to perform some operation in your account:  
n  
-
```

Q2. Write a C++ program to make arithmetic calculator using inline function.

Program:

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

class calculate{
    int a,b;

public:
    int add(int x, int y){
        a=x;
        b=y;
        return a+b;
    }

    int sub(int x, int y){
        a=x;
        b=y;
        return a-b;
    }

    int mul(int x, int y){
        a=x;
        b=y;
        return a*b;
    }

    int div(int x, int y){
        a=x;
        b=y;
        return a/b;
    }

};

int main(){
    int a,i,b;
    clrscr();
    calculate c;
    cout<<"Enter 1st number:";
    cin>>a;

    cout<<"Enter 2nd number:";
    cin>>b;

    cout<<endl<<"SELECT 1 FOR ADDITION"<<endl;
    cout<<"SELECT 2 FOR SUBTRACTION"<<endl;
    cout<<"SELECT 3 FOR MULTIPLICATION"<<endl;
    cout<<"SELECT 4 FOR DIVISION"<<endl;
    cin>>i;
```

```

        switch(i){
            case 1: cout<<"Addition = "<<a<< " + "<<b << " =
"<<c.add(a,b);
                    break;
            case 2: cout<<"Subtraction = "<<a<< " - "<<b << " =
"<<c.sub(a,b);
                    break;
            case 3: cout<<"Multiplication = "<<a<< " * "<<b << " =
"<<c.mul(a,b);
                    break;
            case 4: cout<<"Division = "<<a<< " / "<<b << " =
"<<c.div(a,b);
                    break;
            default: cout<<"Wrong choice";

        }

        getch();
        return 0;
}

```

Output:

[■]

Output

```

Enter 1st number:23
Enter 2nd number:14

SELECT 1 FOR ADDITION
SELECT 2 FOR SUBTRACTION
SELECT 3 FOR MULTIPLICATION
SELECT 4 FOR DIVISION
3
Multiplication = 23 * 14 = 322_

```

Q3. Write a program to find largest among given 3 numbers using inline function.

Program:

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

class maximum{

public:
    void check(int x,int y, int z){

        if (x>y&& x>z){
            cout<<"The largest number is: "<<x;
        }
        else if (y>x && y>z){
            cout<<"The largest number is: "<<y;
        }
        else if (z>x && z>y){
            cout<<"The largest number is: "<<z;
        }
        else{
            cout<<"The numbers are equal";
        }
    }

};

int main(){

    maximum m;
    int a,b,c;
    clrscr();
    cout<<"Enter 3 numbers to check the largest: ";
    cin>>a>>b>>c;

    m.check(a,b,c);
    getch();
    return 0;

}
```

Output:

```
[■] Output
Enter 3 numbers to check the largest: 39
64
21
The largest number is: 64_
```

Q4. Write an inline function to obtain the largest of three numbers.

Program:

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

class maximum{

public:
    void check(int x,int y, int z){

        if (x>y&& x>z){
            cout<<"The largest number is: "<<x;
        }
        else if (y>x && y>z){
            cout<<"The largest number is: "<<y;
        }
        else if (z>x && z>y){
            cout<<"The largest number is: "<<z;
        }
        else{
            cout<<"The numbers are equal";
        }
    }

};

int main(){

    maximum m;
    int a,b,c;
    clrscr();
    cout<<"Enter 3 numbers to check the largest: ";
    cin>>a>>b>>c;

    m.check(a,b,c);
    getch();
    return 0;

}
```

Output:

```
[■] Output
Enter 3 numbers to check the largest: 24
13
11
The largest number is: 24_
```

Q5. Create a class distance which accept the distances in feet & inches from the user & display the sum of two distances in feet & inch. [use object as function argument, Write the function inside the class] Modify above program to return the distance as an object of same class.

Program:

```
#include<iostream.h>
#include<conio.h>
class distance{
    int feet, inches;
public:
    void accept(){
        cout<<"Enter feet: ";
        cin>>feet;

        cout<<"Enter inches: ";
        cin>>inches;
    }

    distance add_distance(distance d1, distance d2){
        distance d;
        int e_feet=0;

        d.inches = d1.inches + d2.inches;
        if(d.inches>11){
            int f_inc= d.inches%12;
            e_feet= d.inches/12;
            d.inches = f_inc;
        }

        d.feet = d1.feet + d2.feet + e_feet;
        return (d);
    }

    void display(distance dis){
        cout<<"Total distance= "<< dis.feet<<" feet "<<dis.inches<<"
inches";
    }
};

int main(){

    distance a, b, c, e;
    clrscr();
    cout<<"Distance 1:"<<endl;
    a.accept();
    cout<<"Distance 2:"<< endl;
    b.accept();
    cout<<"Total distance: "<<endl;
    e = c.add_distance(a,b);
    c.display(e);
    getch ();
    return 0;
}
```


Output:

```
[■] Output
Distance 1:
Enter feet: 24
Enter inches: 12
Distance 2:
Enter feet: 42
Enter inches: 12
Total distance:
Total distance= 68 feet 0 inches_
```

Q6. Write a C++ program to concatenate two strings by showing the use of dynamic constructor.

Program:

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

class concat{
    char *str;
    int len;

    public:
        concat(){
            len = 0;
            str= new char[len+1];
        }

        concat(char *s){
            len =strlen(s);
            str = new char[len+1];
            strcpy(str,s);
        }

        void concatenate(concat, concat);
};

void concat :: concatenate(concat a, concat b){
    concat c;

    c.len = a.len + b.len;
    c.str= new char[c.len + 1];
    int count=0;

    for (int i=0; i<=a.len; i++){

        c.str[i]=a.str[i];
        count=i;
    }

    for (int j=0; b.str[j]!= '\0'; j++){
        c.str[count++]=b.str[j];
    }

    cout<<"Concatenated string = "<<c.str;
}

void main(){
    concat c;

    char a[15],b[15];

    clrscr();
    cout<<"string 1: ";
    cin>>a;
```

```
        cout<<"string 2: ";  
        cin>>b;  
        concat x(a);  
        concat y(b);  
        c.concatenate(x,y);  
  
        getch();  
    }
```

Output:

```
[■] Output  
string 1: Vanessa  
string 2: More  
Concatenated string = VanessaMore_
```

Q7. Write a C++ program to solve Quadratic Equation using constructor.

Program:

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

using namespace std;

class quad{
    float a, b, c, dis, x, y;

public:

    quad(){
        cout<<"Enter coefficients of a, b & c: ";
        cin>>a>>b>>c;
        dis = b*b - 4 *a*c;
    }

    void check();
};

void quad :: check(){
    if(dis> 0){
        x = (-b + sqrt(dis))/ (2*a);
        y = (-b - sqrt(dis))/ (2*a);
        cout<<"Roots are real and different";
        cout<<"\n x= "<<x;
        cout<<"\n y= "<<y;
    }

    else if (dis == 0){
        x = -b/ (2* a);
        cout<<"Roots are real and same";
        cout<<"\nx = y = "<<x;
    }
    else {
        cout<<"Roots are complex and different";
        int real = -b/ (2*a);
        int img = sqrt(-dis)/(2*a);
        cout<<"\nx = "<<real<<" + "<<img<<"i";
        cout<<"\ny = "<<real<<" - "<<img<<"i";
    }
}

int main(){
    clrscr();
    quad Q;

    Q.check();
    getch();
    return 0;
}
```

Output:

```
[■] Output
Enter coefficients of a, b & c: 1
2
15
Roots are complex and different
x = -1 + 3i
y = -1 - 3i_
```



Progressive Education Society's
Modern College of Engineering, Pune
MCA Department
A.Y.2022-23

(310907) Object Oriented Programming Laboratory

Class: FY-MCA

Shift / Div: A

Roll Number: 51043

Name: Vanessa Reetu Prashant More

Assignment No: 3

Date of Implementation: 26. 12. 22

Assignment 2 – Q.4 replaced:

Write an inline function in c++ to find odd & even numbers from a set of given numbers. (Make use of an array).

