****

Program : 1

Given that an EMPLOYEE class contains following members.Data members: Employee_Number, Employee_Name, Basic, DA, IT, Net_Sal. Member Functions: To read the data, To calculate Net_Sal and To print data members. Write a C++ program to read the data of N employees and compute Net_Sal of each employee. (DA= 52% of Basic and Income Tax (IT)=30% of the gross salary).

****

```
#include<iostream.h>
#include<conio.h>
class employee
{
     int empno;
     char empname[10];
     float basic,da,it,ns;
   public :
     void read()
          {
            cout<<"\nEnter employee name   : ";
            cin>>empname;
            cout<<"\nEnter employee number : ";
            cin>>empno;
            cout<<"\nEnter basic salary    : ";
            cin>>basic;
            da=basic*0.52;
            it=basic*0.30;
            ns=basic+da-it;
          }
     void display()
          {
            cout<<"\n   "<<empno;
            cout<<"    "<<empname;
            cout<<"    "<<basic;
            cout<<"   "<<da;
            cout<<"   "<<it;
            cout<<"   "<<ns;
          }
};
void main()
{
   clrscr();
   int n,i;
   employee e[20];
   cout<<"\nEnter the number of employees :  ";
   cin>>n;

   for(i=0;i<n;i++)
```

```
        e[i].read();
   cout<<" Emp_no   Name   Basic   DA   IT  Net Salary\n";
   cout<<"-----------------------------------------------\n";
   for(i=0;i<n;i++)
        e[i].display();
   cout<<"\n-----------------------------------------------";
   getch();
}
```

OUTPUT:

```
Enter the number of employees :  2

Enter employee name   : mina

Enter employee number : 84

Enter basic salary    : 500

Enter employee name   : anju

Enter employee number : 69

Enter basic salary    : 600

 Emp_no   Name    Basic   DA   IT  Net Salary
-----------------------------------------------

  84        mina    500   260     150   610
  69        anju    600   312     180   732
-----------------------------------------------
```

****

Program : 2

Define a STUDENT class with USN, Name and Marks in 3 tests of subject. Declare an array of 1O STUDENT objects. Using appropriate functions, find the average of two better marks for each student. Print the USN, Name and average marks of all the students.

****

```
#include<iostream.h>
#include<conio.h>
class student
 {
    int usn,m1,m2,m3;
    char name[20];
    float avg;
  public :
    void read()
        {
         cout<<"\n\nEnter the student name      : ";
         cin>>name;
         cout<<"\nEnter the student number    : ";
         cin>>usn;
         cout<<"\nEnter Marks in test I,II and III : ";
         cin>>m1>>m2>>m3;
         if(m1<=m2 && m1<=m3)
           avg=(m2+m3)/2.0;
         if(m2<=m1 && m2<=m3)
           avg=(m1+m3)/2.0;
         if(m3<=m1 && m3<=m2)
           avg=(m1+m2)/2.0;
        }
    void display()
        {
         cout<<usn<<"   "<<name<<"   "<<"  "
            <<m1<<"    "<<m2<<"    "<<m3<<"      "<<avg<<endl;
        }
 };
void main()
{
   student s[10];
   clrscr();
   int n=0,i;
   char ch='y';

   while(ch=='y')
    {
        s[n].read();
        cout<<"\nDo u want to add any more records(y/n)";
        cin>>ch;
```

```
        n++;
    }
        cout<<"\n STUDENT RECORDS\n";
        cout<<"USN   Name   TEST1  TEST2  TEST3   Average  \n";
        cout<<"----  ------   -----  -----  -----   -------  \n";
        for(i=0;i<n;i++)
        s[i].display();
  getch();
}
```

OUTPUT:

Enter the student name : sana

Enter the student number : 100

Enter Marks in test I,II and III : 45 46 47

Do u want to add any more records(y/n)y

Enter the student name : sita

Enter the student number : 101

Enter Marks in test I,II and III : 45 50 49

Do u want to add any more records(y/n)n

STUDENT RECORDS

USN	Name	TEST1	TEST2	TEST3	Average
100	sana	45	46	47	46.5
101	sita	45	50	49	49.5

****

Program : 3

Write a C++ program to create a class called COMPLEX and implement the following overloading functions ADD that return a COMPLEX number.

i. ADD(a , s2) - where a is an integer (real part) and s2 is a complex number.

ii. ADD(s1, s2) - where s1 and s2 are complex numbers.

****

```
#include<iostream.h>
#include<conio.h>
class complex
{
     float real,img;
   public :
     void read()
          {
            cout<<"Enter real and imaginary  part : ";
            cin>>real>>img;
          }
     complex add(complex c2)
          {
            complex t;
            t.real=real+c2.real;
            t.img=img+c2.img;
            return t;
          }
     complex add(int n)
          {
            complex t;
            t.real=real+n;
            t.img=img;
            return t;
          }
     void display()
          {
            if(img<0)
              cout<<"\nThe value after adding is : "<<real<<img<<"i";
            else
              cout<<"\nThe value after adding is : "<<real<<"+"<<img<<"i";
          }
};

void main()
{
   clrscr();
   complex c1,c2,c3;
   int n,ch;
   do
   {
```

```
        cout<<"\n\n\n1.Add two complex number.";
        cout<<"\n2.Add a complex number & an integer.";
        cout<<"\n3.Exit.";
        cout<<"\nEnter your choice  : ";
        cin>>ch;



    switch(ch)
        {
           case 1:
             cout<<"\nFirst Complex number :\n";
             c1.read();
             cout<<"\nSecond Complex number :\n";
             c2.read();
             c3=c1.add(c2);
             c3.display();
             break;
          case 2:
             c1.read();
             cout<<"\nEnter any integer  : ";
             cin>>n;
             c3=c1.add(n);
             c3.display();
             break;
          case 3:
             break;
    }
  }while(ch!=3);
  getch();
}
```

OUTPUT:

```
1.Add two complex number.
2.Add a complex number & an integer.
3.Exit.
Enter your choice  : 1

First Complex number :
Enter real and imaginary  part :  2 6

Second Complex number :
Enter real and imaginary  part : 23 9

The value after adding is : 25+15i
```

1.Add two complex number.
2.Add a complex number & an integer.
3.Exit.
Enter your choice : 2
Enter real and imaginary part : 3 8

Enter any integer : 5

The value after adding is : 8+8i

1.Add two complex number.
2.Add a complex number & an integer.
3.Exit.
Enter your choice : 3

****

Program : 4

Write a C++ program to create a class called LIST (linked list) with member functions to insert an element at the front as well as to delete an element: from the front of the list. Demonstrate all the function after creating a list object.

