Assignment 1

//PROGRAM TO FIND AREA OF CIRCLE..

```
#include<iostream>
using namespace std;
int main()
{
  int r;
  float area;
  cout<<"Enter the radius of circle-";
  cin>>r;
  area=3.14*r*r;
  cout<<"Area of circle="<<area;
  return 0;
}
```

//PROGRAM TO FIND MAXIMUM OF THREE NUMBERS

```
#include<iostream>
using namespace std;
int main()
{
  int x,y,z;
  cout<<"Enter first number-";
   cin>>x;
  cout<<"Enter second number-";
   cin>>y;
  cout<<"Enter third number-";
   cin>>z;

   if(x>y && x>z)
```

```
           { cout<<"%d is greater"<<x;}
   if(y>x && y>z)
            { cout<<"%d is greater"<<y;}
  else
            { cout<<"%d is greater"<<z; }
    return 0;
}
//PROGRAM TO FIND GIVEN NUMBER IS ODD OR EVEN
#include<iostream>
using namespace std;
int main()
{
  int num;
  cout<<"Enter the number-";
  cin>>num;
  if(num%2==0)        //...remember this condition
           cout<<"Given number is Even....";
    else
            cout<<"Given number is Odd..";
  return 0;
}
//PROGRAM TO FIND THE FACTORIAL OF GIVEN NUMBER...
#include<iostream>
using namespace std;
int main()
{
  int num,factorial=1;
  cout<<"Accept the number-";
  cin>>num;

  for(int i=1;i<=num;i++)
   {
    factorial*=i;
   }
   cout<<"Factorial is = "<<factorial;
}
//PROGRAM TO PRINT FLOYDS TRIANGLE..
#include<iostream>
using namespace std;
int main()
{
  int rows,num=1;
  cout<<"Enter the number of rows-";
  cin>>rows;
   for(int i=1;i<=rows;++i)
   {
    for(int j=1;j<=i;++j)
      {
        cout<<num<<" ";
        ++num;
      }
    cout<<" "<<endl;
   }
 return 0;
}
//PROGRAM TO GENERATE A TABLE OF ANY NUMBER
```

```
#include<iostream>
using namespace std;
int main()
{
  int num;
  cout<<"Enter the number to generate multiplication table- ";
  cin>>num;
  cout<<"Multiplication table of "<<num<<" is as follows"<<endl;
  for(int i = 1 ; i < = 10  ; + + i)
     {
            cout<<num*i<<endl ;
     }
   return 0;
}
//PROGRAM TO RIGHT ORIENTED TIANGLE
#include<iostream>
using namespace std;
int main()
{
 int i,j,k,row,b=0,a=1;
 cout<<"enter the number of row =";
 cin>>row;
 for(i=row;i>0;i--)
    {
      for(j=i;j>0;j--)
        {
          cout<<"\t";
        }
      for(k=0;k<=b;k++)
       {
         cout<<a<<"\t";
         a++;
       }
      b++;
     cout<<endl;
    }
    return 0;
}
```

===

Assignment 2

```
// set A Q1)
#include<iostream>
using namespace std;
class area
{
private:
    float length, breadth; // for rectangle
    float side; // square
    float radius; // circle
    float base, height; // triangle

public:
    void area_rectangle()
    {
        cout << "Enter the length and breadth of rectangle to calculate area: ";
        cin >> length >> breadth;
        float area = length * breadth;
        cout << "Area of rectangle: " << area << endl;
    }

    void area_square()
    {
        cout << "Enter the side of square to calculate area: ";
        cin >> side;
        float area = side * side;
        cout << "Area of square: " << area << endl;
    }

    void area_triangle()
    {     cout << "Enter the base and height of triangle to calculate area: ";
         cin >> base >> height;
         float area = 1.0/3 * base * height; // Use 1.0 to force floating-point division
         cout << "Area of triangle: " << area << endl;
    }

    void area_circle()
    {   cout << "Enter the radius of circle to calculate area: ";
         cin >> radius;
        float area = 3.14 * radius * radius;
        cout << "Area of circle: " << area << endl;
    }
};

int main()
```

```
{
   area A;      //creating object A
   A.area_rectangle();
   A.area_square();
   A.area_circle();
   A.area_triangle();

}
```

---

set B Q1)

```
#include<iostream>
using namespace std;
class account
{
  private:
        int  acc_no;
        char name[30];
         char ac_type[20];
         int amount;
  public:
       void accept_data(); // function declaration is her but definition is outside the
class
       void display();
};
 void account::accept_data()
   {
   cout<<"Enter Account number:-" ;
   cin>>acc_no;

   cout<<"Enter owner name:-" ;
   cin>>name;

   cout<<"Ener Account type:-" ;
   cin>>ac_type;

   cout<<"Enter Amount:-" ;
   cin>>amount;
   }

void account:: display()
   {
   cout<<"Account number= "<<acc_no<<endl;
   cout<<"Owner Name= "<<name<<endl;
   cout<<"Account type= "<<ac_type<<endl;
   cout<<"Amount= "<<amount<<endl;
   }

int main()
 {
    account A[100];
    int n,i;
    cout<<"Ente the numbers of account to accept details-";
    cin>>n;
     for(i=0;i<n;i++)
        A[i].accept_data();
         cout<<endl;
         cout<<"Details of user are "<<endl;
    for(i=0;i<n;i++)
```

```
          A[i].display();
      cout<<endl;
    return 0;
}
```

// set C Q1)

```
#include<iostream>
using namespace std;
class time1
{
   private:
   int hour,minute,second;
   public:
             void settime(int h,int m,int s)
                       {
                           hour=h;
                           minute=m;
                           second=s;
                       }
         void showtime()
                       {
                           cout<<"Time:-"<<hour<<":"<<minute<<":"<<second<<endl;
                       }
          time1 add(time1 t)
              {
                 time1 r;
                 r.second=second+t.second;
                 r.minute=minute+t.minute;
                 r.hour=hour+t.hour;
                if(r.hour>=24)
                    {       r.hour-=24;         }
                if(r.minute>=60)
                 {       r.minute-=60;
                            r.hour++;         }
                if(r.second>=60)
                 {     r.second-=60;
                       r.minute++;    }
                      return r;
            }
// function to accept data from user
   void gettime()
   {
      int h, m, s;
       cout << "Enter hours: ";
       cin >> h;
       cout << "Enter minutes: ";
       cin >> m;
       cout << "Enter seconds: ";
       cin >> s;
      settime(h, m, s);
  }
 };
int main()
{
  time1 t1;
  time1 t2;
```

```
    time1 t3;
    t1.settime(12,35,59);
    t2.settime(10,25,20);
    cout<<"Time 1: ";
    t1.showtime();
    cout<<"Time 2: ";
    t2.showtime();
    cout << "Time 1 + Time 2:- ";
    t3=t1.add(t2);
    t3.showtime();
    return 0;
}
-------------------------------------------------------------
     // another maon function aacept input from user
Int main()
{
     time1 t1, t2, t3;
    cout << "Enter Time 1:" <<  endl;
    t1.gettime();
     cout << "Enter Time 2:" <<  endl;
    t2.gettime();
     cout << "Time 1: ";
    t1.showtime();
     cout << "Time 2: ";
    t2.showtime();
    t3 = t1.add(t2);
    cout << "Time 1 + Time 2: ";
    t3.showtime();
```

```
   return 0;
 }
```

```
// set C Q2)
 #include<iostream>
 using namespace std;
 class distance1
 {
   private:
   int feet, inch;
   public:
   void getdistance()
   {
       cout<<" Enter feet :-";
       cin>>feet;
       cout<<" Enter inches :-";
       cin>>inch;
   }
   void putdistance()
   {
       cout<<"Distance:-"<<feet<<" feet and "<<inch<<" inches "<<endl;
   }
   distance1 add(distance1 d)
   {
       distance1 r;   //r stands for result
       r.feet=feet+d.feet;
       r.inch=inch+d.inch;
       if(r.inch>=12)
```

```
                {
                    r.inch-=12;
                    r.feet++;
                }
            return r;
        }
 };
 int main()
 {
    distance1 d1,d2,d3;
    cout<<"Enter first distance in feet and inches:-"<<endl;
    d1.getdistance();
    cout<<endl;
    cout<<"Enter second distance in feet and inches:-"<<endl;
    d2.getdistance();
    cout<<endl;
     cout<<"Addition of both distances is:-"<<endl;
     d3=d1.add(d2);
     d3.putdistance();
     return 0;
 }
```