Program:

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

class odd_even{
    int arr[10];

public:
    void get(){

        cout<<"Enter 10 numbers: \n";
        for(int i=0; i<10; i++){
            cout<<"Enter number: ";
            cin>>arr[i];
        }
    }

    void even();
    void odd();
};

inline void odd_even :: even(){
    cout<<"Even Numbers are: ";
    for (int i=0; i<10; i++){
        if (arr[i]%2==0){
            cout<<" "<<arr[i];
        }
    }
}

inline void odd_even :: odd(){
    //int even[10], odd[10];
    //int e=0, o=0;
    cout<<"\nOdd Numbers are: ";
    for (int i=0; i<10; i++){
        if (arr[i]%2!=0){
            cout<<" "<<arr[i];
        }
    }
}
```

```
int main(){
    odd_even obj;
    clrscr();
    obj.get();
    obj.even();
    obj.odd();
    getch();
    return 0;
}
```

Output:

```
Enter 10 numbers:
Enter number: 1
Enter number: 2
Enter number: 3
Enter number: 4
Enter number: 5
Enter number: 6
Enter number: 7
Enter number: 8
Enter number: 9
Enter number: 10
Even Numbers are:  2 4 6 8 10
Odd Numbers are:  1 3 5 7 9
Process returned 0 (0x0)    execution time : 12.322 s
Press any key to continue.
```

Q8. Define a class Animal with their basic features as class members. Create two derived classes from Animal named herbivores and Carnivores (type) with their own features too. Accept name of animal with type and display all the related information.

Program:

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

class animal{
protected:
    char name[10];
    char speak[10];
    int legs;
public:
    void getInfo(){
        legs=4;
        cout<<"Enter Animal Name: ";
        cin>>name;
        cout<<"Enter sound of animal: ";
        cin>>speak;
    }
};

class herbivores : public animal{

public:
    void herbi(){
        cout<<"\n"<<name<<" is a herbivores animal.";
        cout<<"\nNumber of legs: "<<legs;
        cout<<"\nSound of animal: "<<speak;
    }

};

class carnivores : public animal{

public:
    void carni(){
        cout<<"\n"<<name<<" is a carnivores animal.";
        cout<<"\nNumber of legs: "<<legs;
        cout<<"\nSound of animal: "<<speak;
    }

};

int main(){

    int type;
    clrscr();

    cout<<"Enter 1 for Herbivore";
    cout<<"\nEnter 2 for Carnivore"<<endl;
    cin>>type;
```



```
switch(type){  
  
    case 1:  
        herbivores h;  
        h.getInfo();  
        h.herbi();  
        break;  
  
    case 2:  
        carnivores c;  
        c.getInfo();  
        c.carni();  
        break;  
  
    default:  
        cout<<"Wrong choice";  
        break;  
}  
  
getch();  
return 0;  
}
```

Output:

```
Enter 1 for Herbivore  
Enter 2 for Carnivore  
2  
Enter Animal Name: Dog  
Enter sound of animal: barks  
  
Dog is a carnivores animal.  
Number of legs: 4  
Sound of animal: barks  
Process returned 0 (0x0)   execution time : 16.932 s  
Press any key to continue.
```

Q9. Write a program to derive a class rectangle from base class shape using single inheritance.

Program:

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

class shape{
public:
int b, l;
void getdata(){
    cout<<"Function in base class\n";
    cout<<"Enter length: ";
    cin>>l;
    cout<<"Enter breadth: ";
    cin>>b;
}
};

class rectangle : public shape{

public:
void area(){
    cout<<"\nFunction in derived class\n";
    int area = l*b;
    cout<<"The area of rectangle= "<<area;
}
};

int main(){
    rectangle r;
    clrscr();
    r.getdata();
    r.area();
    getch();
    return 0;
}
```

Output:

```
Function in base class
Enter length: 6
Enter breadth: 2

Function in derived class
The area of rectangle= 12
Process returned 0 (0x0)    execution time : 5.648 s
Press any key to continue.
```

Q10. Define a class containing operator function to overload unary minus ('-') operator.

Program:

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

class negative{
    int a,b;
public:
    void getdata(){
        cout<<"Enter 1st number: ";
        cin>>a;
        cout<<"Enter 2nd number: ";
        cin>>b;
    }

    void operator -(){
        a = -a;
        b = -b;
        cout<<"Negated numbers are: "<<a<<" "<<b;
        cout<<"\n";
    }

    /*int operator -(int x){
        return x;
    }*/
};

int main(){
    negative n;
    clrscr();
    n.getdata();
    -n;

    getch();
    return 0;
}
```

Output:

```
Enter 1st number: 2
Enter 2nd number: 4
Negated numbers are: -2 -4
```

Q11. Define base class Student with Roll_No, Name, Marks1, Marks2, Marks3 as data members. Define Sports class with SportName , ParticipationLevel , Achievement as data members and also find sports grade . Inherit Student Class in Sports Class to find final grade of five students. (District/Gold -3 District/Silver – 2 District/Bronze – 1 State/Gold - 6 State /Silver –5 State /Bronze – 4 National/Gold - 9 National/Silver – 8 National/Bronze – 7).

Program:

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

using namespace std;

class student{
protected:
    char s_name[15];
    int s_rollno;
    int marks1;
    int marks2;
    int marks3;

public:
    void accept(){
        cout<<"Enter student name: ";
        cin>>s_name;

        cout<<"Enter roll no. ";
        cin>>s_rollno;

        cout<<"Enter subject 1 marks: ";
        cin>>marks1;

        cout<<"Enter subject 2 marks: ";
        cin>>marks2;

        cout<<"Enter subject 3 marks: ";
        cin>>marks3;

    }
};

class sport : public student{
protected:
    char sportsname[15];
    int achievement;
    char str_ach[10];
    int part_level;
    char str_pl[10];
    int smark;
    char sgrade[10];
    float total;

    float per ;
    char fgrade[10];
```

```

public:
    void getsports();

    void sportgrade(){
        if(part_level==1){
            if (achievement== 1){
                smark =3;
                strcpy(sgrade,"C++");
            }else if (achievement== 2){
                smark =2;
                strcpy(sgrade,"C+");
            }else{
                smark =1;
                strcpy(sgrade,"C");
            }

        }else if(part_level==2){
            if (achievement== 1){
                smark =6;
                strcpy(sgrade,"B++");

            }else if (achievement==2 ){
                smark =5;
                strcpy(sgrade,"B+");
            }else{
                smark =4;
                strcpy(sgrade,"B");
            }
        } else
        {
            if (achievement== 1){
                smark =9;
                strcpy(sgrade,"A++");
            }else if (achievement==2 ){
                smark =8;
                strcpy(sgrade,"A+");
            }else{
                smark =7;
                strcpy(sgrade,"A");
            }
        }
    }

    void finalgrade(){
        total = marks1 + marks2 + marks3 +smark;
        per = total/ 310 * 100;

        if (per>=90){
            strcpy(fgrade,"A");
        }else if( per<90 && per>=70){
            strcpy(fgrade,"B");
        }
    }
}

```

```

        }else if(per<70 && per >=50){
            strcpy(fgrade,"C");

        }else strcpy(fgrade,"D");

    }

    void display(){
        cout<<"Student Name: "<<s_name<<endl;
        cout<<"Student Roll no. : "<<s_rollno<<endl;
        cout<<"Percentage = "<<per<<endl;
        cout<<"Sports Name: "<<sportsname<<endl;
        cout<<"Participation level: "<<str_pl<<endl;
        cout<<"Achievement: "<<str_ach<<endl;
        cout<<"Sports grade: "<<sgrade<<endl;
        cout<<"Final Grade: "<<fgrade<<endl;
    }

};

void sport :: getsports(){

    cout<<"Enter sport name: ";
    cin>>sportsname;

    cout<<"Select Participation level\n1.District\n2.State\n3.National\n";
    cin>>part_level;

    switch(part_level){
        case 1: strcpy(str_pl,"District");
            break;
        case 2: strcpy(str_pl,"State");
            break;
        case 3: strcpy(str_pl,"National");
            break;
        default: cout<<"Wrong Choice";
            break;
    }

    cout<<"Select Achievement\n1. Gold\n2. Silver\n3. Bronze\n";
    cin>>achievement;

    switch(achievement){
        case 1: strcpy(str_ach, "Gold");
            break;
        case 2: strcpy(str_ach, "Silver");
            break;
        case 3: strcpy(str_ach, "Bronze");
            break;
        default: cout<<"Wrong choice";
            break;
    }

}

```

```

int main(){
    //clrscr();
    sport S[5];
    for (int i =0; i<5; i++){
        S[i].accept();
        S[i].getsports();
        S[i].sportgrade();
        S[i].finalgrade();
        S[i].display();
    }

    getch();
    return 0;
}

```

Output:

```

Enter student name: vanessa
Enter roll no. 51043
Enter subject 1 marks: 90
Enter subject 2 marks: 90
Enter subject 3 marks: 90
Enter sport name: badminton
Select Participation level
1.District
2.State
3.National
2
Select Achievement
1. Gold
2. Silver
3. Bronze
1
Student Name: vanessa
Student Roll no. : 51043
Percentage = 89.0323
Sports Name: badminton
Participation level: State
Achievement: Gold
Sports grade: B++
Final Grade: B
Enter student name: arjun
Enter roll no. 24
Enter subject 1 marks: 98
Enter subject 2 marks: 78
Enter subject 3 marks: 80
Enter sport name: cricket
Select Participation level
1.District
2.State
3.National
1

```