****

```
#include<iostream.h>
#include<conio.h>
class node
{
   public :
     int info;
     node *ptr;
};
class list
{
     node *head;
   public :
     list()
          {
            head=NULL;
          }
     void insfro();
     void delfro();
     void display();
};
void list::insfro()
{
   node *newnode;
   newnode=new node;
   int ele;
   cout<<"\nEnter number to be inserted : ";
   cin>>ele;
   newnode->info=ele;
   newnode->ptr=head;
   head=newnode;
}
void list::delfro()
{
   if(head==NULL)
          cout<<"\nList is empty.";
   else
     {
          head=head->ptr;
          cout<<"\nThe first element has been deleted.";
     }
}
void list::display()
```

```
{
   node *t;
   if(head==NULL)
     cout<<"\nList is empty.";
   else
   {
         t=head;
         cout<<"\nThe content of list are : ";
         while(t!=NULL)
    {
            cout<<t->info<<"  ";
            t=t->ptr;
         }
   }
}

void main()
{
   clrscr();
   int ch;
   list l;
   do
   {
         cout<<"\n1.Insert front.\n2.Delete front.";
         cout<<"\n3.Display.\n4.exit.";
         cout<<"\nEnter your choice : ";
         cin>>ch;
         switch (ch)
    {
            case 1 :
              l.insfro();
              break;
            case 2:
              l.delfro();
              break;
            case 3:
              l.display();
              break;
            case 4:;

         }
   }while(ch!=4);
   getch();
}
```

OUTPUT:

```
1.Insert front.
2.Delete front.
3.Display.
4.exit.
Enter your choice : 1

Enter number to be inserted : 23

1.Insert front.
2.Delete front.
3.Display.
4.exit.
Enter your choice : 1

Enter number to be inserted : 34

1.Insert front.
2.Delete front.
3.Display.
4.exit.
Enter your choice : 3

The content of list are : 23 34
1.Insert front.
2.Delete front.
3.Display.
4.exit.
Enter your choice : 2

The first element has been deleted.
1.Insert front.
2.Delete front.
3.Display.
4.exit.
Enter your choice : 3

The content of list are : 23
1.Insert front.
2.Delete front.
3.Display.
4.exit.
Enter your choice : 4
```

****

Program : 5

Write a C++ program to create a template function for Quick Sort and demonstrate sorting of integers and doubles.

****

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
#define max 50

template <class type>
class qsor
{
    private :
          type a[max];
    public :
          int size;
          void read();
          void mainsort();
          void sort(int,int);
          void display();
};

template <class type> void qsor <type>::read()
  {
    cout<<"\nENTER THE ELEMENTS TO SORT :\n";
    for (int i=0;i<size;i++)
         cin>>a[i];
   }
template <class type> void qsor <type>::mainsort(void)
   {
    cout<<"\nEnter the number of elements : ";
    cin>>size;
    read();
    sort(0,size-1);
    cout<<"\nThe array after sorting is : ";
    display();
   }

template <class type> void qsor <type>::sort(int lb,int ub)
 {
   int i,j,flag=1;
   type temp,key;
   if (lb<ub)
    {
         i=lb+1;
         j=ub;
```

```
        key=a[lb];
        while (flag)
         {
           while(a[i]<key && i<j)
            i++;
           while (a[j]>key)
            j--;
           if (i<j)
            {
              temp=a[i];
              a[i]=a[j];
              a[j]=temp;
            }
           else
            flag=0;
         }
     a[lb]=a[j];
     a[j]=key;
     sort(lb,j-1);
     sort(j+1,ub);
   }
  }
template <class type> void qsor <type>::display()
   {
    for (int i=0;i<size;i++)
        cout<<a[i]<<" ";
    cout<<"\n";
   }
void main()
 {
   clrscr();
   int ch;

    cout<<"\n1.Sorting integers.";
    cout<<"\n2.Sorting doubles.";
    cout<<"\n3.Exit.";
   do {
        cout<<"\nEnter your choice : ";
        cin>>ch;
        switch (ch)
        {
         case 1:  qsor<int>s1;
                  s1.mainsort();
                  break;

         case 2:  qsor<double>s2;
                  s2.mainsort();
                  break;
```

```
        case 3:  exit(0);

        default : cout<<"\nInvalid choice.";
    }
   }while(ch!=3);
  getch();
 }
```

OUTPUT:

```
1.Sorting integers.
2.Sorting doubles.
3.Exit.
Enter your choice : 1

Enter the number of elements : 5

ENTER THE ELEMENTS TO SORT :
10
8
-9
3
15

The array after sorting is : -9 3 8 10 15

Enter your choice : 2
Enter the number of elements : 5
ENTER THE ELEMENTS TO SORT :
1.04
23.78
2.9
5.8
14.999