Assignment 3

```
//  set A Q1--
#include<iostream>
#include<cmath>
using namespace std;
inline double root(double num)
{
    return sqrt(num);
}
inline double cube(double num)
{
    return num*num*num;
}
int main()
{
    double num;
    cout<<"Enter the number-";
    cin>>num;
    cout<<"Square root of number is:- "<<root(num)<<endl;
    cout<<"Cube of number is:- "<<cube(num)<<endl;
    return 0;
}
-----------------------------------------------------------------
// set A Q2)
#include<iostream>
using namespace std;
class number2;// class declaration
class number1   // class for accept number one
{
    private:
    int num1;
    public:
    void getnum()
    {
        cout<<"Enter first number :-"<<endl;
        cin>>num1;
    }
        friend int subtract (number1 n1,number2 n2);
        //defining friend function to acces the first number from this class
};
class number2  // class for accept number two
{
    private:
    int num2;
    public:
     void getnum()
    {
        cout<<"Enter second number :-"<<endl;
        cin>>num2;
    }
   friend int subtract (number1 n1,number2 n2);
   //defining friend function to acces the second number from this class
};

int subtract (number1 n1,number2 n2)//subtraction function
{
     return n1.num1-n2.num2;
}

int main()
{
    number1 N1;
    number2 N2;
    cout<<"Enter both the numbers-"<<endl;
    N1.getnum();
    N2.getnum();
    int result=subtract(N1,N2);
    cout<<"subtraction is:"<<result<<endl;
}
-----------------------------------------------------------------------
// set B Q1)
#include<iostream>
using namespace std;
class sport_item;// class declaration
class ele_item   // class for accept amount of electronic items
{
    private:
    int amount1;
    public:
    void getnum()
    {
        cout<<"Enter the amount of electrical items:-"<<endl;
        cin>>amount1;
    }
        friend int add (ele_item n1,sport_item n2);
        //defining friend function to acces the first number from this class
};
class sport_item  // class for accept number of sport items
{
    private:
    int amount2;
    public:
     void getnum()
    {
        cout<<"Enter the amount of sport items:-"<<endl;
        cin>>amount2;
    }
      friend int add(ele_item n1,sport_item n2);
      //defining friend function to acces the second number from this class
};

int add(ele_item n1,sport_item n2)//addition function
{
     return n1.amount1+n2.amount2;
}
int main()
{
    ele_item N1;
    sport_item N2;
    cout<<"Enter the number of items-"<<endl;
    N1.getnum();
    N2.getnum();
    int result=add(N1,N2);
    cout<<"Addition of items is:"<<result<<endl;
}
------------------------------------------------------------
// set B Q2)
#include<iostream>
using namespace std;
```

```
class emp
{
    private:
    int empno;
    string empname;
    float salary;
    public:
    void getdata()
    {
        cout<<"Enter employee number-";
        cin>>empno;
        cout<<"Enter employee name-";
         cin>>empname;
        cout<<"Enter employee salary-";
        cin>>salary;
    }
    inline void display()
    {
        cout<<"Employee number-"<<empno<<endl;
        cout<<"Employee name-"<<empname<<endl;
        cout<<"Employee salary-"<<salary<<endl;
    }

};
int main()
{
    int i;
    emp employee[3];
    cout<<"enter the details of 3 employee-"<<endl;
    for(i=0;i<3;i++)
    {
        cout<<"Enter details of employee "<<i+1<<endl;
        employee[i].getdata();
    }
    cout<<"printing details of three employee "<<endl;

    for(i=0;i<3;i++)
    {
     employee[i].display();
     cout<<endl;
    }
    return 0;
}
-----------------------------------------------------------------------
 // set B Q3)

#include<iostream>
using namespace std;

class sem2; // Forward declaration of sem2 class

class sem1
{
    int total_marks1;
    public:
    void getmarks()
    {
    cout << "Enter sem 1 marks out of 500- ";
    cin >> total_marks1;


    }
    friend int add(sem1, sem2);
 };

class sem2
{
    int total_marks2;


     public:
     void getmarks()
    {
    cout << "Enter sem 2 marks out of 500- ";
    cin >> total_marks2;
    }
     friend int add(sem1 n1, sem2 n2);
 };

int add(sem1 n1, sem2 n2)
 {
    return n1.total_marks1 + n2.total_marks2;
}

int main()
{
    string studentname;
    sem1 s1;
    sem2 s2;
    cout << "Enter student name- ";
    cin >> studentname;
     s1.getmarks();
     s2.getmarks();
    int total = add(s1,s2);
    cout << "Name of the student- " << studentname << endl;
    cout << "Addition of marks of both semesters is " << total << endl;

    return 0;
}------------------------------------------------------------
//set c remaining
#include <iostream>
#include <string>
using namespace std;

class Jewellery
{
private:
    static int count; // static data member to keep track of number of objects
    int jewellery_Code;
    string jewellery_Name;
    float jewellery_Price;
public:
    Jewellery()
    {
```

```
    objectcount(); // Call objectcount() in the constructor
 }
void objectcount()
 {
    count++;
 }

 void acceptInfo()
 {
    cout << "Enter Jewellery Code: ";
    cin >> jewellery_Code;
    cin.ignore();
    cout << "Enter Jewellery Name: ";
    cin>> jewellery_Name;
    cout << "Enter Jewellery Price: ";
    cin >> jewellery_Price;
 }

 void displayInfo()
 {
    cout << "Jewellery Code: " << jewellery_Code << endl;
    cout << "Jewellery Name: " << jewellery_Name << endl;
    cout << "Jewellery Price: " << jewellery_Price << endl;
 }

 static void displayCount()
 {
    cout << "Number of Jewellery objects created: " <<count<< endl;
    }
};
int Jewellery::count=0;
int main()
{
    Jewellery j1, j2, j3; // creating objects of Jewellery class
    j1.acceptInfo();
    j2.acceptInfo();
    j3.acceptInfo();

    cout << "\nDisplaying Information for Jewellery Objects:" << endl;
    j1.displayInfo();
    j2.displayInfo();
    j3.displayInfo();
    Jewellery::displayCount(); // calling static member function to display count
    return 0;
}
```