```

Select Achievement
1. Gold
2. Silver
3. Bronze
2
Student Name: arjun
Student Roll no. : 24
Percentage = 83.2258
Sports Name: cricket
Participation level: District
Achievement: Silver
Sports grade: C+
Final Grade: B
Enter student name: shruti
Enter roll no. 50
Enter subject 1 marks: 89
Enter subject 2 marks: 87
Enter subject 3 marks: 90
Enter sport name: tennis
Select Participation level
1.District
2.State
3.National
3
Select Achievement
1. Gold
2. Silver
3. Bronze
1

```

Student Name: shruti
Student Roll no. : 50
Percentage = 88.7097
Sports Name: tennis
Participation level: National
Achievement: Gold
Sports grade: A++
Final Grade: B
Enter student name: karan
Enter roll no. 56
Enter subject 1 marks: 77
Enter subject 2 marks: 79
Enter subject 3 marks: 68
Enter sport name: chess
Select Participation level
1.District
2.State
3.National
2
Select Achievement
1. Gold
2. Silver
3. Bronze
3

Student Name: karan
Student Roll no. : 56
Percentage = 73.5484
Sports Name: chess
Participation level: State
Achievement: Bronze
Sports grade: B
Final Grade: B
Enter student name: tina
Enter roll no. 43
Enter subject 1 marks: 88
Enter subject 2 marks: 76
Enter subject 3 marks: 91
Enter sport name: basketball
Select Participation level
1.District
2.State
3.National
1
Select Achievement
1. Gold
2. Silver
3. Bronze
1

Student Name: tina
Student Roll no. : 43
Percentage = 83.2258
Sports Name: basketball
Participation level: District
Achievement: Gold
Sports grade: C++
Final Grade: B

Q12. Write C++ program to read the base class Information i.e. employee name, code, designation. And the derived class contains years of experience and age. Design a virtual base class for the item employee name and code.

Program:

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

class employee{

    public:
    char ename[15];
    int ecode;
    void getdata(){
        cout<<"Enter Employee Name: ";
        cin>>ename;

        cout<<"Enter Employee code: ";
        cin>>ecode;
    }
};

class des : virtual public employee{
    public:
    char edesign[15];
    void getdes(){
        cout<<"Enter Employee Designation: ";
        cin>>edesign;
    }
};

class details : virtual public employee{

    public:
    int years;
    int age;

    void getdetails(){
        cout<<"Enter employee age: ";
        cin>>age;

        cout<<"Enter years of experience: ";
        cin>>years;
    }
};
```

```

class item: public des, public details{
    public:
        void print(){
            cout<<"\nEmployee Information: "<<<endl;
            cout<<"Employee name: "<<<ename<<<endl;
            cout<<"Employee code: "<<<ecode<<<endl;
            cout<<"Employee designation: "<<<edesign<<<endl;
            cout<<"Employee age: "<<<age<<<endl;
            cout<<"Years of Experience: "<<<years<<<endl;
        }
};

int main(){
    item i;
    clrscr();
    i.getdata();
    i.getdes();
    i.getdetails();
    i.print();
    getch();
    return 0;
}

```

Output:

```

Enter Employee Name: Vanessa
Enter Employee code: 24
Enter Employee Designation: Manager
Enter employee age: 22
Enter years of experience: 2

Employee Information:
Employee name: Vanessa
Employee code: 24
Employee designation: Manager
Employee age: 22
Years of Experience: 2

```

Q13. Assume that the company markets books and audio tapes. Prepare a class publication having members title and price. From this class derive 2 classes 1) book which adds a page count and 2) class tape that adds playing time in minutes. Further add a base class that holds sales from both books and tapes. Write the program in c++ that displays all books having page count more than 100 and all cassettes that play for more than 60 minutes

Program:

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

class publication{
protected:
    char title[20];
    float price;
public:
    void getInfo(){
        cout<<"Enter title: ";
        cin>>title;
        cout<<"Enter price: ";
        cin>>price;
    }
};

class sales{
protected:
    float sale;
public:
    void getSale(){
        cout<<"Enter sales: ";
        cin>>sale;
    }
};

class book : public publication,public sales{
protected:
    int page_count;
public:
    void getCount(){
        cout<<"Enter page count: ";
        cin>>page_count;
    }

    void display(){

        if (page_count>100){
            cout<<"\nBook Information: "<<endl;
            cout<<"\nBook title: "<<title;
            cout<<"\nPrice: "<<price;
            cout<<"\nPage Count: "<<page_count;
            cout<<"\nSales: Rs "<<sale;
        }
    }
}
```

```
};
```

```
class tape : public publication,public sales{
protected:
    int minutes;
public:
    void getMin(){
        cout<<"Enter playing minutes: ";
        cin>>minutes;
    }

    void show(){

        if (minutes>60){
            cout<<"\nTape Information: "<<endl;
            cout<<"\nTape title: "<<title;
            cout<<"\nPrice: "<<price;
            cout<<"\nPlaying Minutes: "<<minutes;
            cout<<"\nSales: Rs "<<sale;
        }

    }

};
```

```
int main(){

    int c;
    clrscr();
    cout<<"Press 1 for book"<<endl;
    cout<<"Press 2 for tape"<<endl;
    cin>>c;

    switch(c){
    case 1:
        book b[3];
        for(int i=0; i<3; i++){
            cout<<"\nEnter Information for Book "<<i+1<<endl;
            b[i].getInfo();
            b[i].getSale();
            b[i].getCount();
        }

        cout<<"\nBooks with page count more than 100"<<endl;
        for(int i=0; i<3; i++){
            b[i].display();
        }
        break;

    case 2:
        tape t[3];
```

```

        for(int i=0; i<3; i++){
            cout<<"\nEnter Information for Tape "<<i+1<<endl;
            t[i].getInfo();
            t[i].getSale();
            t[i].getMin();
        }

        cout<<"\nTape with minutes more than 60"<<endl;
        for(int i=0; i<3; i++){
            t[i].show();
        }
        break;

    default:
        cout<<"Wrong choice";
        break;
    }

    getch();
    return 0;
}

```

Output:

```

Enter Information for Book 1
Enter title: abc
Enter price: 24
Enter sales: 50
Enter page count: 150

Enter Information for Book 2
Enter title: xyz
Enter price: 60
Enter sales: 90
Enter page count: 90

Enter Information for Book 3
Enter title: pqr
Enter price: 50
Enter sales: 40
Enter page count: 200

Books with page count more than 100

Book Information:

Book title: abc
Price: 24
Page Count: 150
Sales: Rs 50
Book Information:

Book title: pqr
Price: 50
Page Count: 200
Sales: Rs 40
Process returned 0 (0x0)   execution time : 103.104 s
Press any key to continue.

```

Q14. Write a class ‘Point’ with x and y coordinates as data members. Derive two classes ‘Line’ and Circle’ from ‘Point’ with appropriate data members. Derive a class ‘Triangle’ from class ‘Line’. Implement read () and draw () member functions for all the above classes.

Program:

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

class Point{
protected:
    float x_cord, y_cord;
public:
    void read(){
        cout<<"Enter x coordinate: ";
        cin>>x_cord;
        cout<<"Enter y coordinate: ";
        cin>>y_cord;
    }
    void draw(){
        cout<<"Drawing between coordinates: "<<x_cord<<" & "<<y_cord<<endl;
    }
};

class Line : public Point{
protected:
    float length;
public:
    void getLine(){
        cout<<"Enter length of line: ";
        cin>>length;
    }
    void drawLine(){
        cout<<"length of line= "<<length<<endl;
    }
};

class Circle : public Point{
protected:
    float radius;
public:
    void getCircle(){
        cout<<"Enter radius of circle: ";
        cin>>radius;
    }
    void drawCircle(){
        cout<<"Radius of circle= "<<radius<<endl;
    }
};

class Triangle : public Line{
protected:
```

```

        float s1,s2,s3;
public:
    void getTriangle(){
        cout<<"Enter 3 sides of Triangle: ";
        cin>>s1>>s2>>s3;
    }

    void drawTriangle(){
        cout<<"Triangle has sides: "<<s1<<" "<<s2<<" & "<<s3<<endl;
    }
};

int main(){
    Triangle t;
    Circle c;
    clrscr();

    cout<<"Calling member functions of Point & Line Class through Triangle: "<<endl;
        t.read();
        t.draw();
        t.getLine();
        t.drawLine();
        t.getTriangle();
        t.drawTriangle();

    cout<<endl<<"Calling member functions of Point Class through Circle: "<<endl;
        c.read();
        c.draw();
        c.getCircle();
        c.drawCircle();

    getch();
    return 0;
}

```

Output:

```

Calling member functions of Point & Line Class through Triangle:
Enter x coordinate: 6
Enter y coordinate: 9
Drawing between coordinates: 6 & 9
Enter length of line: 5
length of line= 5
Enter 3 sides of Triangle: 3 4 6
Triangle has sides: 3, 4 & 6

Calling member functions of Point Class through Circle:
Enter x coordinate: 3
Enter y coordinate: 9
Drawing between coordinates: 3 & 9
Enter radius of circle: 7
Radius of circle= 7

Process returned 0 (0x0)   execution time : 36.792 s
Press any key to continue.

```

Q15. Create a class FLOAT that contains one float data member. Overload all the four arithmetic operators so that they operate on the objects of FLOAT.