The array after sorting is : 1.04 2.9 5.8 14.999 23.78
Enter your choice : 3
```

****

Program : 6

:Write C++ program to create a class called STACK using array of integers. Implement the following operations by overloading the operators + and --.Also display the status and contents of the stack after each operation, by overloading the operator <<.

i) s1=s1+element; where s1 is a object of the class STACK and element is an integer to be pushed on top of the stack

ii) s1=s1--;where s1 is a object of the class STACK, -- operator pops the element. Handle the STACK empty and STACK full conditions.

****

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
#define max 20
class stack
 {
    int st[max];
  public :
    int top;
    stack()
        { top=-1; }
    void operator +(int);
    void operator -(int);
    friend ostream& operator <<(ostream& , stack&);
 };

void stack::operator +(int ele)
 {
  st[++top]=ele;
 }

void stack::operator -(int)
 {
  --top;
 }

ostream& operator << (ostream& out,stack& s)
 {
   for(int i=0;i<=s.top;i++)
        out<<" "<<s.st[i];
   return out;
  }

void main()
 {
  clrscr();
  stack s;
```

```
   int ch,ele;

   cout<<"\nOPERATIONS ON STACK";
   cout<<"\n1.Push.\n2.Pop.\n3.Display.\n4.Exit.";
   do
    {
        cout<<"\nEnter your choice  : ";
        cin>>ch;
        switch(ch)
         {
           case 1:
            if(s.top==max-1)
                cout<<"\nOverflow.";
            else
                {
                 cout<<"\nEnter the element to insert : ";
                 cin>>ele;
                 s+ele;
                }
            break;
           case 2:
            if(s.top==-1)
                cout<<"\nUnderflow.";
            else
                s-1;
            break;
           case 3:
            if(s.top==-1)
                cout<<"\nThe stack is empty.";
            else
                {
                 cout<<"\nThe content of stack are : ";
                 cout<<s;
                }
            break;
           case 4: exit(0);
           default : cout<<"invalid choice";
         }
    }while(ch!=4);
   getch();

}
```

OUTPUT:

```
OPERATIONS ON STACK
1.Push.
2.Pop.
3.Display.
4.Exit.
Enter your choice  : 1

Enter the element to insert : 10

Enter your choice  : 1

Enter the element to insert : 23

Enter your choice  : 1

Enter the element to insert : 33

Enter your choice  : 1

Enter the element to insert : 11

Enter your choice  : 3

The content of stack are :  10 23 33 11
Enter your choice  : 2

Enter your choice  : 3

The content of stack are :  10 23 33
Enter your choice  : 2

Enter your choice  : 3

The content of stack are :  10 23
Enter your choice  : 5
invalid choice
Enter your choice  : 4
```

****

Program No : 7

Write a C++ program to create a class called DATE. Accept two valid dates in the form dd/mm/yy. Implement the following operations by overloading the operators + and -. After every operation display the results by overloading the operator <<.

1.)no_of_days=d1-d2; where d1 and d2 are DATE objects d1>=d2 and no_of_days is an integer.

2)d2=d1-no_of_days; where d1 is a DATE object and no_of_days is an integer.

****

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
class DATE
{
private:
        int days,month,year;
public:
        void getdate( );
        int operator-(DATE);
        DATE operator+(int );
};

/****--METHOD TO ACCEPT TWO VALID DATES--------****/
void DATE::getdate( )
{
  cout<<"Enter  DATE(dd mm yy):";
  VALID:  cin>>days>>month>>year;
        if((month==2) && (days>29))
        {
          cout<<"\n Wrong Input!!!";
          goto VALID;
        }
        if((month>12)|| (days>31))
        {
          cout<<"\n Wrong Input:";
          goto VALID;
        }
        if((month==4|| month==6||month==9||month== 11) &&(days>30))
        {
          cout<<"Wrong Input:";
          goto VALID;
        }
        if((year%4)!=0 &&(month==2) &&(days>28))
        {
          cout<<"Wrong Input ..";
          goto VALID;
```

```
        }
}

/******METHOD TO OVERLOAD THE – OPERATOR *****/
int DATE::operator-(DATE d2)
{
  int I,nod1,nod2,nody,leapcount,no_of_days;
  nod1=nod2=leapcount=0;
  for(I=1;I<month;I++)
  {
    if(I==1||I==3||I==5||I==7||I==8||I==10||I==12)
    {
        nod1=nod1+31;
    }
    else if(I==2)
    nod1=nod1+28;
    else
    nod1=nod1+30;
  }
  nod1=nod1+days;
  for(I=1;I<d2.month;I++)
  {
    if(I==1||I==3||I==5||I==7||I==8||I==10||I==12)
    {
        nod2=nod2+31;
    }
    else if(I==2)
    nod2=nod2+28;
    else
    nod2=nod2+30;
  }
  nod2=nod2+d2.days;
  nody=(year-d2.year)*365;
  for(I=d2.year;I<year;I++)
  {
    if((I%4)==0)
    {
        leapcount=leapcount+1;
    }
  }
  int y4=year-d2.year;
  while(y4>400)
  {
    leapcount=leapcount+1;
    y4=y4-400;
  }
  if((month>2) &&(year%4)==0)
  {
```

```
    leapcount=leapcount+1;
   }
   if((d2.month>2) && (d2.year%4)== 0)
  {
    leapcount=leapcount-1;
  }
  no_of_days=nody+nod1-nod2+leapcount;
  if(no_of_days>0)
  {
    cout<<"\n Difference between two DATES is :";
    return(no_of_days);
  }
  else
  {
    return(no_of_days);
  }
}

/******METHOD TO OVERLOAD THE OPERATOR +  **********/
DATE DATE::operator+(int nd)
{
  DATE dd3;
  while(nd>365)
  {
   year=year+1;
    nd=nd-365;
  }
  while(nd>30)
  {

if(month==1||month==3||month==5||month==7||month==8||month==10||month==12)
    {
        nd=nd-31;
        month=month+1;
    }
  else if(month==2)
    {
        nd=nd-28;
        month=month+1;
    }
  else
    {
        nd=nd-30;
        month=month+1;
    }
  if(month>12)
    {
        year=year+1;
```