Assignment 4

```
//Set A q1)
#include<iostream>
using namespace std;
class volume
{
    public:
    int vol(int side)
    {
        return side*side*side;
    }
    double vol(double height,double radius)
    {
        return 3.14*radius*radius*height;
    }
    double vol(double radius)
    {
        return (4.0/3.0)*3.14*radius*radius*radius;
    }

};
int main()
{
   volume v;
   double radius,height;
   int side;
   cout<<"Enter the side of the cube-";
   cin>>side;
   cout<<"Volume of cube is-"<<v.vol(side)<<endl;

   cout<<"Enter the radius and height of cylinder-";
   cin>>height>>radius;
   cout<<"Volume of cylinder is-"<<v.vol(height,radius)<<endl;

   cout<<"Enter the radius of sphere-";
   cin>>radius;
   cout<<"Volume of sphere is-"<<v.vol(radius)<<endl;
  return 0;
```

```
}
------------------------------------------------------
// set A Q2)
#include <iostream>
using namespace std;
class intrest
{
public:
    int calculate(int p, int n, int r = 5)
    {
        return (p * n * r) / 100;
    }

    double calculate(double p, int n = 10, float r = 7.5)
    {
        return (p * n * r) / 100;
    }
};

int main()
{
    intrest t,l;
    int amount, n;
    double r;
    cout << "Enter the amount: ";
    cin >> amount;
    cout << "Enter the duration: ";
    cin >> n;
    cout << "Total interest on " << amount << " in duration of " << n << " years by
the rate 5 is " << t.calculate(amount, n) << endl;

    cout << "Total interest on " << amount << " by the rate 7.5 in 10 years " <<
l.calculate(amount) << endl;

    return 0;
}
```

```
//Set B Q1)

#include<iostream>
using namespace std;
class overload
{
public:
    void calculate()
    {
        cout << "Welcome to function overloading program" << endl;
    }

    int calculate(int num)
    {
        cout << "absolute value of accepted number:" << num << endl;
        return num;
    }

    float calculate(int num1, float num2)
    {
        cout << "Addition of both parameters is: ";
        return num1 + num2;
    }

    int calculate(int x, int y, int z)
    {
        if(x > y && x > z)
            return x;
        else if(y > x && y > z)
            return y;
        else
            return z;
    }
};

int main()
{
    overload f;
    int num1, x, y, z;
    float num2;

    f.calculate();//first

    cout << "Enter an integer: ";
    cin >> num1;
    f.calculate(num1);//second
    cout << "Enter two numbers (an integer and a decimal): ";
    cin >> num1 >> num2;
    cout << "Addition of both parameters is: "<< f.calculate(num1, num2) <<
endl;//third

    cout << "Enter three integers to check which one is greater: ";
    cin >> x >> y >> z;
    cout << "The greatest number is: " << f.calculate(x, y, z) << endl;//fourth
    return 0;
}
---------------------------------------------
//set c Q1)
#include<iostream>
#include<string>
using namespace std;

class person
{
private:
    string p_name;
    string city;
    int mob_no;
public:
    void acceptdata()
    {
        cout<<"Enter persons name-";
        cin>>p_name;
        cout<<"Enter city-";
        cin>>city;
        cout<<"Enter mobile number-";
        cin>>mob_no;
    }
    void display()
    {
        cout<<"Person name-"<<p_name<<endl;
        cout<<"city-"<<city<<endl;
        cout<<"mobile number-"<<mob_no<<endl;
    }
    int searchmobno(string name)
    {
        if(name==p_name)
         return mob_no;
        else
         return-1;
    }
    void searchinfo(int mobno)
    {
        if(mobno==mob_no)
         {
           cout<<"Person name-"<<p_name<<endl;
           cout<<"City-"<<city<<endl;
         }
    }
};
int main()
{
    person p1,p2;
```

```
   string searchname;
   int mobileno;
   cout<<"Enter details of person 1:"<<endl;
   p1.acceptdata();
   cout<<"Enter details of person 2:"<<endl;
   p2.acceptdata();
   cout<<"\nDetails of both person's as follows--"<<endl;
   p1.display();
   cout<<"\n";
   p2.display();
   cout<<"\nEnter name to search mobile number- ";
   cin>>searchname;
   cout<<"Mobile number of "<<searchname<<" is-
"<<p1.searchmobno(searchname)<<endl;

   cout<<"\nEnter mobile number to search details-";
   cin>>mobileno;
   cout<<"Persons details for mobile number"<<mobileno<<endl;
   p1.searchinfo(mobileno);
   p2.searchinfo(mobileno);

   return 0;
}
```

```
 //set c Q2)
 #include<iostream>
 using namespace std;
 class article
 {
```

```
private:
 int id;
 string name;
 int qty;
 float price;
public:
void acceptdetails()
{
    cout<<"Enter article id-";
    cin>>id;
    cout<<"Enter article name-";
    cin>>name;
    cout<<"Enter article quantity-";
    cin>>qty;
    cout<<"Enter article price-";
    cin>>price;
}
void display()
{
    cout<<"Article id-"<<id<<endl;
    cout<<"Article name-"<<name<<endl;
    cout<<"Article Quaantity-"<<qty<<endl;
    cout<<"Article price-"<<price<<endl;
}
bool priceexceed(float limit)
{
    return price>limit;
}
```

```
   bool qtyexceed(float limit)
   {
       return qty>limit;
   }
};
int main()
{
   article arr[100];
   int n,i;
   cout<<"Enter the number of articles-";
   cin>>n;
   for(i=0;i<n;i++)
    {
       cout<<"Enter details of article "<<i+1<<"-"<<endl;
       arr[i].acceptdetails();
    }
    int choice;
    do
      {
        cout<<"1.Details of article-"<<endl;
        cout<<"2.Details of article whose price exceed by 500-"<<endl;
        cout<<"3.Details of article whose quantity exceed by 50-"<<endl;
        cout<<"4.Exit-"<<endl;
        cout<<"Enter your choice-"<<endl;
        cin>>choice;
       switch (choice)
       {
       case 1:
            for(i=0;i<n;i++)
            arr[i].display();
        break;
        case 2:
            for(i=0;i<n;i++)
           if(arr[i].priceexceed(500))
            arr[i].display();
        break;
        case 3:
            for(i=0;i<n;i++)
            if(arr[i].qtyexceed(50))
            arr[i].display();
        break;
        case 4:
            cout<<"Existing......!"<<endl;
        }
    }while(choice!=4);
    return 0;
 }
```

Assignment 5

```
//set A Q1)
 #include <iostream>
using namespace std;
class number
{
    private:
    int x,y;
    public:
```

```
    number()                          //default constructor
      {
               x=8;
               y=5;
      }
     number(int a,int b )       //parameterized constructor
     {
               x=a;
               y=b;
     }
    int display()
    {
         cout<<"Maximum of two numbers is-";
        if(x>y)
          {   cout<<x;    }
        else
          {   cout<<y;   }
    }
};
int main()
{
   number ob1;
   number ob2(23,56);

   cout<<"default constructor-"<<endl;
   ob1.display();
   cout<<"\nparameterized constructor-"<<endl;
   ob2.display();
   return 0;
}
----------------------------------------------
//set A Q2)
#include<iostream>
using namespace std;
class intrest
{
    private:
      int amount;
      float time;
      float rate;
      public:
         intrest(int a,float t,float r=5.5)
         {
             amount=a;
             time =t;
             rate=r;
           }
         float intrestcal( )
         {
            return  (amount*time*rate)/100;
         }
};
int main()
{
       int amount;
    float time,rate;
    cout<<"Enter the principal amount:-";
    cin>>amount;
    cout<<"Enter the time duration:-";
    cin>>time;
    intrest t1(amount,time);
    cout<<"The intrest calculated with rate 5.5 is:-"<<t1.intrestcal()<<endl;
    return 0;
}
-----------------------------------------------
 //set B Q1)
#include<iostream>
using namespace std;
class BMI
{
  float weight,height;
  public:
  BMI()
  {
    weight=0;
    height=0;
  }
  BMI(float w,float h)
  {
    weight=w;
    height=h;
  }
  float BMIcal()
  {
    if(height==0)
    {
      cout<<"Height is zero ...cant calculate BMI..!"<<endl;
      return-1;
    }
    return weight/(height*height);
  }
  ~BMI()
  {
    cout<<"Memory is free....!"<<endl;
  }
};
int main()
{
    float weight,height;

    cout<<"Enter the weight in kg:-";
    cin>>weight;
    cout<<"Enter the height in meters:-";
    cin>>height;
    BMI ob1(weight,height);//dont forget this line in every constructor
    cout<<"BMI value for given data is:-"<<ob1.BMIcal()<<endl;
    return 0;
}
------------------------------------------------------------
//set B Q2)
#include<iostream>
```

```
using namespace std;
class player
{
  string name;
  int p_no;
  int matches;
  public:

  player(string nm,int no,int mt)
  {
   name=nm;
   p_no=no;
   matches=mt;
  }
  player()
  {
  }
  void acceptdata()
  {
    cout<<"Enter the name of player:-";
    cin>>name;
    cout<<"Enter the number of player:-";
    cin>>p_no;
    cout<<"Enter the number of matches played:-";
    cin>>matches;
     }
  void displaydata()
  {
    cout<<"Name of player:-"<<name<<endl;
    cout<<"Number of player:-"<<p_no<<endl;
    cout<<"Number of mathches:-"<<matches<<endl;
  }
};

int main()
{
  string name;
  int p_no;
  int matches;

  player p1("nilesh",101,421);
  p1.displaydata();

  player(name,p_no,matches);
  player p2;
  p2.acceptdata();
  p2.displaydata();
  return 0;
}
-----------------------------------------------
 //Set C Q1)
#include<iostream>
 using namespace std;
 class worker
 {
  string name;
  int days;
  float payrate;
  public:
  worker()  // deffault
  {
    name="";
    days=0;
    payrate=0;
  }
  worker(string nm,int dy,float pr)  // parameterized
  {
    name=nm;
    days=dy;
    payrate=pr;
  }
  worker( worker &x)  //copy
 {
   name=x.name;
   days=x.days;
   payrate=x.payrate;
 }

  float salarycal()
  {
    return (days*payrate);
  }

  void displaydata()
```

```
  {
   cout<<"Worker name is:-"<<name<<endl;
   cout<<"Number of days worked:-"<<days<<endl;
   cout<<"Payrate is:-"<<payrate<<endl;
   cout<<"salary is:-"<<salarycal()<<endl;
  }

};

int main()
{
   string name;
 int days;
 float payrate;
 cout<<"Information of worker 1:-"<<endl;
 worker w1("nilesh",25,400);
 w1.displaydata();

 cout<<"Information of worker 2:-"<<endl;
 worker w2("pranav",23,450);
 w2.displaydata();

 worker w3(w2);
 cout<<"Information of worker 3 (copied from worker 2):-"<<endl;
 w3.displaydata();
}
```