Program:

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

class FLOAT{

    float n;

public:
    void get(int num){
        n = num;
    }

    void display(){
        cout<<n;
    }

    FLOAT operator + (FLOAT f1){
        FLOAT f2;
        f2.n = n + f1.n;
        return f2;
    }

    FLOAT operator - (FLOAT f1){
        FLOAT f2;
        f2.n = n - f1.n;
        return f2;
    }

    FLOAT operator * (FLOAT f1){
        FLOAT f2;
        f2.n = n * f1.n;
        return f2;
    }

    FLOAT operator / (FLOAT f1){
        FLOAT f2;
        f2.n = n / f1.n;
        return f2;
    }

};

int main(){

    FLOAT f1, f2, f3;
    float n1, n2;
    clrscr();
```



```
cout<<"Enter Number 1: ";
cin>>n1;

cout<<"Enter Number 2: ";
cin>>n2;

f1.get(n1);
f2.get(n2);

cout<<"\nAddition of num: ";
f3 = f1 + f2;
f3.display();

cout<<"\nSubtraction of num: ";
f3 = f1 - f2;
f3.display();

cout<<"\nMultiplication of num: ";
f3 = f1 * f2;
f3.display();

cout<<"\nDivision of num: ";
f3 = f1 / f2;
f3.display();

getch();
return 0;
}
```

Output:

```
Enter Number 1: 24.1
Enter Number 2: 143.2

Addition of num: 167
Subtraction of num: -119
Multiplication of num: 3432
Division of num: 0.167832
Process returned 0 (0x0)    execution time : 9.362 s
Press any key to continue.
```

Q16. Create a class MAT of size m*n. Define all possible matrix operations for MAT type objects.

Program:

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

class MAT{
    int matrix[10][10];
    int m,n;

public:

    MAT(){
        //default constructor
    }

    MAT (int r, int c){
        //parameterized constructor
        m = r;
        n = c;
    }

    MAT sub(MAT);
    MAT add(MAT);
    MAT mul(MAT);

    void accept(){
        int i,j;

        for (i=0; i<m; i++){
            for (j=0; j<n; j++){
                cout<<"Enter value: "<<i<<","<<j<<": ";
                cin>>matrix[i][j];
            }
        }
    }

    void display(){
        int i,j;

        for (i=0; i<m; i++){
            for (j=0; j<n; j++){
                cout<<matrix[i][j]<<" ";
            }
            cout<<endl;
        }
    }

};
```

```

MAT MAT :: sub(MAT M2){

    MAT M3(m,n);
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<n;j++)
        {
            M3.matrix[i][j]=matrix[i][j] - M2.matrix[i][j];
        }
    }
    return M3;

}

MAT MAT :: add (MAT M2){
    MAT M3(m,n);
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<n;j++)
        {
            M3.matrix[i][j]=matrix[i][j]+ M2.matrix[i][j];
        }
    }
    return M3;
}

MAT MAT :: mul (MAT M2){
    MAT M3(m,n);
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<n;j++)
        {
            M3.matrix[i][j]=matrix[i][j] * M2.matrix[i][j];
        }
    }
    return M3;
}

int main(){

int r,c;
clrscr();
cout<<"Enter row: ";
cin>>r;
cout<<"Enter column: ";
cin>>c;

MAT m1(r,c);
MAT m2(r,c);
MAT m3(r,c);

cout<<"Elements for matrix 1: "<<endl;
m1.accept();

```

```

        cout<<"\nElements for matrix 2: "<<endl;
        m2.accept();

        cout<<"\nAddition of Matrices= "<<endl;
        m3= m1.add(m2);
        m3.display();

        cout<<"\nSubtraction of Matrices= "<<endl;
        m3 = m1.sub(m2);
        m3.display();

        cout<<"\nMultiplication of Matrices= "<<endl;
        m3 = m1.mul(m2);
        m3.display();

        getch();
        return 0;
    }

```

Output:

```

Enter row: 3
Enter column: 2
Elements for matrix 1:
Enter value: 0,0: 1
Enter value: 0,1: 2
Enter value: 1,0: 3
Enter value: 1,1: 4
Enter value: 2,0: 1
Enter value: 2,1: 2

Elements for matrix 2:
Enter value: 0,0: 3
Enter value: 0,1: 2
Enter value: 1,0: 1
Enter value: 1,1: 3
Enter value: 2,0: 1
Enter value: 2,1: 1

Addition of Matrices=
4 4
4 7
2 3

Subtraction of Matrices=
-2 0
2 1
0 1

Multiplication of Matrices=
3 4
3 12
1 2

Process returned 0 (0x0)   execution time : 30.824 s
Press any key to continue.

```

Q17. Write a C++ program to concatenate two Strings using operator + function.

Program:

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

class concat{
    char str[25];

public:
    void getString(){
        cout<<"Enter string: ";
        cin>>str;
    }

    concat operator + (concat c1){
        concat c2;
        strcat(str , c1.str );
        strcpy(c2.str, str);
        return c2;
    }
    void display(){
        cout<<str;
    }
};

int main(){
    concat c1, c2, c3;
    clrscr();
    c1.getString();
    c2.getString();
    c3 = c1 + c2;
    cout<<"\nConcatenated string: ";
    c3.display();
    getch();
    return 0;
}
```

Output:

```
Enter string: hello
Enter string: vanessa

Concatenated string: hellovanessa
Process returned 0 (0x0)   execution time : 6.337 s
Press any key to continue.
```



Progressive Education Society's
Modern College of Engineering, Pune
MCA Department
A.Y.2022-23

(310907) Object Oriented Programming Laboratory

Class: FY-MCA

Shift / Div: A

Roll Number: 51043

Name: Vanessa Reetu Prashant More

Assignment No: 4

Date of Implementation: 16. 1. 23

Q18. Create a base class shape. Derive two classes as Triangle and Rectangle from the base class shape. Take getdata() and display_area() as member functions of base class. Make display_area() as a virtual function and redefine it in derived classes to suit their requirement. Design a program that will accept dimensions of a triangle or rectangle interactively and display the area.

Program:

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

class shape{
    protected:
        float length ,base ,height, breadth;

    public:

        virtual void get_data(){
            cout<<"Get_data function called in base class\n";
        }

        virtual void display_area(){
            cout<<"Display Area function called in base class\n";
        }

};

class triangle: public shape
{
    public:
        void get_data(){
            cout<<"\nEnter Triangle dimensions";
            cout<<"\nEnter base & height in m: ";
            cin>>base>>height;
        }

        void display_area(){
            float area = 0.5 * base * height;
            cout<<"\nArea of triangle= "<<area<<" sq m";
        }

};

class rectangle : public shape
{
    public:
```

```

        void get_data(){
            cout<<"\nEnter Rectangle dimensions";
            cout<<"\nEnter length & breadth in m: ";
            cin>>length>>breadth;
        }

        void display_area(){
            float area = length * breadth;
            cout<<"\nArea of rectangle= "<<area<<" sq m";
        }
    };
    int main() {

        clrscr();
        shape s;
        triangle t;
        rectangle r;

        s.get_data();
        s.display_area();

        cout<<"\nGet_data function called in Triangle derived class";
        t.get_data();
        cout<<"\nDisplay Area function called in Triangle derived class";
        t.display_area();

        cout<<"\n\nGet_data function called in Rectangle derived class";
        r.get_data();
        cout<<"\nDisplay Area function called in Rectangle derived class";
        r.display_area();

        getch();
        return 0;
    }

```

Output:

```

Get_data function called in base class
Display Area function called in base class

Get_data function called in Triangle derived class
Enter Triangle dimensions
Enter base & height in m: 4 8

Display Area function called in Triangle derived class
Area of triangle= 16 sq m

Get_data function called in Rectangle derived class
Enter Rectangle dimensions
Enter length & breadth in m: 12 5

Display Area function called in Rectangle derived class
Area of rectangle= 60 sq m
Process returned 0 (0x0)   execution time : 31.883 s
Press any key to continue.

```

Q19. Write a C++ program to demonstrate a pure virtual function which is invoked from the object of derived class through the pointer of the base class. Base class contains getdata() and display(). Display the information of employee using this.

Program:

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

class base{
protected:
    char name[15];
    int id;
public:
    virtual void getdata()=0; //Pure Virtual Function

    virtual void display()=0; //Pure Virtual Function
};

class derive: public base{
protected:
    char design[10];
    int exp;
    float sal;

public:
    void getdata(){
        cout<<"\n\nTaking info from derived class";
        cout<<"\nEnter Employee Name: ";
        cin>>name;
        cout<<"\nEnter Employee Id No. : ";
        cin>>id;
        cout<<"\nEnter Designation: ";
        cin>>design;
        cout<<"\nEnter Years of Experience: ";
        cin>>exp;
        cout<<"\nEnter Salary: ";
        cin>>sal;
    }

    void display(){
        cout<<"\nPrinting Employee Info from Derived class";
        cout<<"\nEmployee name: "<<name;
        cout<<"\nEmployee Id: "<<id;
        cout<<"\nEmployee Designation: "<<design;
        cout<<"\nEmployee Experience: "<<exp<<" years";
        cout<<"\nEmployee Salary: "<<sal;
    }
};
```



```
int main(){

clrscr();
base *ptr;

derive d;

ptr=&d;
ptr->getdata();
ptr->display();

getch();
return 0;
}
```

Output:

```
Taking info from derived class
Enter Employee Name: vanessa

Enter Employee Id No. : 24

Enter Designation: manager

Enter Years of Experience: 2

Enter Salary: 50000

Printing Employee Info from Derived class
Employee name: vanessa
Employee Id: 24
Employee Designation: manager
Employee Experience: 2 years
Employee Salary: 50000
```

Q20. Consider a book shop which sales books and video tapes. Define a class Media which store title and price of a publication. Create two derive classes book and tape where book class is used to store no of pages of book and tape class is used to store playing time of a tape. Use display function in all the classes and show the use of virtual function.

Program:

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

class media
{
    protected:
        char title[10];
        float price;

    public:
        void get_media(){
            cout<<"\nEnter Publication Name: ";
            cin>>title;
            cout<<"\nEnter Price: ";
            cin>>price;
        }

        virtual void display(){
            cout<<"\nName of Publication: "<<title;
            cout<<"\nPrice: Rs "<<price;
        }

};

class book : public media
{
    protected:
        int pc;
    public:
        void get_book(){
            cout<<"\nEnter Page count: ";
            cin>>pc;
        }

        void display(){
            cout<<"\nName of Publication: "<<title;
            cout<<"\nPrice: Rs "<<price;
            cout<<"\nPage count: "<<pc;
        }

};
```

```

class tapes : public media
{
    protected:
        int pt;
    public:
        void get_tape(){
            cout<<"\nEnter playing time: ";
            cin>>pt;
        }

        void display(){
            cout<<"\nName of Publication: "<<title;
            cout<<"\nPrice: Rs "<<price;
            cout<<"\nPlaying Time: "<<pt;
        }
};

int main(){

    clrscr();
    media m;
    book b;
    tapes t;

    m.get_media();
    m.display();

    cout<<"\n\nEnter Book Info: ";
    b.get_media();
    b.get_book();
    cout<<"\nBook Info: ";
    b.display();

    cout<<"\n\nEnter Tapes Info: ";
    t.get_media();
    t.get_tape();
    cout<<"\nTapes Info: ";
    t.display();

    getch();
    return 0;
}

```

Output:

```
Enter Publication Name: penguin
Enter Price: 300
Name of Publication: penguin
Price: Rs 300
Enter Book Info:
Enter Publication Name: star
Enter Price: 250
Enter Page count: 90
Book Info:
Name of Publication: star
Price: Rs 250
Page count: 90
Enter Tapes Info:
Enter Publication Name: tseries
Enter Price: 40
Enter playing time: 60
Tapes Info:
Name of Publication: tseries
Price: Rs 40
Playing Time: 60
Process returned 0 (0x0)   execution time : 45.929 s
Press any key to continue.
```

Q21. Write a C++ program to maintain the records of person with details (Name and Age) and find the eldest among them. The program must use this pointer to return the result.

Program:

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

class person{

    char name[15];
    int age;

    public:
        void getdetails(){
            cout<<"\nEnter Name: ";
            cin>>name;
            cout<<"\nEnter Age: ";
            cin>>age;
        }

        void display(){
            cout<<"\nPerson Name: "<<name;
            cout<<"\nPerson Age: "<<age;
        }

        person check(person p1){
            if(p1.age >= age)
                return p1;
            else
                return *this;
        }
};

int main(){

    clrscr();
    person p[10];
    person &p1 = p[0];
    int i,n;

    cout<<"Enter number of people to compare: ";
    cin>>n;

    for (i=0;i<n;i++){
        p[i].getdetails();
    }
```

```
    for (i=0;i<n;i++){  
        p1 = p1.check(p[i]);  
    }  
  
    cout<<"\nEldest Person is: ";  
    p1.display();  
  
    getch();  
    return 0;  
}
```

Output:

```
Enter number of people to compare: 4  
  
Enter Name: ajay  
  
Enter Age: 24  
  
Enter Name: riya  
  
Enter Age: 32  
  
Enter Name: anuj  
  
Enter Age: 18  
  
Enter Name: lisa  
  
Enter Age: 44  
  
Eldest Person is:  
Person Name: lisa  
Person Age: 44  
Process returned 0 (0x0)    execution time : 91.408 s  
Press any key to continue.
```

Q22. Create a class 'Account' that stores customer name, account number and type of account. From this derive the classes cur_acct and sav_acct to make them more specific to their requirements. Include necessary member functions in order to achieve the following tasks:

- a) Accept deposit from a customer and update the balance.**
- b) Display the balance.**
- c) Compute and deposit interest.**
- d) Permit withdrawal and update the balance.**

Program:

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

class account{
protected:
    char name[15];
    int acc_num;
    float balance = 0;

public:
    void getinfo(){
        cout<<"\nEnter Customer name: ";
        cin>>name;
        cout<<"\nEnter Account Number: ";
        cin>>acc_num;
    }

    void display(){
        cout<<"\nAccount Holder's Name: "<<name;
        cout<<"\nAccount Number: "<<acc_num;
        cout<<"\nAvailable Balance: Rs "<<balance;
    }

    void deposit(){
        float amt;
        cout<<"\nEnter Amount to deposit: ";
        cin>>amt;
        balance += amt;
        cout<<"\nUpdated Balance: Rs "<<balance;
    }

};

class cur_acct : public account{
protected:
    float amt;
public:
    void withdraw(){
        cout<<"\nEnter Amount to withdraw: ";
        cin>>amt;

        if (amt>balance){
            cout<<"\nInsufficient balance!!";
```

```

        }else{
            balance -= amt;
            cout<<"\nUpdated Balance: Rs "<<balance;
        }
    }
};

```

```

class sav_acct : public account{
protected:
    int intr;
public:
    void interest(){
        int t=2,r=8;
        intr=(balance * r * t)/100;
        cout<<"\nInterest: Rs "<<intr;
        balance+=intr;
        cout<<"\nUpdated Balance: Rs "<<balance;
    }

    void withdraw(){
        float amt;
        cout<<"\nEnter Amount to withdraw: ";
        cin>>amt;

        if (amt>balance){
            cout<<"\nInsufficient balance!!";
        }else{
            balance -= amt;
            cout<<"\nUpdated Balance: Rs "<<balance;
        }
    }
};

```

```

int main(){
    account a;
    sav_acct s;
    cur_acct c;
    int ch;

    cout<<"Select Account Type";

    int type;
    cout<<"\nEnter 1 for Savings Account\nEnter 2 for Current Account"<<endl;
    cin>>type;

    switch(type){
    case 1:
        s.getinfo();
        s.display();

        do{
            cout<<"\n\nPress 1 to Deposit\nPress 2 to Withdraw\nPress 3 to Display\nPress 4 to Compute
and deposit Interest\nPress 5 to Exit\n";

```



```

cin>>ch;

switch(ch){
case 1:
    s.deposit();
    break;
case 2:
    s.withdraw();
    break;
case 3:
    s.display();
    break;
case 4:
    s.interest();
    break;
case 5:
    exit(0);
    break;
default:
    cout<<"Wrong choice";
}
}while(ch<=5);
break;

```

```

case 2:
    c.getinfo();
    c.display();

do{
    cout<<"\n\nPress 1 to Deposit\nPress 2 to Withdraw\nPress 3 to Display\nPress 4 to Exit\n";
    cin>>ch;

    switch(ch){
    case 1:
        c.deposit();
        break;
    case 2:
        c.withdraw();
        break;
    case 3:
        c.display();
        break;
    case 4:
        exit(0);
        break;
    default:
        cout<<"Wrong choice";
        break;
    }
}while(ch<=4);
break;

```

```

default:
    cout<<"Wrong choice";

    }
getch();
return 0;
}

```

Output:

```

Select Account Type
Enter 1 for Savings Account
Enter 2 for Current Account
1

Enter Customer name: vanessa

Enter Account Number: 2424

Account Holder's Name: vanessa
Account Number: 2424
Available Balance: Rs 0

Press 1 to Deposit
Press 2 to Withdraw
Press 3 to Display
Press 4 to Compute and deposit Interest
Press 5 to Exit
1

Enter Amount to deposit: 5000

Updated Balance: Rs 5000

Press 1 to Deposit
Press 2 to Withdraw
Press 3 to Display
Press 4 to Compute and deposit Interest
Press 5 to Exit
2

```

```

Enter Amount to withdraw: 200

Updated Balance: Rs 4800

Press 1 to Deposit
Press 2 to Withdraw
Press 3 to Display
Press 4 to Compute and deposit Interest
Press 5 to Exit
3

Account Holder's Name: vanessa
Account Number: 2424
Available Balance: Rs 4800

Press 1 to Deposit
Press 2 to Withdraw
Press 3 to Display
Press 4 to Compute and deposit Interest
Press 5 to Exit
4

Interest: Rs 768
Updated Balance: Rs 5568

Press 1 to Deposit
Press 2 to Withdraw
Press 3 to Display
Press 4 to Compute and deposit Interest
Press 5 to Exit
5

```



Progressive Education Society's
Modern College of Engineering, Pune
MCA Department
A.Y.2022-23

(310907) Object Oriented Programming Laboratory

Class: FY-MCA

Shift / Div: A

Roll Number: 51043

Name: Vanessa Reetu Prashant More

Assignment No: 5

Date of Implementation: 20. 2. 23

23. Write a C++ program for Exception Handling Divide by zero Using C++ Programming.

Program:

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

using namespace std;

int main(){

    float n1,n2,div;

    try {
        cout<<"Enter 2 numbers: ";
        if(!(cin>>n1>>n2)){
            throw "Wrong Input";
        }
        else if(n2==0){
            throw (n2);
        }
        else{
            div=n1/n2;
            cout<<"Division n1/n2 = "<<div;
        }
    }
    catch(float n2){
        cout<<"Division by 0 is not possible\n n2= "<<n2;
    }

    catch(...){
        cout<<"Wrong Input";
    }
    return 0;
}
```

Output:

```
Enter 2 numbers: 24 0
Division by 0 is not possible
n2= 0
```

In Editor:

```
Enter 2 numbers: 24 0
Division by 0 is not possible
n2= 0
Process returned 0 (0x0)   execution time : 9.468 s
Press any key to continue.
```

▮

24. Write a C++ program to sort an array in ascending order using function template.

Program:

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

template <class T>
T sort(T arr[], int n){
    int size, i, j, index;
    T temp,min;
    size = sizeof(T);

    for (i=0; i<n; i++){
        min=arr[i];
        index=i;

        for(j=i;j<n;j++){
            {
                if(arr[j]<min){
                    index=j;
                    min=arr[j];
                }
            }
            temp=arr[index];
            arr[index]=arr[i];
            arr[i]=temp;
        }
    }

}

int main(){
    int min, i, j, n;
    float arr[100];
    cout<<"\nEnter the number of element in array:";
    cin>>n;

    cout<<"\nEnter array elements:"<<endl;
    for(i=0;i<n;i++){
        cin>>arr[i];
    }

    sort<float>(arr,n);

    cout<<"\nSorted Array elements:"<<endl;
    for(i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }

    return 0;
}
```

Output:

Enter the number of element in array:10

Enter array elements:

24 56 111 34 98 42 92 88 40 13

Sorted Array elements:

13 24 34 40 42 56 88 92 98 111

In Editor:

```
Enter the number of element in array:10
```

```
Enter array elements:
```

```
24 56 111 34 98 42 92 88 40 13
```

```
Sorted Array elements:
```

```
13 24 34 40 42 56 88 92 98 111
```

```
Process returned 0 (0x0)    execution time : 26.107 s
```

```
Press any key to continue.
```

25. Write a C++ program to swap a data using function template.

Program:

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

template <class T>
int swap_numbers(T& x, T& y)
{
    T t;
    t = x;
    x = y;
    y = t;
    return 0;
}

int main()
{
    int a, b;
    cout<<"Enter 2 numbers: ";
    cin>>a>>b;

    cout<<"\nNumbers before swap: ";
    cout<<"\na= "<<a<<" b= "<<b;
    // Invoking the swap()
    swap_numbers(a, b);
    cout<<"\nNumbers after swap: ";
    cout<<"\na= "<<a<<" b= "<<b;
    return 0;
}
```

Output:

Enter 2 numbers: 40 30

Numbers before swap:

a= 40 b= 30

Numbers after swap:

a= 30 b= 40

In Editor:

```
Enter 2 numbers: 40 30

Numbers before swap:
a= 40 b= 30
Numbers after swap:
a= 30 b= 40
Process returned 0 (0x0)    execution time : 5.834 s
Press any key to continue.
```

26. Write a C++ program to make a simple calculator using class template.

Program:

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

template <class T>
class Calculator
{
private:
    T num1, num2;

public:
    Calculator(T n1, T n2)
    {
        num1 = n1;
        num2 = n2;
    }

    void displayResult()
    {
        cout << "Numbers are: " << num1 << " and " << num2 << "." << endl;
        cout << "Addition is: " << add() << endl;
        cout << "Subtraction is: " << subtract() << endl;
        cout << "Product is: " << multiply() << endl;
        cout << "Division is: " << divide() << endl;
    }

    T add() { return num1 + num2; }

    T subtract() { return num1 - num2; }

    T multiply() { return num1 * num2; }

    T divide() { return num1 / num2; }
};

int main()
{
    Calculator<int> intCalc(30, 5);
    Calculator<float> floatCalc(24.2, 10.2);

    cout << "Int results:" << endl;
    intCalc.displayResult();

    cout << endl << "Float results:" << endl;
    floatCalc.displayResult();

    return 0;
}
```


Output:

Int results:
Numbers are: 30 and 5.
Addition is: 35
Subtraction is: 25
Product is: 150
Division is: 6

Float results:
Numbers are: 24.2 and 10.2.
Addition is: 34.4
Subtraction is: 14
Product is: 246.84
Division is: 2.37255

In Editor:

```
Int results:
Numbers are: 30 and 5.
Addition is: 35
Subtraction is: 25
Product is: 150
Division is: 6

Float results:
Numbers are: 24.2 and 10.2.
Addition is: 34.4
Subtraction is: 14
Product is: 246.84
Division is: 2.37255

Process returned 0 (0x0)   execution time : 0.199 s
Press any key to continue.
```

27. Write a C++ program implementing stack and it's operations using template class.

Program:

```
#include <iostream>
#define MAX 5
using namespace std;

template <class T>

class Stack{
protected:
    T arr[MAX];
    int top;
public:
    Stack(){
        cout<<"Stack created Successfully"<<endl;
        top =-1;
    }

    void push(){
        T a;
        if(isFull()){
            cout<<"Stack is full"<<endl;

        }
        else{
            cout<<"Enter element u want to push: ";
            cin>>a;
            arr[++top]=a;
        }
    }

    void pop(){
        T a;
        if (isEmpty()){
            cout<<"Stack is empty"<<endl;
        }
        else{
            a=arr[top--];
            cout<<"Popped element is: "<<a<<endl;
        }
    }

    int isFull(){
        if(top==MAX-1){
            return 1;
        }
        else{
            return 0;
        }
    }

    int isEmpty(){
```

```

        if(top==-1)
            return 1;
        else
            return 0;
    }

    void display(){
        for(int i=0;i<=top;i++){
            cout<<" "<<arr[i];
        }
    }
};

int main(){
    int ch;
    Stack<int> s;
    do{
        cout<<"\n1.Push\n2.Pop\n3.isFull\n4.isEmpty\n5.Display\n6.Exit\nEnter your choice:";
        cin>>ch;

        switch(ch){
            case 1:
                s.push();
                break;

            case 2:
                s.pop();
                break;

            case 3:
                if(s.isFull()){
                    cout<<"\nStack is full"<<endl;
                }
                else{
                    cout<<"\nStack is not full"<<endl;
                }
                break;

            case 4:
                if(s.isEmpty()){
                    cout<<"\nStack is Empty"<<endl;
                }
                else{
                    cout<<"\nStack is not Empty"<<endl;
                }
                break;

            case 5:
                s.display();
                break;

            case 6:
                break;

```

```

        default:
            cout<<"Enter correct choice: "<<endl;
            break;
        }
    }while(ch!=6);

    return 0;
}

```

Output:

Stack created Successfully

1.Push
 2.Pop
 3.isFull
 4.isEmpty
 5.Display
 6.Exit
 Enter your choice:1
 Enter element u want to push: 10

1.Push
 2.Pop
 3.isFull
 4.isEmpty
 5.Display
 6.Exit
 Enter your choice:1
 Enter element u want to push: 20

1.Push
 2.Pop
 3.isFull
 4.isEmpty
 5.Display
 6.Exit
 Enter your choice:1
 Enter element u want to push: 30

1.Push
 2.Pop
 3.isFull
 4.isEmpty
 5.Display
 6.Exit
 Enter your choice:1
 Enter element u want to push: 40

1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit
Enter your choice:1
Enter element u want to push: 50

1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit
Enter your choice:1
Stack is full

1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit
Enter your choice:2
Popped element is: 50

1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit
Enter your choice:3

Stack is not full

1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit
Enter your choice:4

Stack is not Empty

1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit

Enter your choice:5

10 20 30 40

1.Push

2.Pop

3.isFull

4.isEmpty

5.Display

6.Exit

Enter your choice:6

In Editor:

```
Stack created Successfully
```

```
1.Push
```

```
2.Pop
```

```
3.isFull
```

```
4.isEmpty
```

```
5.Display
```

```
6.Exit
```

```
Enter your choice:1
```

```
Enter element u want to push: 10
```

```
1.Push
```

```
2.Pop
```

```
3.isFull
```

```
4.isEmpty
```

```
5.Display
```

```
6.Exit
```

```
Enter your choice:1
```

```
Enter element u want to push: 20
```

```
1.Push
```

```
2.Pop
```

```
3.isFull
```

```
4.isEmpty
```

```
5.Display
```

```
6.Exit
```

```
Enter your choice:1
```

```
Enter element u want to push: 30
```

```
1.Push
```

```
2.Pop
```

```
3.isFull
```

```
4.isEmpty
```

```
5.Display
```

```
6.Exit
```

```
Enter your choice:1
```

```
Enter element u want to push: 40
```

```
1.Push
```

```
2.Pop
```

```
3.isFull
```

```
4.isEmpty
```

```
5.Display
```

```
6.Exit
```

```
Enter your choice:1
```

```
Enter element u want to push: 50
```

```
1.Push
```

```
2.Pop
```

```
3.isFull
```

```
4.isEmpty
```

```
5.Display
```

```
6.Exit
```

```
Enter your choice:1
```

```
Stack is full
```

```
1.Push
```

```
2.Pop
```

```
3.isFull
```

```
4.isEmpty
```

```
5.Display
```

```
6.Exit
```

```
Enter your choice:2
```

```
Popped element is: 50
```

```
1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit
Enter your choice:3
```

```
Stack is not full
```

```
1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit
Enter your choice:4
```

```
Stack is not Empty
```

```
1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit
Enter your choice:5
 10 20 30 40
```

```
1.Push
2.Pop
3.isFull
4.isEmpty
5.Display
6.Exit
Enter your choice:6
```

28. Write a C++ program implementing linked list & some required operations on it using class template.

Program:

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

using namespace std;

template<class T>

class Node{
public:
    T data;
    Node *next;
};

template <class T>
class LinkedList{
protected:
    Node<T> *start;
public:
    LinkedList(){
        start = NULL;
    }

    void createList(){
        Node<T> *temp,*newnode;

        int n,i;

        cout<<"\nEnter number of nodes: ";
        cin>>n;

        temp=start;
        for(i=0;i<n;i++){
            newnode =createNode();
            if(start==NULL)
                temp=start=newnode;
            else{
                temp->next=newnode;
                temp=temp->next;
            }
        }
    }

    void display(){
        Node<T> *temp;
        cout<<endl;
        cout<<"\nLinked list= ";
        for(temp=start; temp!=NULL;temp=temp->next)
            cout<<temp->data<<" ";
    }
}
```



```

Node<T> *createNode(){
    Node<T> *newnode;
    newnode = (Node<T> *)malloc(sizeof(Node<T>));
    cout<<"\nEnter data: ";
    cin>>newnode->data;
    newnode->next=NULL;
    return newnode;
}

void insert_pos(){
    int pos,i;
    Node<T> *t1, *t2, *temp;
    temp = (Node <T>*)malloc(sizeof(Node<T>));
    cout<<"Enter Data: ";
    cin>>temp->data;
    cout<<"Enter Position: ";
    cin>>pos;
    t2=start;
    for(i=0;i<pos-1;i++){
        t1=t2;
        t2=t1->next;
    }
    t1->next=temp;
    temp->next=t2;
}

void insertStart(){
    Node<T> *newnode;
    newnode = createNode();
    if(start==NULL)
        start=newnode;
    else{
        newnode->next=start;
        start=newnode;
    }
}

void insertEnd(){
    Node<T> *newnode, *temp;
    newnode = createNode();
    if(start==NULL)
        start=newnode;
    else{
        for(temp=start;temp->next!=NULL; temp=temp->next);
        temp->next=newnode;
    }
}

```

```

void deleteStart(){
    Node<T> *temp, *temp1;
    if(start==NULL)
        cout<<"\nList is already empty"<<endl;
    else{
        temp = start;
        start = start->next;
        temp->next=NULL;
        free(temp);
    }
}

void deleteEnd(){
    Node<T> *temp, *temp1;
    for(temp=start; temp->next->next!=NULL;temp=temp->next);
    temp1=temp->next;
    temp->next=NULL;
    free(temp);
}

void del_pos(){
    int i, pos;
    Node<T> *t1, *t2;
    t2 =start;

    cout<<"Enter position:";
    cin>>pos;
    for(i=0; i<pos-1;i++)
    {
        t1=t2;
        t2 = t1->next;
    }
    t1->next=t2->next;
    free(t2);
}

};

int main(){

    LinkedList<int> ls;

    int ch;
    do{
        cout<<"\n1.CreateList.\n2.Insert at Start\n3.Insert at End\n4.Insert at position\n5.Delete
Start\n6.Delete End\n7.Delete position\n8.Display\n9.Exit"<<"\nMake a choice"<<endl;
        cin>>ch;
    }
}

```

```
switch(ch){
    case 1:
        ls.createList();
        ls.display();
        break;

    case 2:
        ls.insertStart();
        ls.display();
        break;

    case 3:
        ls.insertEnd();
        ls.display();
        break;

    case 4:
        ls.insert_pos();
        ls.display();
        break;

    case 5:
        ls.deleteStart();
        ls.display();
        break;

    case 6:
        ls.deleteEnd();
        ls.display();
        break;

    case 7:
        ls.del_pos();
        ls.display();
        break;

    case 8:
        ls.display();
        break;

    case 9:
        exit(0);
        break;
    default:
        cout<<"Wrong choice";
        break;
}
}while(ch!=9);

return 0;
}
```

Output:

- 1.CreateList.
- 2.Insert at Start
- 3.Insert at End
- 4.Insert at position
- 5.Delete Start
- 6.Delete End
- 7.Delete position
- 8.Display
- 9.Exit

Make a choice
1

Enter number of nodes: 3

Enter data: 20

Enter data: 30

Enter data: 40

Linked list= 20 30 40

- 1.CreateList.
- 2.Insert at Start
- 3.Insert at End
- 4.Insert at position
- 5.Delete Start
- 6.Delete End
- 7.Delete position
- 8.Display
- 9.Exit

Make a choice
2

Enter data: 10

Linked list= 10 20 30 40

- 1.CreateList.
- 2.Insert at Start
- 3.Insert at End
- 4.Insert at position
- 5.Delete Start
- 6.Delete End
- 7.Delete position
- 8.Display
- 9.Exit

Make a choice
3

Enter data: 50

Linked list= 10 20 30 40 50

- 1.CreateList.
 - 2.Insert at Start
 - 3.Insert at End
 - 4.Insert at position
 - 5.Delete Start
 - 6.Delete End
 - 7.Delete position
 - 8.Display
 - 9.Exit
- Make a choice
- 4
- Enter Data: 15
- Enter Position: 2

Linked list= 10 15 20 30 40 50

- 1.CreateList.
 - 2.Insert at Start
 - 3.Insert at End
 - 4.Insert at position
 - 5.Delete Start
 - 6.Delete End
 - 7.Delete position
 - 8.Display
 - 9.Exit
- Make a choice
- 5

Linked list= 15 20 30 40 50

- 1.CreateList.
 - 2.Insert at Start
 - 3.Insert at End
 - 4.Insert at position
 - 5.Delete Start
 - 6.Delete End
 - 7.Delete position
 - 8.Display
 - 9.Exit
- Make a choice
- 6

Linked list= 15 20 30 40

1.CreateList.

2.Insert at Start

3.Insert at End

4.Insert at position

5.Delete Start

6.Delete End

7.Delete position

8.Display

9.Exit

Make a choice

7

Enter position:2

Linked list= 15 30 40

1.CreateList.

2.Insert at Start

3.Insert at End

4.Insert at position

5.Delete Start

6.Delete End

7.Delete position

8.Display

9.Exit

Make a choice

8

Linked list= 15 30 40

1.CreateList.

2.Insert at Start

3.Insert at End

4.Insert at position

5.Delete Start

6.Delete End

7.Delete position

8.Display

9.Exit

Make a choice

9

In Editor:

```
1.CreateList.
2.Insert at Start
3.Insert at End
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
1
Enter number of nodes: 3
Enter data: 20
Enter data: 30
Enter data: 40
Linked list= 20 30 40
1.CreateList.
2.Insert at Start
3.Insert at End
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
2
Enter data: 10
```

```
Linked list= 10 20 30 40
1.CreateList.
2.Insert at Start
3.Insert at End
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
3
Enter data: 50
Linked list= 10 20 30 40 50
1.CreateList.
2.Insert at Start
3.Insert at End
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
4
Enter Data: 15
Enter Position: 2
Linked list= 10 15 20 30 40 50
```

```
Linked list= 10 15 20 30 40 50
1.CreateList.
2.Insert at Start
3.Insert at End
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
5
Linked list= 15 20 30 40 50
1.CreateList.
2.Insert at Start
3.Insert at End
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
6
```

```
Linked list= 15 20 30 40
1.CreateList.
2.Insert at Start
3.Insert at End
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
7
Enter position:2
Linked list= 15 30 40
1.CreateList.
2.Insert at Start
3.Insert at End
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
8
```

```
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
8
Linked list= 15 30 40
1.CreateList.
2.Insert at Start
3.Insert at End
4.Insert at position
5.Delete Start
6.Delete End
7.Delete position
8.Display
9.Exit
Make a choice
9
Process returned 0 (0x0)   execution time : 79.055 s
Press any key to continue.
```



Progressive Education Society's
Modern College of Engineering, Pune
MCA Department
A.Y.2022-23

(310907) Object Oriented Programming Laboratory

Class: FY-MCA

Shift / Div: A

Roll Number: 51043

Name: Vanessa Reetu Prashant More

Assignment No: 6

Date of Implementation: 20. 2. 23

29. Write a program to implement I/O operations on characters. I/O operations includes inputting a string, Calculating length of the string, Storing the string in a file, fetching the stored characters from it, etc.

Program:

```
#include<iostream.h>
#include<conio.h>
#include<fstream.h>
#include<string.h>
int main(){
    char string[80];
    clrscr();
    cout<<"Enter a string\n";
    cin>>string;
    int len=strlen(string);

    fstream file;
    file.open("Text",ios::in|ios::out);

    cout<<"String succesfully put in file"<<endl;

    for(int i=0;i<len;i++){
        file.put(string[i]);
    }

    file.seekg(0);
    char ch;
    cout<<"File data: "<<endl;
    while(file){
        file.get(ch);
        cout<<ch;
    }

    return 0;
}
```

Output:

```
Enter a string
DemoFile
String succesfully put in file
File data:
DemoFile
```


In Editor:

[]

Output

3=[]

```
Enter a string
DemoFile
String succesfully put in file
File data:
DemoFile
```

30. Write a program to copy the contents of one file to another.

Program:

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

int main(){

    char src[50],tgt[50];
    char ch;
    clrscr();

    cout<<endl<<"Enter source filename: ";
    cin>>src;

    cout<<endl<<"Enter target filename: ";
    cin>>tgt;

    ifstream infile(src);
    ofstream outfile(tgt);

    while(infile){

        infile.get(ch);
        outfile.put(ch);

    }
    cout<<"\nData copied successfully";

    return 0;
}
```

Output:

```
Enter source filename: F1.txt

Enter target filename: F2

Data copied successfully
```