```
            month=month+1;
        }
    } // end of while
    days=days+nd;
    if(days>30)
    {
        if(month==4||month==6||month==9||month==11)
        {
            month=month+1;
            days=days-30;
        }
        else if(month==2)
        {
            month=month+1;
            days=days-28;
        }
        else if(days>31)
        {
            month=month+1;
            days=days-31;
        }
    } // end of if
    cout<<"\n The DATE after addition is :";
    cout<<days<<"/"<<month<<"/"<<year<<endl;
    return(dd3);
} // end of data function.
/*******************MAIN FUNCTION**************/
void main( )
{
    DATE d1,d2;
    int res,num;
    clrscr( );
    cout<<"\nDATE1 should be greater than DATE2\n";
    d1.getdate( );
    d2.getdate( );
    cout<<d1-d2;
    cout<<"\nEnter the number of days to be added to DATE1";
    cin>>num;
    d2=d1+num;
    getch( );
}
```

OUTPUT:

```
Enter day, month and year : 10 1 2003
Enter day, month and year : 25 12 2002
No of days = 16
```

Enter no of days to be removed from the first date : 25
The date before 25 days is : 16/12/2002

Enter day, month and year : 29 2 2003
Enter day, month and year : 29 2 2000

One or both of the dates are invalid.

****

Program No : 8

Write a C++ program to create a class called MATRIX using a two- dimensional array of integers. Implement the following operations by overloading the operator== which checks the compatibility of two matrices to be added and subtracted. Perform the addition and subtraction by overloading the operators + and -. Display the result by overloading the operator <<.

if(m1==m2) Then m3=m1-m2 and m4=m1+m2
else display error

****

```
#include<iostream.h>
#include<conio.h>
class matrix
 {
  private :
    int m[5][5];
    int r,c;
  public :
    matrix()
        { r=0;c=0; }
    void read();
    friend int operator ==(matrix,matrix);
    friend matrix operator +(matrix,matrix);
    friend matrix operator -(matrix,matrix);
    friend ostream& operator << (ostream&,matrix&);
 };

void matrix::read()
 {
  cout<<"\nEnter the order of the matrix (r&c) : ";
  cin>>r>>c;
  cout<<"\nEnter the elements : \n";
  for(int i=0;i<r;i++)
   for(int j=0;j<c;j++)
    cin>>m[i][j];
 }

int operator ==(matrix a,matrix b)
 {
  if((a.r==b.r) && (a.c==b.c))
    return 1;
  else
    return 0;
 }

matrix operator +(matrix a,matrix b)
 {
```

```
    matrix t;
    for(int i=0;i<a.r;i++)
     for(int j=0;j<a.c;j++)
       t.m[i][j]=a.m[i][j]+b.m[i][j];
    t.r=a.r;
    t.c=a.c;
    return t;
  }

matrix operator -(matrix a,matrix b)
  {
    matrix t;
    for(int i=0;i<a.r;i++)
     for(int j=0;j<a.c;j++)
       t.m[i][j]=a.m[i][j]-b.m[i][j];
    t.r=a.r;
    t.c=a.c;
    return t;
  }

ostream& operator << (ostream& os,matrix& a)
  {
    for(int i=0;i<a.r;i++)
      {
            for(int j=0;j<a.c;j++)
              os<<a.m[i][j]<<" ";
            cout<<"\n";
      }
    return os;
  }

void main()
  {
    clrscr();
    matrix a,b,c,d;
    a.read();
    b.read();
    if(a==b)
      {
           c=a+b;
           cout<<"\nThe sum of two matrices is : \n";
           cout<<c;
           d=a-b;
           cout<<"\nThe difference of two matrices is : \n";
           cout<<d;
      }
    else
      cout<<"\nThe matrix addition & subtraction is not possible.";
```

```
   getch();
}
```

OUTPUT:

Enter the order of the matrix (r&c) : 2 2

Enter the elements :
12
13
14
15

Enter the order of the matrix (r&c) : 2 2

Enter the elements :
1
2
3
4

The sum of two matrices is :
13 15
17 19

The difference of two matrices is :
11 11
11 11

****

Program No : 9

Program to create a class called OCTAL which has the characteristics of an octal number. Implement the following operations by writing an appropriate constructor and an overloaded operator +.

I . OCTAL h=x; where x is an integer.

II. int y=h+k; where h is an octal object and k is an integer.

Display the OCTAL results by overloading the operator <<. Also Display the values of h and y.

****

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

class OCTAL
{
   private:  int num;
   public :
            {
                 cout<<"Enter a number"<<endl;
                 cin>>num;
            }
            OCTAL(int x)
            {
                 num = x;
            }
            friend int operator + (OCTAL oct1, OCTAL oct2);
            friend ostream& operator << (ostream&, OCTAL&);

};
int operator + (OCTAL oct1,OCTAL oct2)
{
   return (oct1.num + oct2.num);
}
ostream& operator << (ostream& out, OCTAL& oct)
{
   out<<"Number in base 8 is "<<setbase(8)<< oct.num<<endl;
   return out;
}
void main()
{  OCTAL h(8);
  int x,y,k;

  clrscr();
  cout<<h<<endl;
```

```
  cout<<"Enter a integer"<<endl;
  cin>>k;
  cout<<"Its equivalent octal value is "<<k<<endl;

  y = h + k;
  cout<<"\nSo,y = "<<y;
}
```

OUTPUT:

```
Number in base 8 is 10
Enter a integer
20
Its equivalent octal value is 24

So,y = 34


Number in base 8 is 10

Enter a integer
012
Its equivalent octal value is 12

So,y = 22
```

****

Program No : 10

Program to create a class called QUEUE with the member functions to add and to delete an element from the queue.
Using these functions, implement a queue of integer and double. Demonstrate the operations by displaying the contents of the queue after every operation.