Assignmrnt 6

```
//set A Q1)
#include<iostream>
using namespace std;
class employee
{
   protected://keep it (protected) instead of (private) becaude these data
members are used in derived class
      string name;
      int code;
      string des;
   public:
      void getempdata()
      {
       cout<<"Enter employee name:-";
       cin>>name;
       cout<<"Enter employee code:-";
       cin>>code;
       cout<<"Enter employee designation:-";
       cin>>des;
      }
};
class manager:public employee
{
   private:
      int expyear;
      int salary;
```

```
    public:
      void getmanagerdata()
      {
        cout<<"Enter years of experience:-";
        cin>>expyear;
        cout<<"Enter salary:-";
        cin>>salary;
      }

      void displayinfo()
      {
        cout<<"Employee name-"<<name<<endl;
        cout<<"Employee code-"<<code<<endl;
        cout<<"Employee designaion-"<<des<<endl;
        cout<<"Experience-"<<expyear<<endl;
        cout<<"Salary-"<<salary<<endl;


      }
};
int main()
{
    manager m;
    m.getempdata();
    m.getmanagerdata();
    cout<<"\n";
    m.displayinfo();
    return 0;
}-------------------------------------------
```

```
//set B Q1)
#include<iostream>
using namespace std;
class vehicle
{
    private:
      int v_no;
      string o_name;
    public:
       void acceptdata()
       {
        cout<<"Enter vehicle number:-";
        cin>>v_no;
        cout<<"Enter owner name:-";
        cin>>o_name;
       }
       void displaydata()
       {
        cout<<"Vehicle number-"<<v_no<<endl;
        cout<<"Vehicle owner name-"<<o_name<<endl;
       }
};
class twowheeler:public vehicle
{
    private:
      string type;
    public:
     void accept()
     {
       //vehicle::acceptdata();       //for second case


       cout<<"Enter type of vehicle(scooter or bike)";
       cin>>type;
     }
     void display()
     {
      // vehicle::display();    //for second case


      cout<<"Vehicle type-"<<type<<endl;
     }
};
int main()
{
    int n,i;
    twowheeler arr[100]; // Fixed-size array to store details of vehicles
    cout <<"Enter the number of vehicles:";
    cin >> n;


    for (i=0;i<n;i++)
    {
     cout <<"Enter details for vehicle "<<i+1<<":"<<endl;
     arr[i].acceptdata();
     arr[i].accept();               // no need to write this in second case
    }
     cout <<"Vehicle details:"<<endl;
     for (i=0;i<n;i++)
    {
       cout <<"Details of vehicle "<<i+1<<":"<<endl;
       arr[i].displaydata();
       arr[i].display();              // no need to write this in second case
       cout<<endl;
    }

   return 0;
}
----------------------------------------
//Set B Q2)
#include<iostream>
using namespace std;
class person
{
  private:
    string firstname,lastname;
  public:
    void acceptdata1()
    {
      cout<<"Enter first name:-";
      cin>>firstname;
      cout<<"Enter last name:-";
      cin>>lastname;
    }
    void displaydata1()
    {
      cout<<"Person's first name is-"<<firstname<<endl;
      cout<<"Person's last name is-"<<lastname<<endl;
    }
};
class employee:public person
{
  private:
   int empid;
   string joindate;
  public:
   void acceptdata2()
    {
      person::acceptdata1();// from class person
      cout<<"Enter employee id:-";
      cin>>empid;
      cout<<"Enter joidate:-";
      cin>>joindate;
    }
    void displaydata2()
    {
      person::displaydata1();// from class person
      cout<<"Employee id is-"<<empid<<endl;
      cout<<"Employee's join date is-"<<joindate<<endl;
    }
};
class emp_sal:public employee
{
  private:
   int TA,DA,bonus;
  public:
   void accpetdata3()
  {
    employee::acceptdata2();// from class employe
    cout<<"Enter TA (Travel Allowance) amount:-";
    cin>>TA;
    cout<<"Enter DA (Dinner Allowance) amount:-";
    cin>>DA;
    cout<<"Enter bonus amount:-";
    cin>>bonus;
  }
  void displaydata3()
    {
      employee::displaydata2();// from class employee
      cout<<"Travel allowance:-"<<TA<<endl;
      cout<<"Dinner allowance:-"<<DA<<endl;
      cout<<"Bonus:-"<<bonus<<endl;

      int netsalary=TA+DA+bonus; //simple formula no need of separate function
      cout<<"Netsalary:-"<<netsalary<<endl;
    }
};
int main()
{
  emp_sal e1;
  cout<<"Accept details of employee-"<<endl;
  e1.accpetdata3();
  cout<<"\nHere the details......"<<endl;
  cout<<"\n";
  e1.displaydata3();
  return 0;
}
--------------------------------------------------------------
//set C Q1)....to long
```

