Course Title: Object Oriented Programming with C++ and Java			
Course Code: P15IS36	**Semester** : III	**L- T- P-H** : 3-0-2-5	**Credits:**4

<u>Contents</u>

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

2. Define a STUDENT class with USN, Name, and Marks in 3 tests of a subject. Declare an array of 10 STUDENT objects. Using appropriate functions, find the average of the two better marks for each student. Print the USN, Name and the average marks of all the students.

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

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 functions after creating a list object.

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

6. Write a C++ program to create a class called STACK using an array of integers. Implement the following operations by overloading the operators '+' and '-':
   (i) s1 = s1 + element; where s1 is an object of the class STACK and element is an integer to be pushed on the top of the stack
   (ii) s1 = s1- ; where s1 is an object of the class STACK. '-'operator pops the element.
   Handle the STACK empty and full conditions. Also display the contents of the stack after each operation, by overloading the << operator.

7. Create a class called MATRIX using 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 + and – operators respectively. Display the results by overloading the operator <<.
   If (m1==m2) then m3 = m1+m2 and m4 = m1-m2 else display error.

8. Write a C++ 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 result by overloading the operator <<. Also display the values of h and y.

9. Write a C++ program to create a class called QUEUE with member functions to add an element and to delete an element from the queue. Using the member functions, implement a queue of integers and double. Demonstrate the operations by displaying the contents of the queue after every operation.

10. Write a C++ program to create a class called STUDENT with data members USN, Name and Age. Using inheritance, create the classes UGSTUDENT and PGSTUDENT having fields as Semester, Fees and Stipend. Enter the data for at least 5 students. Find the semester-wise average age for all UG and PG students separately.

11. Write a C++ program to create a class called DLIST (doubly Linked List) with member functions to insert a node at a specified position and delete a node from a specified position of the list. Demonstrate the operations by displaying the content of the list after every operation.

12. Write a C++ program to create a class called EXPRESSION. Using appropriate member functions convert a given valid Infix expression into postfix form. Display the infix and postfix expressions.

13. Write a C++ 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 = “ISE”
    (ii) STRING s2 = “PESCE”
    (iii) STRING s3 = s1 + s2 (Use copy constructor)

14. Write a C++ program to create a class called BIN_TREE (Binary Tree) with member functions to perform in-order, preorder and post-order traversals. Create a BIN_TREE object and demonstrate the traversals.

15. Write a JAVA Program to demonstrate Constructor overloading and Method Overloading.

Course Outcomes

The student is able to

CO1. Develop application programs using non-object oriented features of C++.

CO2. Demonstrate automatic initialization/de-initialization of objects and Inheritance using C++.

CO3. Apply the concepts of virtual functions and operator overloading for a given problem.

CO4. Implement the concepts of stream handling, templates and exception handling.

CO5. Develop programs using object oriented programming language like Java for a given scenario.

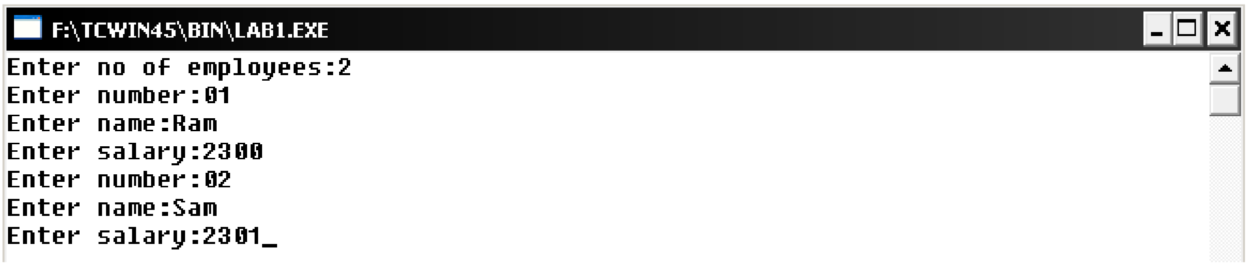
1. Given that an EMPLOYEE class contains the following members:
Data Members: Employee_Number, Employee_Name, Basic, DA, IT, Net_Sal
Member Functions: to read data, to calculate Net_Sal and to print data members
Write a C++ program to read data on N employees and compute the Net_Sal of each employee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

Program:

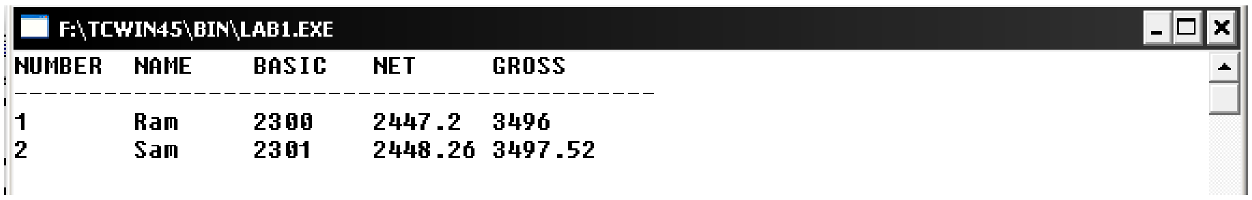
```
#include <iostream.h>
#include <conio.h>
class employee
{
	char name[10];
	int no;
	float basic,da,it,ns,gs;
	public:
	void input()
	{
		cout <<"Enter number:";
		cin >> no;
		cout <<"Enter name:";
		cin >> name; cout <<"Enter salary:";
		cin >> basic;
	}
	void calculate()
	{
	da = 0.52 * basic;
	gs = da + basic;
	it = 0.3 * gs;
	ns = gs - it;
	}
	void output()
	{
		cout<<no<<'\t'<<name<<'\t'<<basic<<'\t'<<ns<<'\t'<<gs <<'\n';
	}
};
void main()
{
	employee emp[20];
	int n,i;
	clrscr();
	cout << "Enter no of employees:";
```

```
        cin >> n;
        for(i=0;i<n;i++)
        {
                emp[i].input();
                emp[i].calculate();
        }
        clrscr();
        cout<<"NUMBER"<<'\t'<<"NAME"<<'\t'<<"BASIC"<<'\t'<<"NET"<<'\t'<<"GROSS"
        << "\n";
        cout<<"------------------------------------------\n";
        for(i=0;i<n;i++)
        {
                emp[i].output();
        }
        getch();
}
```

OUTPUT:



```
F:\TCWIN45\BIN\LAB1.EXE
Enter no of employees:2
Enter number:01
Enter name:Ram
Enter salary:2300
Enter number:02
Enter name:Sam
Enter salary:2301_
```



```
F:\TCWIN45\BIN\LAB1.EXE
NUMBER  NAME    BASIC   NET     GROSS
-------------------------------------------
1       Ram     2300    2447.2  3496
2       Sam     2301    2448.26 3497.52
```

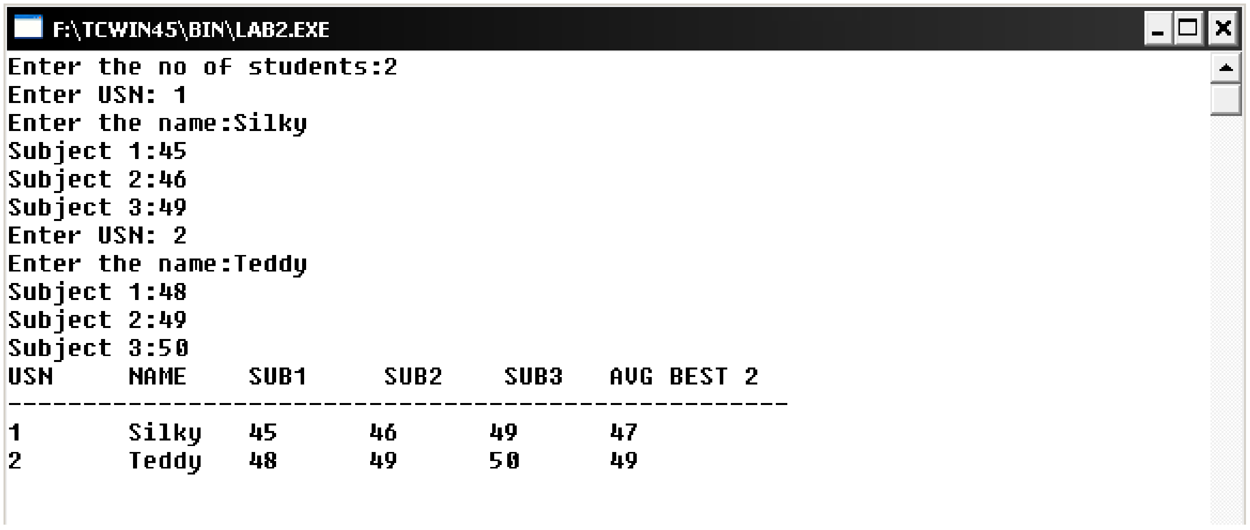
2. Define a STUDENT class with USN, Name, and Marks in 3 tests of a subject. Declare an array of 10 STUDENT objects. Using appropriate functions, find the average of the two better marks for each student. Print the USN, Name and the average marks of all the students.