****

```
#include <iostream.h>
#include <iomanip.h>
#include <conio.h>
template <class T>
class queue
{
  public :
            queue(int max = 4);
            ~queue()
            {  delete [] pqptr; };

            void insert(const T&) ;

            int qfull() const
            { return rear == nsize ; }

            int qempty() const
            { return (front >= rear || rear == 0) ; }

            T del() ;

            void display() ;

  private :
                    int nsize;
                    int rear;
                    int front;
                    T *pqptr;
} ;

template <class T>

queue <T> :: queue(int nx)
{
   nsize = nx > 0 && nx < 100 ? nx : 10 ;
   front = 0 ;
   rear = 0 ;
   pqptr = new T[nsize];
} ;
```

```
template <class T>
void queue <T> :: insert(const T &nitem)
{
  if (qfull())
  {
        cout << "Queue overflow ....." << endl ;
        getch();
        return ;
  }

    pqptr[++rear] = nitem ;
        return ;
};

template <class T>

T queue <T> :: del(void)
{
        T p ;

        if (qempty())
        {
          cout << "Queue empty ....." << endl;
          getch();
          return p ;
        }

        p = pqptr[front++];
        cout << "front " << front << " rear = " << rear ;
        return p ;
} ;



template <class T>

void queue <T> :: display()
{
  if (qempty())
  {
        cout << "Empty queue....." << endl ;
        getch();
        return;
  }

  cout << "front =  " << front << " rear =  "  << rear << endl ;
  int i = front;
```

```
  cout << "\nFront ";

  for (i = i+1 ; i <= rear ; i++)
  {
        cout << " <- "  <<  pqptr[i] ;
  }

  cout << "<-Rear";

  return ;
} ;

int menu()
{
 int m ;

 cout << endl << setw(10) << "1. Insert";
 cout << endl << setw(10) << "2. Delete"; ;
 cout << endl << setw(11) << "3. Display";
 cout << endl << setw(8) <<  "4. Exit";
 cout << endl << endl << setw(9) << "Enter your choice : " ;
 cin >> m ;
 return m;
};

void main(void)
{
 int menu();
 int choice1 = 0 , choice2 = 0 , i ;
 double d;
 queue <int> iq(4) ;
 queue <double> dq(4) ;
 clrscr();
 while (choice1 != 3)
 {
        choice1 = 0 ;
        cout << endl << setw(10) << " 1. Integer Queue";
        cout << endl << setw(10) << " 2. Double Queue" ;
        cout << endl << setw(7) <<  " 3. Exit";
        cout << endl << endl << setw(10) << "Enter your choice : " ;
        cin >> choice1 ;

        switch(choice1)
        {
          case 1 : {
                    choice2 = 0 ;
                    while (choice2 != 4)
                    {
```

```
                choice2 = menu();

                switch(choice2)
                {
                 case 1 : {  cout << endl
                          << "Enter the element to be inserted : " ;
                              cin  >> i ;

                              iq.insert(i);
                              break;
                          }
                 case 2 : {
                              iq.del();
                              break;
                          }

                     case 3 : {
                              iq.display();
                              break;
                          }

                          case 4 : break ;

                          default : break;
                                }
                }

                 break ;
              }                          // End of CASE-1


      case 2 : {
                choice2 = 0 ;

                while (choice2 != 4)
                {
                      choice2 = menu();
                      switch(choice2)
                      {
                          case 1 : {
                        cout << endl << "Enter the element to be inserted : " ;
                                    cin >> d ;

                                    dq.insert(d);
                                    break;
                                }

                          case 2 : {
```

```
                                        dq.del();
                                        break;
                                    }

                                case 3 : {
                                        dq.display();
                                        break;
                                    }

                                case 4 : break ;

                                default : break;
                            }
                        }

                        break ;
                }

                case 3 : break ;

                default : break ;

            }

        }
}
```

OUTPUT:

```
 1. Integer Queue
 2. Double Queue
 3. Exit
Enter your choice : 1
 1. Insert
 2. Delete
 3. Display
 4. Exit
Enter your choice : 1

Enter the element to be inserted : 10

Enter your choice : 1

Enter the element to be inserted : 1

Enter your choice : 20
```

1. Insert
2. Delete
3. Display
4. Exit
Enter your choice : 1
Enter the element to be inserted : 30
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice : 3
front = 0 rear = 3

Front <- 10 <- 20 <- 30<-Rear

Enter your choice : 2
front 1 rear = 3

Enter your choice : 2
front 2 rear = 3

Enter your choice : 2
front 3 rear = 3

Enter your choice : 2
Queue empty .....
Enter your choice : 4

Enter your choice : 2

Enter your choice : 1

Enter the element to be inserted : 12.45

Enter your choice : 1

Enter the element to be inserted : 34.56

Enter your choice : 1

Enter the element to be inserted : 45.67

Enter your choice :3
front = 0 rear = 3

Front <- 12.45 <- 34.56 <- 45.67<-Rear

Enter your choice : 2
front 1 rear = 3

Enter your choice : 2
front 2 rear = 3

Enter your choice : 2
front 3 rear = 3
1. Insert
2. Delete
3. Display
4. Exit

Enter your choice : 2
Queue empty .....

1. Insert
2. Delete
3. Display
4. Exit
Enter your choice : 4
1. Integer Queue
2. Double Queue
3. Exit
Enter your choice : 3

****

Program No : 11

Program to create a class called DLIST (doubly linked list) with the member functions to insert a node at a specified position and delete a node at a specified position of the list. Demonstrate the operations by displaying the contents of the list after every operation.