Assignment 8

```
//set A
#include<iostream>
using namespace std;
class shape
{
   public:
    virtual double area()=0;
    virtual void display()=0;
};
class circle:public shape
{
  private:
    double r;//radius
  public:
    double area()override
    {
      return 3.14*r*r;
    }
    void display()override
    {
      cout<<"Circle"<<endl;
      cout<<"Radius is:-"<<r<<endl;
      cout<<"Area:-"<<area()<<endl;
    }


};
class rectangle:public shape
{
  private:
    double l,w;//length and width
  public:
    double area()override
    {
      return l*w;
    }
    void display()override
    {
      cout<<"Rectangle"<<endl;
      cout<<"Lenght:-"<<l<<endl;
      cout<<"Width:-"<<w<<endl;
      cout<<"Area:-"<<area()<<endl;
    }


};
class trapezoid:public shape
{
  private:
    double b1,b2,h;//base1 ,base2 and height
  public:
    double area()override
    {
      return ((b1 + b2) / 2) * h;
    }
    void display()override
    {
      cout<<"Rectangle"<<endl;
      cout<<"Base 1:-"<<b1<<endl;
      cout<<"Base 2:-"<<b2<<endl;
      cout<<"Height:-"<<h<<endl;
      cout<<"Area:-"<<area()<<endl;
    }
};

int main()
{
double radius,lenght,width,base1,base2,height;
cout<<"\n Enter the radius of circle:-";
cin>>radius;
circle A;
A.area();
A.display();
cout<<"_._._._._._._._._._._._._._._._._._._._._._._._._._._";

cout<<"\n Enter the Lenght and Width of Rectangle:-";
cin>>lenght>>width;
rectangle B;
```

```
B.area();
B.display();
cout<<"_._._._._._._._._._._._._._._._._._._._._._._._._";

cout<<"\n Enter two Bases and height of Trapezoid:-";
cin>>base1>>base2>>height;
trapezoid C;
C.area( );
C.display();
cout<<"_._._._._._._._._._._._._._._._._._._._._._._._._";

return 0;
}
---------------------------------------------------------------------------
//set B

#include<iostream>
using namespace std;
class student
{
  public:
   virtual void process()=0;
};
class engineering:public student
{
  public:
   void process()override
   {
      cout<<"Processing Engineering Student.....!"<<endl;
    }


};
class medicine:public student
{
   public:
    void process()override
    {
      cout<<"Processing Medical Student.....!"<<endl;
    }


};
class science:public student
{
   public:
    void process()override
    {
      cout<<"Processing Science Student.....!"<<endl;
    }
 };
 int main()
           {
            engineering E;
            medicine M;
            science S;
            student * arr[]={&E,&M,&S};
            int i;
```

```
          for(i=0;i<3;i++)
                    {
                      arr[i]->process();
                    }
 }
```

```
//set c
/*
#include<iostream>
using namespace std;
class media
{
  public:
  virtual void accept()=0;
   virtual void display()=0;
};
class book:public media
{
  private:
    string title;
    int pages,price;
  public:
    void accept()
    {
      cout<<"Enter the name of book:-"<<endl;
      cin>>title;
      cout<<"Number of pages:-"<<endl;
      cin>>pages;
      cout<<"Enter price of book:-"<<endl;
      cin>>price;
    }
    void display()
    {
      cout<<"Book details as follows:-"<<endl;
      cout<<"Name:- "<<title<<endl;
      cout<<"Number of pages:- "<<pages<<endl;
      cout<<"Price:- "<<price<<endl;
    }
};
class CD:public media
{
  private:
     float playtime;
  public:
    void accept()
    {
      cout<<"Enter the playtime of CD:-"<<endl;
      cin>>playtime;
    }
    void display()
    {
      cout<<"CD details as follows:-"<<endl;
      cout<<playtime<<"-is playtime of CD"<<endl;
    }
};
int main()
{
    book B;
    CD C;
    B.accept();
    C.accept();
    B.display();
    C.display();
}
```

Assignment 9

```
//set A q1)
/*
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main()
{
    string sourceFileName, targetFileName;

    // Input source and target file names
    cout << "Enter source file name: ";
    cin >> sourceFileName;
    cout << "Enter target file name: ";
    cin >> targetFileName;
```

```
// Open source file for reading
ifstream sourceFile(sourceFileName);

if (!sourceFile)
 {
    cout << "Error: Unable to open source file " << sourceFileName << endl;
    return 1;
}

// Open target file for writing
ofstream targetFile(targetFileName);

if (!targetFile)
{
    cout << "Error: Unable to open target file " << targetFileName << endl;
    return 1;
}

// Copy contents from source file to target file
char ch;
while (sourceFile.get(ch)) {
    targetFile.put(ch);
}

// Close files
sourceFile.close();
targetFile.close();
```

```
    cout << "File copied successfully." << endl;

    return 0;
}
*/
//set A Q2)
/*
#include <iostream>
#include <fstream>
#include <string>
#include <cctype>

using namespace std;

bool isVowel(char ch)
 {
    ch = tolower(ch);
    return (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u');
}

int main()
 {
    string fileName;
    char ch;
    int vowelCount = 0;

    // Input file name
    cout << "Enter file name: ";
    cin >> fileName;

    // Open file for reading
    ifstream file(fileName);

    if (!file)
    {
        cerr << "Error: Unable to open file " << fileName << endl;
        return 1;
    }

    // Read characters from file and count vowels
    while (file.get(ch)) {
        if (isalpha(ch) && isVowel(ch))
         {
            vowelCount++;
        }
    }

    // Close file
    file.close();

    cout << "Number of vowels in the file: " << vowelCount << endl;

    return 0;
}
*/
```

```
//set B Q1)
/*
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main()
{
    // Step 1: Create a file and store content
    ofstream outFile("sample.txt");
    if (!outFile) {
        cerr << "Error: Unable to create file." << endl;
        return 1;
    }

    string content = "A static member is shared by all objects of the class.PPPPP";
    outFile << content;
    outFile.close();
    cout << "File created and content stored successfully." << endl;

    // Step 2: Reopen the file and read its contents
    ifstream inFile("sample.txt");
    if (!inFile) {
        cerr << "Error: Unable to open file." << endl;
        return 1;
    }
    string fileContent;
    char ch;
    while (inFile.get(ch)) {
        fileContent += ch;
    }
    inFile.close();

    cout << "Content of the file: " << fileContent << endl;

    // Step 3: Count the number of times character 'P' is read
    int countP = 0;
    for (char c : fileContent) {
        if (toupper(c) == 'P') {
            countP++;
        }
    }

    cout << "Number of times character 'P' is read: " << countP << endl;

    return 0;
}
*/
//set B  Q2)
/*
#include <iostream>
#include <fstream>
#include <string>
```

```
using namespace std;

class Employee
{
private:
    string name;
    string designation;

public:
    // Member function to get data from the user
    void getData()
    {
        cout << "Enter name: ";
         cin>>name;
        cout << "Enter designation: ";
         cin>> designation;
    }

    // Member function to display data
    void putData()
    {
        cout << "Name: " << name << endl;
        cout << "Designation: " << designation << endl;
    }

    // Member function to write data to a file
    void writeToFile(ofstream& file)
    {
        file << "Name: " << name << endl;
        file << "Designation: " << designation << endl;
    }
};

int main() {
    // Create an object of the Employee class
    Employee emp;

    // Step 1: Open a file and write content using write()
    ofstream outFile("employee.txt");
    if (!outFile) {
        cerr << "Error: Unable to open file." << endl;
        return 1;
    }

    // Get data from the user
    emp.getData();

    // Write data to the file
    emp.writeToFile(outFile);

    // Close the file
    outFile.close();
    cout << "Data written to file successfully." << endl;

    // Step 2: Reopen the file in read mode and display contents on the screen
    ifstream inFile("employee.txt");
    if (!inFile) {
        cerr << "Error: Unable to open file." << endl;
        return 1;
    }

    string line;
    cout << "\nContents of the file:\n";
    while (getline(inFile, line)) {
        cout << line << endl;
    }

    // Close the file
    inFile.close();

    return 0;
}
```