Program:

```
#include <iostream.h>
#include <conio.h>
#define small(a,b) (a<b)?a:b
class student
{
        int no;
        char name[10];
        int s1,s2,s3;
        float avg;
        public :
        void input()
        {
                cout<<"Enter USN: " ;
                cin>>no;
                cout<<"Enter the name:";
                cin>>name;
                cout<<"Subject 1:";
                cin>>s1;
                cout<<"Subject 2:";
                cin>>s2;
                cout<<"Subject 3:";
                cin>>s3;
        }
        void calculate()
        {
                int s,t;
                t =small(s2,s3);
                s=small(s1,t);
                avg = (s1+s2+s3-s)/2 ;
        }
        void output()
        {
                cout<<no<<'\t'<<name<<'\t'<<s1<<'\t'<<s2<<'\t'<<s3<<'\t'<<avg<<"\n";
        }
};
```

```
void main()
{
	student e[10];
	int n,i;
	clrscr();

	cout<<"Enter the no of students:";
	cin>>n;
	for(i=0;i<n;i++)
	{
		e[i].input();
		e[i].calculate();
	}
	cout<<"USN"<<'\t'<<"NAME"<<'\t'<<"SUB1"<<'\t'<<"SUB2"<<'\t'<<"SUB3"
	<<'\t'<<"AVG BEST 2"<<endl;
	cout<<"-----------------------------------------------------\n";
	for(i=0;i<n;i++)
	{
		e[i].output();
	}
	getch();
}
```

OUTPUT:



```
Enter the no of students:2
Enter USN: 1
Enter the name:Silky
Subject 1:45
Subject 2:46
Subject 3:49
Enter USN: 2
Enter the name:Teddy
Subject 1:48
Subject 2:49
Subject 3:50
USN     NAME    SUB1     SUB2    SUB3   AVG BEST 2
----------------------------------------------------
1       Silky   45      46      49      47
2       Teddy   48      49      50      49
```

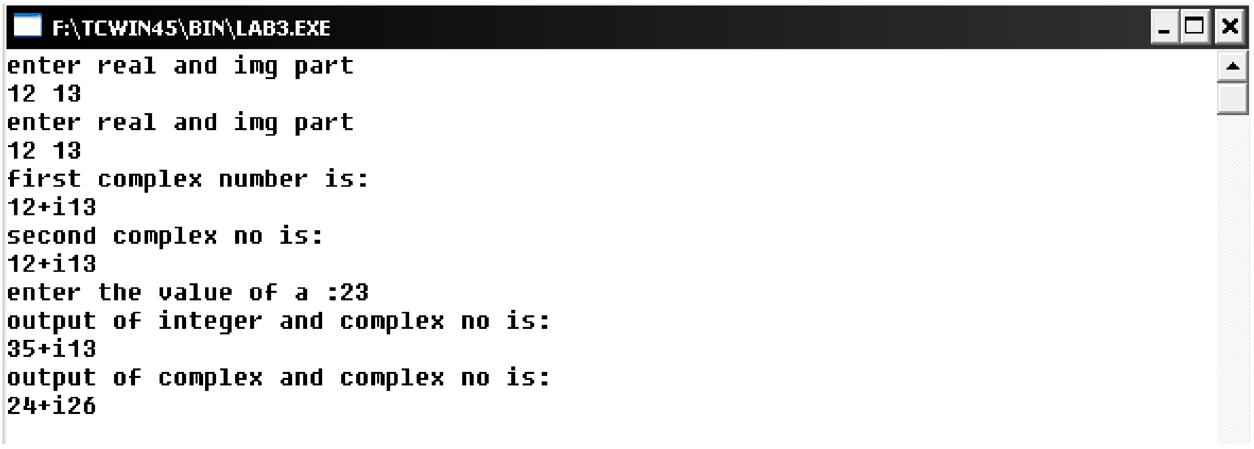
3. Write a C++ program to create a class called COMPLEX and implement the following overloading functions ADD that return a complex number:
(i) ADD (a, s2) – where 'a' is an integer (real part) and s2 is a complex number
(ii) ADD (s1, s2) – where s1 and s2 are complex numbers

Program:

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
class complex
{
        int real;
        int img;
        public:
        void input()
        {
                cout << "enter real and img part" << '\n';
                cin >> real >> img;
        }
        void output()
        {
                if(img<0)
                        cout << real<< img << "i" << '\n';
                else
                        cout << real << "+"<< "i" << img << '\n';
        }
        friend complex add(int,complex);
        friend complex add(complex,complex);
};
complex add(int a , complex s2)
{
        complex temp;
        temp.real = s2.real + a;
        temp.img = s2.img; return temp;
}
complex add(complex s1, complex s2)
{
        complex s3;
        s3.real = s1.real + s2.real;
        s3.img = s1.img + s2.img;
        return s3;
}
```