****

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

class node
{
 public:  int  num;
          node *next;
          node *prev;

          void display()
          {
               cout<<"\nNode Element: "<<num;
          }
};

class dlist
{
 public:  node *first;
          node *last;
          int totalnodes;

          dlist()
          {
               first=NULL;
               last=NULL;
               totalnodes=0;
          }

          void add()
          {
               node *temp;
               temp =new node;

               if (totalnodes==0)
               {
                  first=temp;
                  last=temp;
                  last->next=NULL;
```

```
            first->prev=NULL;
        }
        else
        {
            temp->prev=last;
            last->next=temp;
            last=temp;
            temp->next=NULL;
        }

        cout<<"\nEnter the value of the node : ";
        cin>>temp->num;

        totalnodes++;
    }

    void display()
    {
        cout<<"\nThe Contents of the Doubly Linked list:\n";
        node* temp;
        temp=first;

        if(temp==NULL)
        {
            cout<<"The List is EMPTY\n";
        }
        else
        {
            cout<<"Head=>";
            while(temp!=NULL)
            {
                cout<<temp->num<<"<=>";
                temp=temp->next;
            }
            cout<<"Null\n";
        }
    }

    void remove()
    {
        int n;
        node* temp;
        node *curr;
        node *nex;
        int found=0;
        int f=0;
        temp=first;
        if(temp==NULL)
```

```
        {
          cout<<"\n***The List is EMPTY***\n";
        }
        else
        {
          cout<<"\nEnter the value of the element to be deleted\n";
          cin>>n;
          temp=first;
          if (temp->num==n)
          {
                first=temp->next;
                first->prev=NULL;
                temp->next=NULL;
                temp->prev=NULL;
                f=1;
        cout<<"The element "<<temp->num<<" is deleted from the list\n";
                totalnodes--;
          }
else
          {
                while((temp!=NULL)&&(found!=1))
                {
                  if(temp->num==n)
                  {
                     found=1;
                     cout<<"\nElement "<<temp->num<<" found"<<endl;
                   }

                   if(!found)
                   {
                         temp=temp->next;
                          curr=temp->prev;
                          nex=temp->next;
                   }

                }

           }

           if (found==1)
           {
                cout<<"And it is deleted from the list\n";

                curr->next=nex;
                nex->prev=curr;
                temp->next=NULL;
                temp->prev=NULL;
```

```
                    totalnodes--;
                    cout<<"Remaining nodes :"<<totalnodes<<endl;
                }
                else if ((found==0)&&(f==0))
                {
                    cout<<"\nElement not found\n";
                }
            }

        }
};

void main()
{
        dlist d;
        int rpt = 1;
        char c;

        clrscr();

        while(rpt)
        {
          cout<<"\n1.Display List\n2.Insert element in the list";
          cout<<"\n3.delete element from the list\n4.Exit\n";

          fflush;
          cout<<"Enter your choice..."<<endl;
          cin>>c;

      switch(c)
          {
              case '1': d.display();
                        break;
              case '2': d.add();
                        break;
              case '3': d.remove();
                        break;
              case '4': exit(0);

          }
          cout<<"\nDo you want to continue(Type 0 or 1)?"<<endl;
          cin>>rpt;
        }
}
```

OUTPUT:

```
1.Display List
2.Insert element in the list
3.delete element from the list
4.Exit
Enter your choice...
1

The Contents of the Doubly Linked list:
The List is EMPTY
Do you want to continue(Type 0 or 1)?
1

1.Display List
2.Insert element in the list
3.delete element from the list
4.Exit
Enter your choice...
2

Enter the value of the node : 10
Enter your choice...
2

Enter the value of the node : 20
Do you want to continue(Type 0 or 1)?
1

Enter your choice...
2

Enter the value of the node : 30
Do you want to continue(Type 0 or 1)?
1

Enter your choice...
4
```

****

Program No : 12

Program to create a class called STUDENT with data members USN, Name and Age. Using inheritance, create the classes UGSTUDENT and PGSTUDENT having fields a semester, fees and stipend. Enter the data for at least 5 students. Find the average age for all UG and PG students separately.

****

```
#include <conio.h>
#include <stdio.h>
#include <iostream.h>
#define MAX 2

class STUDENT
{
  private: char *name;
           int regno;
           int age;
  public:
           void putdata();
           int  getage()
           {
                return(age);
           }
};

void STUDENT::putdata()
{
        cout<<"Enter the name    : ";
        cin >> name;
        cout<<"Enter the register number : ";
        cin >> regno;
        cout<<"Enter the age     : ";
        cin >> age;
}

class UGstudent:public STUDENT
{
  private:      int semister;
                float fees;
                float stipend;
  public:
                void putugdata();
                int  getsemister()
                {
                        return(semister);
                }
};
```

```
void UGstudent::putugdata()
{
        STUDENT::putdata();
        cout<<"\nEnter the semester(1,2,3,4,5,6,7,8) :";
        cin >> semister;
        cout<<"Enter the fees    : ";
        cin >> fees;

    cout<<"Enter the stipend : ";
        cin >> stipend;
}

class PGstudent:public STUDENT
{
  private:      int semister;
                float fees;
                float stipend;
  public:
                void putpgdata();
                int  getsemister()
                {
                        return(semister);
                }
};

void PGstudent::putpgdata()
{
        STUDENT::putdata();
        cout<<"\nEnter the semester(1,2,3) : ";
        cin >> semister;
        cout<<"Enter the fees    : ";
        cin >> fees;
        cout<<"Enter the stipend : ";
        cin >> stipend;
}

void main()
{
        UGstudent ugstudent[MAX];
        PGstudent pgstudent[MAX];
        clrscr();
        cout<<"Enter the details of UG students"<<endl;
        for(int i=0;i<MAX;i++)
        {
            ugstudent[i].putugdata();
        }
```

```
int total1=0,total2=0,total3=0,total4=0,total5=0,
   total6=0,total7=0,total8=0;


int tco1=0,tco2=0,tco3=0,tco4=0,tco5=0,
   tco6=0,tco7=0,tco8=0;
int tempsem=0,tempage=0;

for(i=0;i<MAX;i++)
{
        tempsem=ugstudent[i].getsemister();

        switch (tempsem)
        {
                case 1 : tempage=ugstudent[i].getage();
                         total1 += tempage;
                         ++tco1;
                         break;

                 case 2 : tempage=ugstudent[i].getage();
                         total2 += tempage;
                         ++tco2;
                         break;
                case 3 : tempage=ugstudent[i].getage();
                         total3 += tempage;
                         ++tco3;
                         break;
                case 4 : tempage=ugstudent[i].getage();
                         total4 += tempage;
                         ++tco4;
                         break;
                case 5 : tempage=ugstudent[i].getage();
                         total5 += tempage;
                         ++tco5;
                         break;
                case 6 : tempage=ugstudent[i].getage();
                         total6 += tempage;
                         ++tco6;
                         break;
                case 7 : tempage=ugstudent[i].getage();
                         total7 += tempage;
                         ++tco7;
                         break;
                case 8 : tempage=ugstudent[i].getage();
                         total8 += tempage;
                         ++tco8;
                         break;
         }
```