ssignment 10

```
//Set A Q1)
#include <iostream>
using namespace std;
int main()
 {

        double x,y,z;
        cout << "Enter the first number: ";
        cin >>x;
        cout << "Enter the second number: ";
        cin >>y;
        try
          {
            if(y==0)
              {
                throw"\n Error Detected...dividing by 0";
              }
            else
              {
                z=x/y;
                cout<<"Answer of division is:-"<<z<<endl;
              }
          }
       catch(const char *error)
       {
         cout<<"Division is not possible..."<<error<<endl;
       }
     return 0;
}
---------------------------------------------------------------
//Set A Q2)
#include<iostream>
using namespace std;
void print(int stopnum)
{
   int i;
     for (i = 1; i < stopnum; i++)
```

```
      {
        cout << i << endl;
      }
     throw stopnum;
}
int main()
{
    try
     {
        print(15);
     }
     catch(int x)
     {
        cout<<"Caught an exception for value:-"<<x<<endl;
     }
     return 0;
}
```

```
//Set A Q3)
 #include<iostream>
using namespace std;
#include<cmath>
int main()
{
    int x;
    float y;
    cout<<"Enter a number:-"<<endl;
    cin>>x;
    try
```

```
          {
           if(x<0)
                throw "\n Number is negative.....!\n";
           else
             {
                y=sqrt(x);
                cout<<"Squareroot id "<<y<<endl;
             }
          }
    catch(const char * z)
          {
             cout<<"Error detected...!"<<z<<endl;
          }
 return 0;
 }
```

```
//Set B Q1)
#include<iostream>
using namespace std;
class calculator
{
    private:
     int x,y;
    public:
     void getdata()
     {
        cout<<"Enter first number:-";
        cin>>x;
```

```
   cout<<"Enter second number:-";
   cin>>y;
}

void calculate()
{
 try
   {
     if(x<=0||y<=0)    //condition
       {
           throw"Numbers should be positive....!";
       }
     if(x<=500||x>=1000||y<=500||y>=1000)     //condition
       {
         throw"Numbers should be in range 500-1000";
       }

    int sum=x+y;
    int prod=x*y;
    cout<<"sum is:-"<<sum<<endl;
    cout<<"product is:-"<<prod<<endl;
   }

catch(const char* error )
  {
     cout<<"Error detected...!"<<error<<endl;
   }
}
```

```
};
int main()
{
   calculator A;
   A.getdata();
   A.calculate();
   return 0;
}
```

```
//Set B Q2)
#include<iostream>
#include<string>
using namespace std;
void student(string name, int age, string year) {
   try {
      if (age < 18 || age > 25) {
         throw "Invalid age...!";
      }
      if (year != "FY" && year != "SY" && year != "TY") {
         throw "Invalid year...!";
      }
   } catch(const char* e) {
      cout << "Error detected: " << e << endl;
      return;
   }
   cout << "Student details:" << endl;
   cout << "Name: " << name << endl;
   cout << "Age: " << age << endl;
   cout << "Year: " << year << endl;
```

```
}
int main() {
   string name, year;
   int age;
   cout << "Enter student name: ";
   cin >> name;
   cout << "Enter student age: ";
   cin >> age;
   cout << "Enter student year (FY, SY, TY): ";
   cin >> year;
   student(name, age, year);
   return 0;
}
```

//Set C Q1)

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

long long factorial(int n)
 {
   if(n<0)
     {
     throw "\n Negative number....!";
     }
   if(n>20)
     {
     throw "\n Number is greater than 20..!";
     }
       long long result = 1;
```

```
    for(int i=2;i<=n;++i)
     {
         result *=i;
          cout<<"Factorial is: "<<result<<endl;
     }
}
int main()
{
    try
    {
        int num;
        cout<<"Enter a number to find its factorial:";
        cin>>num;
        factorial(num);
    }
    catch (const char* X)
    {
        cout<<"\n Error....."<< X<< endl;
    }
    return 0;
}
```

===

Assignment 11

```
//Set A Q1)
#include<iostream>
using namespace std;
template<class T>
void sort(T a[],int n)
```

```
{
    int i,j,temp;
    for(i=0;i<n-1;i++)
    {
        for(j=0;j<n-1;j++)
        {
            if(a[j]>a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
}
int main()
{   int n=8;
    int arr[]={32,71,12,45,26,80,53,33};
    sort(arr,n);
    cout<<"sorted array as follows:-\n";
    for(int i=0;i<n;i++)
    {
        cout<<arr[i]<<" ";
    }
}
```

```
//Set A Q2)
#include<iostream>
using namespace std;
```

```
template<class T>
T square(T num)
{
    T result=num*num;
    cout<<"Square of given number is "<<result<<endl;
    return 0;
}
int main()
{
    int num1;
    cout<<"Enter number 1:-";
    cin>>num1;
    square(num1);

    float num2;
    cout<<"Enter number 2:-";
    cin>>num2;
    square(num2);
    return 0;
}
```

```
//Set B Q1)
#include<iostream>
using namespace std;
template <class T>
class diffarrays
{
    private:
```

```
    T array[3];
    public:
    void setarray(T x,T y,T z)
    {
        array[0]=x;
        array[1]=y;
        array[2]=z;
    }
    void print()
    {
        for(int i=0;i<3;i++)
        {
            cout<<array[i]<<" "<<endl;
        }
    }
};
int main()
{
    diffarrays<int> A;
    diffarrays<float> B;
    diffarrays<char> C;

    A.setarray(1,7,5);
    B.setarray(1.56,7.4,5.15);
    C.setarray('X','Y','Z');

    cout<<"Integer array:-"<<endl;
    A.print();
    cout<<"Float array:-"<<endl;
    B.print();
    cout<<"Character array:-"<<endl;
    C.print();
    return 0;
}
-------------------------------------------------------------------
//Set C Q1)
#include <iostream>
using namespace std;

template<class T>// Template class for calculator
class Calculator
{
public:
    T add(T a, T b) // Function to add two numbers
     {
        return a + b;
     }

    T subtract(T a, T b) // Function to subtract two numbers
     {
        return a- b;
     }

    T multiply(T a, T b)  // Function to multiply two numbers
     {
        return a * b;
```

```
  }

   T divide(T a, T b) // Function to divide two numbers
        {
          if (b == 0)
           {
             cout << "Error! Division by zero!" << endl;
             return 0;
           }
          return a / b;
        }
};

int main()
{
  // Creating Calculator objects for different data types
  Calculator<int> intCalc;
  Calculator<float> floatCalc;
  Calculator<double> doubleCalc;

  // Integer calculations ------------------------------------------
  cout << "Integer Calculations:" << endl;

  cout << "Addition: " << intCalc.add(10, 5) << endl;

  cout << "Subtraction: " << intCalc.subtract(10, 5) << endl;

  cout << "Multiplication: " << intCalc.multiply(10, 5) << endl;
```

```
    cout << "Division: " << intCalc.divide(10, 5) << endl;

    // Floating-point calculations------------------------------------
    cout << "\nFloating-point Calculations:" << endl;

    cout << "Addition: " << floatCalc.add(10.5f, 5.5f) << endl;

    cout << "Subtraction: " << floatCalc.subtract(10.5f, 5.5f) << endl;

    cout << "Multiplication: " << floatCalc.multiply(10.5f, 5.5f) << endl;

    cout << "Division: " << floatCalc.divide(10.5f, 5.5f) << endl;

    // Double precision calculations---------------------------------------
    cout << "\nDouble Precision Calculations:" << endl;

    cout << "Addition: " << doubleCalc.add(10.123, 5.456) << endl;

    cout << "Subtraction: " << doubleCalc.subtract(10.123, 5.456) << endl;

    cout << "Multiplication: " << doubleCalc.multiply(10.123, 5.456) << endl;

    cout << "Division: " << doubleCalc.divide(10.123, 5.456) << endl;

    return 0;
}
*/
```

}

Data structure

Assignment 1---------------searching

```
//Set A Q1)-----------linear search num
/*
#include <stdio.h>
 int main()
{
   int  a[100], key, c, n, position;
   printf("Enter how many numbers you want to add in list:-");
      scanf("%d", &n);
   printf("Enter %d numbers\n", n);
      for (int i = 0; i < n; i++)
         scanf("%d", &a[i]);
   printf("Enter number to search\n");
      scanf("%d",&key);
   position = linear_search(a, n, key);//function calling
      if (position ==-1)
          printf("%d is not present in  list.\n", key);
      else
           printf("%d is present at location %d.\n", key, position+1);
   return 0;
}

int linear_search(int a[100], int n, int key)
{
    for (int i = 0 ;i < n ; i++ )
        {
         if (a[i] == key)
         return i;
        }
    return-1;
}
*/