In Editor:



```
[■] Output [■]
```

```
Enter source filename: F1.txt
Enter target filename: F2
Data copied successfully_
```

31. Write a program to maintain a elementary database of employees using files.

Program:

```
#include<iostream.h>
#include<string.h>
#include<stdio.h>
#include<stdlib.h>
#include<fstream.h>
#include<conio.h>

//using namespace std;
char empfile[30] = "Employee.txt";
char ITfile[20] = "IT.txt";
char Adminfile[25] = "Admin.txt";
char Prodfile[30] = "Production.txt";
char Salesfile[30] = "Sales.txt";

class emp
{
    int empid;
    char name[30];
    char address[60];
    int age;
public:
    char dept[15];
    void get();
    char *getdept()
    {
        return dept;
    }
};

void emp::get()
{
    cout<<"\n Enter Employee Id   : ";
    cin>>empid;
    cout<<"\n Enter Name           : ";
    cin>>name;
    cout<<"\n Enter Address        : ";
    cin>>address;
    cout<<"\n Enter Department Name:(Admin/Sales/IT/Production) : ";
    cin>>dept;
    cout<<"\n Enter Age    : ";
    cin>>age;
}

void insert()
{
    emp e;

    ofstream fout; //ofstream is a class, fout is its object. It can be used only to write into the file.

    //file is open in the binary, append and nocreate mode.
    fout.open("Employee.txt",ios::in | ios::out | ios::binary | ios::app | ios::ate);
```

```

if (fout.fail())
{
    cout<<"\n Unable to Open the File!!!";
    goto err;
}
e.get(); // accepting the details from the user.
fout.write((char *)&e,sizeof(e)); //writing into the file with fout object.
if(fout.tellp()%sizeof(e)==0)
{
    cout<<"\n Record Inserted !!!"<<endl;
}
else
{
    cout<<"\n Insertion Failed !!!";
    goto err;
}
err:
    fout.close();
}

void sort() // This function will insert the record according to department in respective file.
{
    emp e;
    ofstream adm,sal,pro,it; //all files have been created for writing mode.
    ifstream fin; // fin object belongs to the ifstream class, it is used to read the file contents only.
    adm.open(Adminfile, ios::out | ios::binary | ios::app);
    sal.open(Salesfile, ios::out | ios::binary | ios::app);
    pro.open(Prodfile, ios::out | ios::binary | ios::app);
    it.open(ITfile, ios::out | ios::binary | ios::app);
    fin.open(empfile, ios::in | ios::binary);
    while(fin.read((char *)&e,sizeof(e))) //reading the file contents till it reaches end of file.
    {
        if(strcmp(e.getdept(),"Admin")==0)
        {
            adm.write((char *)&e,sizeof(e));
            cout<<"\n Record Inserted into ADMIN File!!!";
        }
        else if(strcmp(e.getdept(),"Sales")==0)
        {
            sal.write((char *)&e,sizeof(e));
            cout<<"\n Record Inserted into SALES File!!!";
        }
        else if(strcmp(e.getdept(),"IT")==0)
        {
            it.write((char *)&e,sizeof(e));
            cout<<"\n Record Inserted into IT File!!!";
        }
        else if(strcmp(e.getdept(),"Production")==0)
        {
            pro.write((char *)&e,sizeof(e));
            cout<<"\n Record Inserted into Production File!!!";
        }
        else
    }
}

```

```

        cout<<"\n Insert Correct Record!!!";
    }
    fin.close();
    adm.close();
    sal.close();
    it.close();
    pro.close();
}
int main()
{
    int n;
    clrscr();
    cout<<"\n Enter No. of Records You Want? : ";
    cin>>n;
    for(int i=0; i<n; i++)
    {
        insert();
    }
    sort();
    return 0;
}

```

Output:

Enter Name : vanessa

Enter Address : pune

Enter Department Name:(Admin/Sales/IT/Production) : IT

Enter Age : 23

Record Inserted !!!

Enter Employee Id : 2

Enter Name : Tina

Enter Address : bangalore

Enter Department Name:(Admin/Sales/IT/Production) : Sales

Enter Age : 30

Record Inserted !!!

Record Inserted into IT File!!!

Record Inserted into SALES File!!!

In Editor:

```
Output
Enter Name      : vanessa
Enter Address   : pune
Enter Department Name:(Admin/Sales/IT/Production) : IT
Enter Age      : 23
Record Inserted !!!
Enter Employee Id : 2
Enter Name      : Tina
Enter Address   : bangalore
Enter Department Name:(Admin/Sales/IT/Production) : Sales
Enter Age      : 30
F1 Help  ↑↓↔ Scroll
```

```
Output
Enter Department Name:(Admin/Sales/IT/Production) : IT
Enter Age      : 23
Record Inserted !!!
Enter Employee Id : 2
Enter Name      : Tina
Enter Address   : bangalore
Enter Department Name:(Admin/Sales/IT/Production) : Sales
Enter Age      : 30
Record Inserted !!!
Record Inserted into IT File!!!
Record Inserted into SALES File!!!_
F1 Help  ↑↓↔ Scroll
```

32. Write a program for reading and writing data to and from the file using command line arguments.

Program:

```
#include<fstream.h>
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<stdlib.h>
//using namespace std;

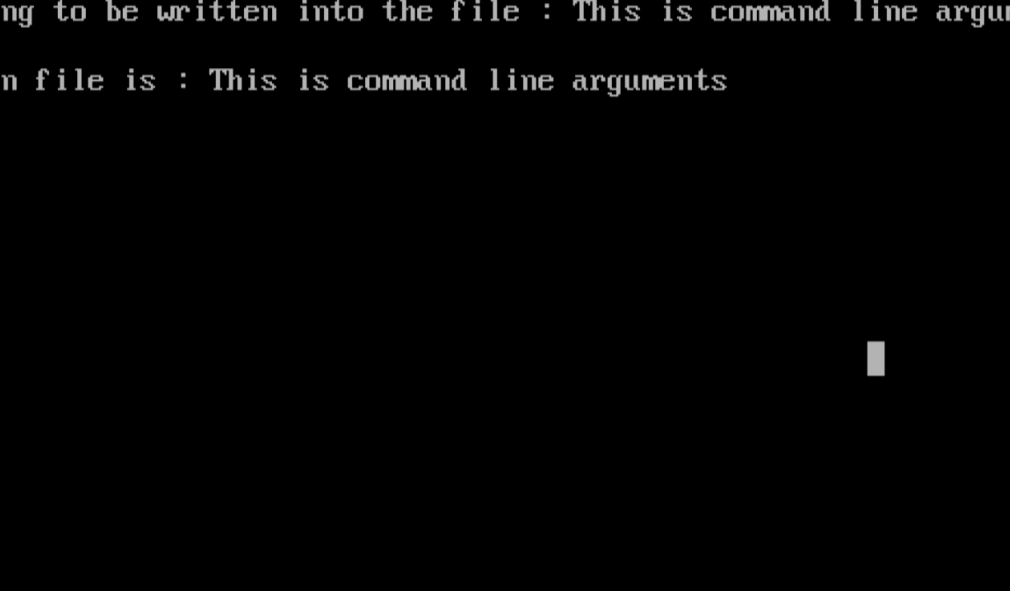
int main ( int argc, char *argv[] )
{
    char ch[50];
    clrscr();
    ofstream ofs;
    system("clear");
    ofs.open(argv[1]);
    if(!ofs)
    {
        cout<<"\ncould not open file";
        exit(1);
    }
    cout<<"\nEnter string to be written into the file : ";
    gets(ch);
    ofs<<ch;
    ofs.close();
    if ( argc != 2 )
        cout<<"usage: "<< argv[0] <<" <filename>\n";
    else
    {
        ifstream fs(argv[1]);
        if (!fs)
        {
            cout<<"could not open file\n";
            exit(1);
        }
        else
        {
            cout<<"\nthe data in file is : ";
            char x;
            while (fs.get(x))
                cout<< x;
        }
    }
    return 0;
}
```

Output:

Enter string to be written in the file: This is command line arguments

the data in file is: This is command line arguments

In Editor:



```
[ ] Output 2=[ ]
Illegal command: clear.

Enter string to be written into the file : This is command line arguments
the data in file is : This is command line arguments

F1 Help  ↑↓↔ Scroll
```


33. Write a function in C++ to count and display the number of lines not starting with alphabet 'A' present in a text file "STORY.TXT".

Program:

```
#include<iostream.h>
#include<fstream.h>
#include<conio.h>
//using namespace std;
int main()
{
    ifstream fin;
    clrscr();
    fin.open("Story.txt");

    char str[100];
    int count=0;
    int count1=0;

    while(!fin.eof())
    {
        fin.getline(str,100);
        if(str[0]!='A')
        {
            count++;
        }
        else if(str[0]=='A'&&str[1]!=' ')
        {
            count1++;
        }
    }

    cout<<"The number of lines not starting with 'A' are:"<<count<<"\n";
    cout<<"The number of lines starting with 'A' are:"<<count1<<"\n";

    fin.close();
    return 0;
}
```

Output:

The number of lines not starting with 'A' are:5
The number of lines starting with 'A' are: 1;

In Editor:



```
[ ■ ] Output
The number of lines not starting with 'A' are:5
The number of lines starting with 'A' are:1
```