```
}
clrscr();
cout<<"\nEnter the details of PG students"<<endl;

for(i=0;i<MAX;i++)
{
        pgstudent[i].putpgdata();
}

int tot1=0,tot2=0,tot3=0;

int tc1=0,tc2=0,tc3=0;
int temsem=0,temage=0;

for(i=0;i<MAX;i++)
{
        temsem=pgstudent[i].getsemister();

        switch (temsem)
        {
           case 1 : temage=pgstudent[i].getage();
                    tot1 += temage;
                    ++tc1;
                    break;
           case 2 : temage=pgstudent[i].getage();
                    tot2 += temage;
                    ++tc2;
                    break;

   case 3 : temage=pgstudent[i].getage();
                    tot3 += temage;
                    ++tc3;
                    break;
        }
}
clrscr();

cout<<"\nUG STUDENT'S INFORMATION..."<<endl;
if (tco1 != 0)
{
        cout <<"\nAverage age of 1st semester student : ";
        cout <<(float)total1/tco1;
}
if (tco2 != 0)
{
        cout <<"\nAverage age of 2nd semester student : ";
        cout <<(float)total2/tco2;
}
```

```
        if (tco3 != 0)
        {
                cout <<"\nAverage age of 3rd semester student : ";
                cout <<(float)total3/tco3;
        }
        if (tco4 != 0)
        {
                cout <<"\nAverage age of 4th semester student : ";
                cout <<(float)total4/tco4;
        }
        if (tco5 != 0)
        {
                cout <<"\nAverage age of 5th semester student : ";
                cout <<(float)total5/tco5;
        }
        if (tco6 != 0)
        {
                cout <<"\nAverage age of 6th semester student : ";
                cout <<(float)total6/tco6;
        }
        if (tco7 != 0)
        {
                cout <<"\nAverage age of 7th semester student : ";
                cout <<(float)total7/tco7;
        }
        if (tco8 != 0)
        {
                cout <<"\nAverage age of 8th semester student : ";
                cout <<(float)total8/tco8;
        }
        cout<<"\n\nPG STUDENT'S INFORMATION...";
        if (tc1 != 0)
        {
                cout <<"\nAverage age of 1st semester student : ";
                cout <<(float)tot1/tc1;
        }



    if (tc2 != 0)
        {
                cout <<"\nAverage age of 2nd semester student : ";
                cout <<(float)tot2/tc2;
        }
        if (tc3 != 0)
        {
```

```
                cout <<"\nAverage age of 3rd semester student : ";
                cout <<(float)tot3/tc3;
        }
        getch();
}
```

OUTPUT:

```
Enter the details of UG students
Enter the name    : Ravi
Enter the register number : 111
Enter the age     : 20

Enter the semester(1,2,3,4,5,6,7,8) :5
Enter the fees    : 5000
Enter the stipend : 500
Enter the name    : Sharat
Enter the register number : 122
Enter the age     : 19

Enter the semester(1,2,3,4,5,6,7,8) :3
Enter the fees    : 6000
Enter the stipend : 400

Enter the details of PG students
Enter the name    : Priya
Enter the register number : 789
Enter the age     : 25

Enter the semester(1,2,3) : 3
Enter the fees    : 15000
Enter the stipend : 2000
Enter the name    : Rahul
Enter the register number : 777
Enter the age     : 24

Enter the semester(1,2,3) : 3
Enter the fees    : 15000
Enter the stipend : 1000

UG STUDENT'S INFORMATION...
Average age of 3rd semester student : 19
Average age of 5th semester student : 20

PG STUDENT'S INFORMATION...
Average age of 3rd semester student : 24.
```

****

Program No : 13

Program to create a class called STRING and implement the following operations. Display the results after every operation by overloading the operator <<.

I . STRING s1= "VTU"

II . STRING s2= "BELGAUM"

III. STRING s3= s1+s2; (use copy operator).

****

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

class string
{
  private:
        char str[20];

  public:

        string()
        {
          strcpy(str," ");
        }


        string(char *s)
        {
          strcpy(str, s);
        }

        string  operator +(string ss)
        {
              string temp;
              strcpy(temp.str, str);
              strcat(temp.str, ss.str);

              return temp;
        }

        string operator <<(string ss)
        {
              cout<< ss.str;
              return 0;
        }

};
```

```
void main()
{
  clrscr();
  string s1 = "VTU";
  string s2 = "BELGAUM";
  cout<< "String 1: ";
  s1 << s1<<"\n";


  cout<< "String 2: ";
  s2 <<s2<<"\n";

  string s3 = s1 + s2;

  cout<< "String 3: ";
  s3 << s3;

  getch();

}
```

OUTPUT:

```
String 1: VTU
String 2: BELGAUM
String 3: VTUBELGAUM
```

****

Program No : 14

write a C++ program to create a class called BIN_TREE (binary tree) with member functions to perform inorder, preorder and postorder traversal. Create a BIN_TREE object and demonstrate the traversals.