//Set A Q2)-----------linear search char
 /*
 #include<stdio.h>
 #include<string.h>
 int linsearch(char names[][100], int n, char *key);
 int main()
 {
   char name[100][100];
   char key[100];
   int n,pos;
   printf("Enter how many names you want to add in list-\n");
   scanf("%d",&n);
   printf("Enter %d names\n",n);
   for(int i=0;i<n;i++)
   {
    scanf("%s",&name[i]);  // uuse %s to read string
   }
   printf("Enter name to search in list:-\n");
   scanf("%s",&key);
```

```
    pos=linsearch(name,n,key);


    if(pos==-1)
      printf("%s is not in list\n",key);
     else
     printf("%s is in list at position %d\n",key,pos+1);


     return 0;


  }


  int linsearch(char names[][100], int n, char *key)
  {
   for(int i=0;i<n;i++)
     {
         if(strcmp(names[i],key)==0)
         return i;
     }
     return-1;
 }
  */

//Set B Q1)-----------binary search num
/*
#include <stdio.h>
int BinarySearch(int key, int A[], int L, int H);

int main()
```

```
 {
   int a[100], key, c, n, position;
   printf("Enter how many numbers you want to add in list: ");
   scanf("%d", &n);

   printf("Enter %d numbers\n", n);
   for (int i = 0; i < n; i++)
      scanf("%d", &a[i]);

   printf("Enter number to search: ");
   scanf("%d", &key);

   position = BinarySearch(key, a, 0, n- 1);
   if (position ==-1)
      printf("%d is not present in the list.\n", key);
   else
      printf("%d is present at location %d.\n", key, position + 1);
   return 0;
}

int BinarySearch(int key, int A[], int L, int H)
 {
   if (L > H) // key does not exist
      return-1;
   int mid = (L + H) / 2;
   if (key == A[mid]) // base case
      return mid;
   else if (key < A[mid]) // recursive case I
```

```
        return BinarySearch(key, A, L, mid- 1);
    else // recursive case II
        return BinarySearch(key, A, mid + 1, H);
}
*/

//Set B Q2)-----------binary search char
/*
#include <stdio.h>
#include <string.h>

int BinarySearch(char names[][50], int n, char ele[50]);

int main() {
    int n, position;
    char key[50];

    printf("Enter how many names you want to add in list: ");
    scanf("%d", &n);

    char names[n][50]; // Declare names array after getting the value of n

    printf("Enter %d names in alphabetical order:\n", n);
    for (int i = 0; i < n; i++) {
        scanf("%s", names[i]);
    }

    printf("Enter key name to search: ");
    scanf("%s", key);

    position = BinarySearch(names, n, key);
    if (position ==-1)
        printf("%s is not present in the list.\n", key);
    else
        printf("%s is present at location %d.\n", key, position + 1);

    return 0;
}

int BinarySearch(char names[][50], int n, char ele[50]) {
    int L = 0, H = n- 1, mid;

    while (L <= H) {
        mid = (L + H) / 2;
        int result = strcmp(names[mid], ele);
        if (result == 0)
            return mid;
        else if (result < 0)
            L = mid + 1;
        else
            H = mid- 1;
    }
    return-1;
}

*/
```

```
//Set C)-----------binary search char
/*
#include <stdio.h>
#include <string.h>
#define MAX_CITIES 100
struct City
{
    char name[50];
    float area;
    int population;
};
void acceptCities(struct City cities[], int n);
void printCityInfo(struct City cities[], int n);

int linearSearch(struct City cities[], int n, const char key[]);
int binarySearch(struct City cities[], int n, const char key[]);

int main()
 {
    struct City cities[MAX_CITIES];
    int n, position;
    char key[50];

    printf("Enter the number of cities: ");
    scanf("%d", &n);

    acceptCities(cities, n);
```

```
printf("\nCities Information:\n");
printCityInfo(cities, n);

printf("\nEnter the city name to search: ");
scanf("%s", key);

// Perform linear search
position = linearSearch(cities, n, key);
if (position !=-1)
{
    printf("Linear Search Result:\n");
    printf("Area: %.2f sq km\n", cities[position].area);
    printf("Population: %d\n", cities[position].population);
}
 else
{
    printf("City not found using Linear Search.\n");
}

// Perform binary search (assuming cities are sorted by name)
position = binarySearch(cities, n, key);
if (position !=-1)
{
    printf("\nBinary Search Result:\n");
    printf("Area: %.2f sq km\n", cities[position].area);
    printf("Population: %d\n", cities[position].population);
}
```

```
    else
    {
        printf("City not found using Binary Search.\n");
    }

    return 0;
}

// Function to accept city information

void acceptCities(struct City cities[], int n)
 {
    for (int i = 0; i < n; i++)
    {
        printf("Enter name of city %d: ", i + 1);
        scanf("%s", cities[i].name);
        printf("Enter area of city %d (in sq km): ", i + 1);
        scanf("%f", &cities[i].area);
        printf("Enter population of city %d: ", i + 1);
        scanf("%d", &cities[i].population);
    }
}

// Function to print city information
void printCityInfo(struct City cities[], int n)
{
    for (int i = 0; i < n; i++)
    {
        printf("City Name: %s, Area: %.2f sq km, Population: %d\n", cities[i].name,
cities[i].area, cities[i].population);
    }
}

// Function to perform linear search
int linearSearch(struct City cities[], int n, const char key[])
{
    for (int i = 0; i < n; i++)
    {
        if (strcmp(cities[i].name, key) == 0)
        {
            return i; // City found, return its index
        }
    }
    return-1; // City not found
}

// Function to perform binary search
int binarySearch(struct City cities[], int n, const char key[])
{
    int low = 0, high = n- 1, mid;
    while (low <= high)
     {
        mid = (low + high) / 2;
        int cmp = strcmp(cities[mid].name, key);
        if (cmp == 0) {
            return mid; // City found, return its index
        } else if (cmp < 0) {
            low = mid + 1; // Search in the right half
        } else {
            high = mid- 1; // Search in the left half
        }
    }
    return-1; // City not found
}

*/
```

Assignment 2---------------treeee

```
//set A Q1    traversals
/*
#include <stdio.h>
#include<stdlib.h>
struct node
{
  int data;
  struct node*left;
  struct node*right;
};
struct node*createnode(int value)
{
    struct node*newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data=value;
    newnode->left=NULL;
    newnode->right=NULL;
    return newnode;
```

```
}
//-----------------------------------------------------
struct  node*insert(struct node* root,int value)
{
    if(root==NULL)
    {
        return createnode(value);
    }
    if(value < root->data)
    {
        root->left = insert(root->left,value);
    }
   else if(value > root->data)
    {
        root->right = insert(root->right,value);
    }
    return root;
}
 //-----------------------------------------------------
void inorder(struct node* root)
{
    if(root!=NULL)
    {
        inorder(root->left);
        printf("%d\t",root->data);
         inorder(root->right);


    }
```

```
}
//-----------------------------------------------------------
void preorder(struct node* root)
{
   if(root!=NULL)
   {
       printf("%d\t",root->data);
       preorder(root->left);
       preorder(root->right);


   }
}
//-------------------------------------------------------------
void postorder(struct node* root)
{
   if(root!=NULL)
   {
       postorder(root->left);
       postorder(root->right);
      printf("%d\t",root->data);
   }
}
//-------------------------------------------
int main()
{
 struct node* root=NULL;
 int choice,n;
 do
```

```
{
  printf("1.insert node\n");
  printf("2.inorder traversal\n");
  printf("3.preorder traversal\n");
  printf("4.postorder traversal\n");
  printf("5.exit....!\n");
  printf("enter your choice\n");
  scanf("%d",&choice);

  switch(choice)
  {
     case 1:
      printf("Enter value to insert:-");
      scanf("%d",&n);
      root=insert(root,n);
      break;
      case 2:
      printf("inorder traversal:-");
      inorder(root);
      printf("\n");
      break;
      case 3:
      printf("preorder traversal:-");
      preorder(root);
      printf("\n");
      break;
      case 4:
      printf("postorder traversal:-");
```

```
        postorder(root);
        printf("\n");
        break;
        case 5:
        printf("existing....!");
        break;
    }
 }
 while(choice!=5);
 return 0;
}
*/
//Set A Q2)------------------------------------  bst search
/*
#include <stdio.h>
#include <stdlib.h>

struct node
{
    int data;
    struct node *left;
    struct node *right;
};

struct node *createnode(int value)
{
    struct node *newnode = (struct node *)malloc(sizeof(struct node));
    newnode->data = value;
    newnode->left = NULL;
    newnode->right = NULL;
    return newnode;
}

struct node *insert(struct node *root, int value)
{
    if (root == NULL)
    {
        return createnode(value);
    }
    if (value < root->data)
    {
        root->left = insert(root->left, value);
    }
    else if (value > root->data)
    {
        root->right = insert(root->right, value);
    }
    return root;
}

struct node *search(struct node *root, int key)
{
    if (root == NULL || root->data == key)
    {
        return root;
    }
    if (root->data < key)
    {
        return search(root->right, key);
    }
    return search(root->left, key);
}

int main()
{
    struct node *root = NULL;
    root = insert(root, 50);
    insert(root, 30);
    insert(root, 20);
    insert(root, 40);
    insert(root, 70);
    insert(root, 60);
    insert(root, 80);

    int key = 60;
    struct node *result = search(root, key);
    if (result != NULL)
    {
        printf("%d found in the tree.\n", key);
    }
    else
    {
        printf("%d not found in the tree.\n", key);
    }
    key = 90;
    result = search(root, key);
    if (result != NULL)
    {
        printf("%d found in the tree.\n", key);
    }
    else
    {
        printf("%d not found in the tree.\n", key);
    }

    return 0;
}
*/
//Set A Q3)  heap sort
/*
#include <stdio.h>

void heapify(int arr[], int n, int i)
{
    int largest = i;
    int left = 2 * i + 1;
    int right = 2 * i + 2;

    if (left < n && arr[left] > arr[largest])
        largest = left;
    if (right < n && arr[right] > arr[largest])
        largest = right;

    if (largest != i)
     {
        int temp = arr[i];
        arr[i] = arr[largest];
        arr[largest] = temp;
        heapify(arr, n, largest);
    }
}

void heapSort(int arr[], int n)
{
    for (int i = n / 2- 1; i >= 0; i--)
        heapify(arr, n, i);

    for (int i = n- 1; i > 0; i--)
    {
        int temp = arr[0];
        arr[0] = arr[i];
        arr[i] = temp;
        heapify(arr, i, 0);
    }
}

void printArray(int arr[], int n)
 {
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");
}

int main()
 {
    int arr[] = {12, 11, 13, 5, 6, 7};
    int n = sizeof(arr) / sizeof(arr[0]);

    printf("Original array: \n");
    printArray(arr, n);

    heapSort(arr, n);

    printf("Sorted array: \n");
    printArray(arr, n);
    return 0;
}
*/
```

Assignment 3---------------------graph

```
// program to read graph as adjacency matrix
#include <stdio.h>

#define MAX_VERTICES 100

// Function to read graph as an adjacency matrix
```

```
void readGraph(int Matrix[MAX_VERTICES][MAX_VERTICES], int n )
 {
   printf("Enter the adjacency matrix for the graph:\n");
   for (int i = 0; i < n ; i++)
    {
      for (int j = 0; j < n ; j++)
      {
         scanf("%d", &Matrix[i][j]);
      }
   }
}

// Function to print the adjacency matrix
void printGraph(int Matrix[MAX_VERTICES][MAX_VERTICES], int n)
{
   printf("Adjacency matrix representation of the graph:\n");
   for (int i = 0; i < n; i++)
   {
      for (int j = 0; j < n; j++)
       {
         printf("%d ", Matrix[i][j]);
      }
      printf("\n");
   }
}

int main() {
   int n ;
```

```
    printf("Enter the number of vertices in the graph: ");
    scanf("%d", &n );

    int Matrix[MAX_VERTICES][MAX_VERTICES];

    readGraph(Matrix, n );
    printGraph(Matrix, n );

    return 0;
}
//-----------------------------------------------
//for adjacency list
#include <stdio.h>
#include <stdlib.h>

#define MAX_VERTICES 100

struct Node
{
    int vertex;
struct Node* next;
 };

struct Node* createNode(int v)
{
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->vertex = v;
    newNode->next = NULL;
     return newNode;
}

void addEdge(struct Node* adjList[], int src, int dest)
 {
    struct Node* newNode = createNode(dest);
    newNode->next = adjList[src];
    adjList[src] = newNode;
}

void printGraph(struct Node* adjList[], int numVertices)
 {
    for (int i = 0; i < numVertices; i++)
     {
        printf("Adjacency list of vertex %d:\n", i);
        struct Node* temp = adjList[i];
        while (temp)
         {
            printf("%d-> ", temp->vertex);
             temp = temp->next;
        }
        printf("NULL\n");
    }
}

int main()
{
    int numVertices, numEdges, src, dest;
    printf("Enter the number of vertices in the graph: ");
     scanf("%d", &numVertices);
    struct Node* adjList[MAX_VERTICES] = { NULL };
    printf("Enter the number of edges in the graph: ");
     scanf("%d", &numEdges);
    printf("Enter the edges (src dest):\n");
    for (int i = 0; i < numEdges; i++)
    {
        scanf("%d %d", &src, &dest);
         addEdge(adjList, src, dest);
    }
    printf("Graph created successfully.\n");
     printGraph(adjList, numVertices);
      return 0;
}
//-----------------------------------------------------
//dfs bfs
#include <stdio.h>
#include <stdbool.h>

#define MAX_VERTICES 100

// Function to perform Depth First Search (DFS)
void dfs(int adjMatrix[MAX_VERTICES][MAX_VERTICES], bool
visited[MAX_VERTICES], int vertex, int numVertices) {
    visited[vertex] = true;
    printf("Visited vertex: %d\n", vertex);
    for (int i = 0; i < numVertices; i++) {
        if (adjMatrix[vertex][i] && !visited[i]) {
            dfs(adjMatrix, visited, i, numVertices);
        }
    }
}

// Function to perform Breadth First Search (BFS)
void bfs(int adjMatrix[MAX_VERTICES][MAX_VERTICES], bool
visited[MAX_VERTICES], int startVertex, int numVertices) {
    int queue[MAX_VERTICES];
    int front = 0, rear = 0;

    visited[startVertex] = true;
    queue[rear++] = startVertex;

    while (front < rear) {
        int currentVertex = queue[front++];
        printf("Visited vertex: %d\n", currentVertex);

        for (int i = 0; i < numVertices; i++) {
            if (adjMatrix[currentVertex][i] && !visited[i]) {
                visited[i] = true;
                queue[rear++] = i;
            }
        }
    }
}
```

```
int main() {
    int numVertices, numEdges;

    printf("Enter the number of vertices in the graph: ");
    scanf("%d", &numVertices);

    int adjMatrix[MAX_VERTICES][MAX_VERTICES] = {0};
    bool visited[MAX_VERTICES] = {false};

    printf("Enter the number of edges in the graph: ");
    scanf("%d", &numEdges);

    printf("Enter the edges (src dest):\n");
    for (int i = 0; i < numEdges; i++) {
        int src, dest;
        scanf("%d %d", &src, &dest);
        adjMatrix[src][dest] = 1; // Assuming undirected graph
        adjMatrix[dest][src] = 1; // Assuming undirected graph
    }

    printf("DFS Traversal:\n");
    for (int i = 0; i < numVertices; i++) {
        if (!visited[i]) {
            dfs(adjMatrix, visited, i, numVertices);
        }
    }

    printf("\nBFS Traversal:\n");
    for (int i = 0; i < numVertices; i++) {
        visited[i] = false;
    }

    for (int i = 0; i < numVertices; i++) {
        if (!visited[i]) {
            bfs(adjMatrix, visited, i, numVertices);
        }
    }

    return 0;
}
```