****

```
#include <iostream.h>
#include <conio.h>
#define true 1
#define false 0
#define maxnode 10

struct binary_tree
{
    int item;
    struct binary_tree *left;
    struct binary_tree *right;
};
typedef struct binary_tree node;

class bin_tree
{
  protected:
          node element[maxnode];
  public:
          void create(node *record1,node *record2,int num);
          void inorder(node *record);
          void preorder(node *record);
          void postorder(node *record);
};
void bin_tree::create(node *base,node *newnode,int num)
{
   while(1)
   {
     if(num == base->item)
     {
          cout<<"Duplicate number!!\n";
          break;
     }

     if(num > base->item)
     {
          if(base->right == NULL)
          {
            base->right = newnode;
            base = base->right;
```

```
            break;
          }
          else
          {
            base = base->right;
            continue;
          }
      }


      else if(num < base->item)
      {
          if(base->left == NULL)
          {
                base->left = newnode;
                base = base->left;
                break;
          }
          else
          {
                 base = base->left;
                 continue;
          }
      }

    }
}

void bin_tree::inorder(node *base)
{
  if(base != NULL)
  {
    inorder(base->left);
    cout<<base->item<<" ";
    inorder(base->right);
    return;
  }
}



void bin_tree::preorder(node *base)
{
  if(base != NULL)
  {
    cout<<base->item<<" ";
    preorder(base->left);
```

```
    preorder(base->right);
    return;
  }
}


void bin_tree::postorder(node *base)
{
  if(base!=NULL)
  {
    postorder(base->left);
    postorder(base->right);
    cout<<base->item<<" ";
    return;
  }
}









void main()
{
  int no;
  char ch = 'y';
  clrscr();
  bin_tree tree;
  node *btree, *newNode, *temp;
  btree = new node;
  btree = NULL;
  while(ch == 'y')
  {
        cout<<"Enter the number: ";
        cin>>no;
        newNode = new node;
        newNode->item = no;
        newNode->left = NULL;
        newNode->right = NULL;

        if(btree == NULL)
        {
          *btree = *newNode;
           btree = newNode;
           temp = btree;
        }
        else
```

```
        {
           tree.create(temp,newNode,no);
        }
        cout<<"Do you want to enter more numbers(y/n)? :";
        cin>>ch;
    }
    *temp = *btree;

    cout<<"\nINORDER TRAVERSAL\n";
    cout<<"*****************\n";
    tree.inorder(temp);

    cout<<"\n";
    cout<<"\nPREORDER TRAVERSAL\n";
    cout<<"*****************\n";
    tree.preorder(temp);

    cout<<"\n";
    cout<<"\nPOSTORDER TRAVERSAL\n";
    cout<<"*****************\n";
    tree.postorder(temp);

    getch();
}
```

OUTPUT:

```
Enter the number: 10
Do you want to enter more numbers(y/n)? :y
Enter the number: 15
Do you want to enter more numbers(y/n)? :y
Enter the number: 5
Do you want to enter more numbers(y/n)? :y
Enter the number: 4
Do you want to enter more numbers(y/n)? :y
Enter the number: 18
Do you want to enter more numbers(y/n)? :n

INORDER TRAVERSAL
*****************
4 5 10 15 18

PREORDER TRAVERSAL
*****************
10 5 4 15 18
```

POSTORDER TRAVERSAL
****
4 5 18 15 10

****

Program No : 15
Write a C++ program to create a class called EXPRESSION. Using appropriate member functions to convert a valid infix expression into a postfix form. Display the infix and postfix expression.
****

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

class EXPRESSION
{
  private:  char infix[20];
            char postfix[20];
            char stack[20];
            int top;

  public :  EXPRESSION()
            {
                top = -1;
                strset(postfix,'\0');
                strset(stack, '\0');
            }
            void getINFIX(void);
            void infix_to_postfix(void);
            void push (char symb);
            int  pop ();
            int  preced(char symb);
            void putPOSTFIX(void);
};

void EXPRESSION :: getINFIX(void)
{
   cout<<"Enter a valid infix expression"<<endl;
   cin>>infix;
}
void EXPRESSION ::infix_to_postfix (void)
{
        int length;
        int index = 0, pos = 0;
        char symbol,temp;
```

```
        length = strlen(infix);
        push ('#');

        while (index < length)
        {
                symbol = infix[index];

                switch (symbol)
                {
                        case '(' : push (symbol);
                                   break;

            case ')' : temp = pop();
                                   while (temp != '(')
                                   {
                                        postfix[pos] = temp;
                                        pos++;
                                        temp = pop();
                                   }
                                   break;
                        case '+' :
                        case '-' :
                        case '*' :
                        case '/' :
                        case '^' : while (preced(stack[top]) >= preced(symbol))
                                   {
                                     temp = pop();
                                     postfix[pos] = temp;
                                     pos++;
                                   }
                                   push (symbol);
                                   break;
                        default : postfix[pos++] = symbol;
                                  break;
                }
                index++;
        }
        while (top > 0)
        {
          temp = pop();
          postfix[pos++] = temp;
        }
        return;
}

void EXPRESSION::push (char symb)
{
        top = top + 1;
```

```
        stack[top] = symb;
}
int EXPRESSION::pop ()
{
        char symb;
        symb = stack[top];
        top = top - 1;
        return(symb);
}

int EXPRESSION :: preced(char symb)
{
        int p;
        switch (symb)
        {
                case '^' : p = 3;
                        break;
                case '*' :
                case '/' : p = 2;
                        break;
                case '+' :
                case '-' : p = 1;
                        break;


        case ')' :
                case '(' : p = 0;
                        break;
                case '#' : p = -1;
                        break;
        }
        return (p);
}

void EXPRESSION :: putPOSTFIX(void)
{
   cout<<"Postfix expression is ";
   cout<<postfix<<endl;
}

void main ()
{
        EXPRESSION e;
        clrscr();

        e.getINFIX();
        e.infix_to_postfix ();
        e.putPOSTFIX();
```

```
}
```

OUTPUT:

```
Enter a valid infix expression
(a+b)-(c*d)^(e/f)
Postfix expression is ab+cd*ef/^-

Enter a valid infix expression
a*b-(c^d)+e/f
Postfix expression is ab*cd^-ef/+
```