```
void main()
{
        complex s1,s2,sum1,sum2;
        int a;
        clrscr();
        s1.input();
        s2.input();
        cout << "first complex number is:"<< '\n';
        s1.output();
        cout << "second complex no is:"<< '\n';
        s2.output();
        cout << "enter the value of a :";
        cin >> a;
        sum1 = add(a,s2);
        sum2 = add(s1,s2);
        cout << "output of integer and complex no is:"<< '\n';
        sum1.output();
        cout << "output of complex and complex no is:"<< '\n';
        sum2.output();
        getch();
}
```

OUTPUT:



F:\TCWIN45\BIN\LAB3.EXE

```
enter real and img part
12 13
enter real and img part
12 13
first complex number is:
12+i13
second complex no is:
12+i13
enter the value of a :23
output of integer and complex no is:
35+i13
output of complex and complex no is:
24+i26
```

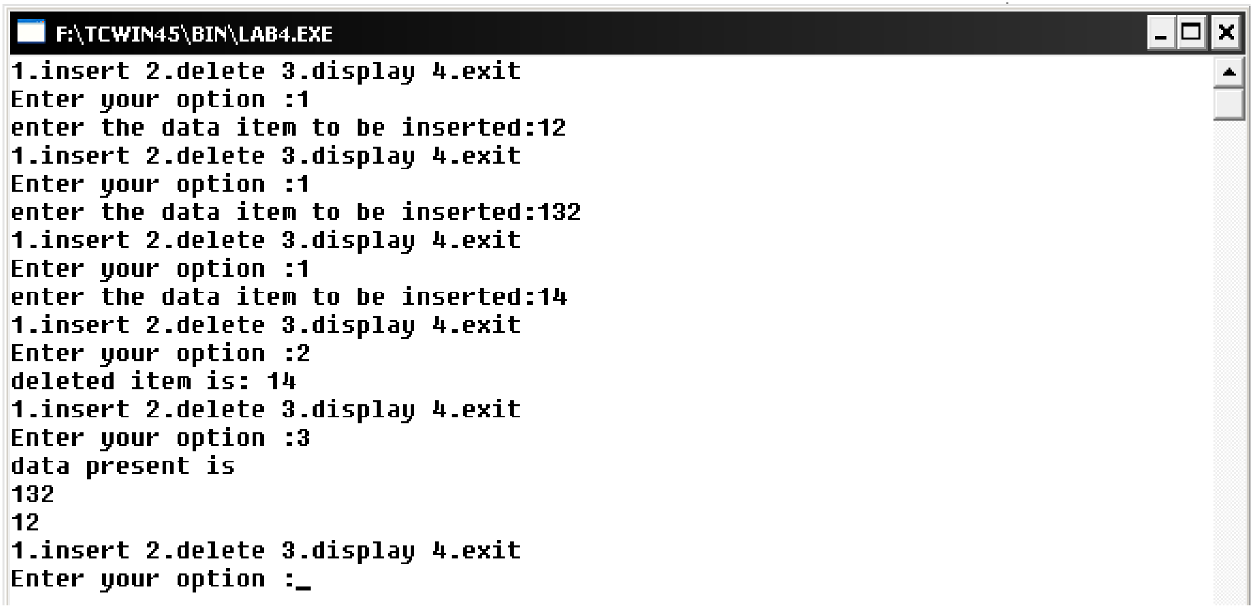
4. Write a C++ program to create a class called LIST (linked list) with member functions to insert an element at the front as well as to delete an element from the front of the list. Demonstrate all the functions after creating a list object.

Program:

```
#include<iostream.h>
#include <conio.h>
#include <process.h>
struct node
{
        int data;
        struct node *link;
};
typedef struct node N;
class List
{
        N * start;
        public:
        List()
        {
                start=NULL;
        }
        void insert_front();
        void delete_front();
        void display();
};
void List :: insert_front()
{
        N *temp;
        int item;
        temp = new N;
        if(temp==NULL)
        {
                cout<<endl<<"memory allocation failed"<<'\n';
                exit(0);
        }
        cout<<"enter the data item to be inserted:";
        cin>> item;
        temp->data =item;
        temp->link =start;
        start = temp;
}
```

```
void List :: delete_front()
{
        N *temp;
        if(start==NULL)
        {
                cout<<"list does not exist"<<endl;
                return;
        }
        temp = start;
        start = temp->link;
        cout << "deleted item is: "<< temp->data<<'\n';
        delete temp;
}
void List :: display()
{
        N *temp;
        if(start==NULL)
        {
                cout<<"list does not exist"<<'\n';
                return;
        }
        temp = start;
        cout << "data present is" <<'\n';
        while(temp != NULL)
        {
                cout << temp->data<<'\n';
                temp=temp->link;
        }
}
void main()
{
        List Lt; int oper;
        clrscr();
        for(;;)
        {
                cout <<"1.insert 2.delete 3.display 4.exit"<<'\n';
                cout<<"Enter your option :";
                cin>> oper;
                switch(oper)
                {
                        case 1: Lt.insert_front();
                                break;
                        case 2: Lt.delete_front();
                                break;
                        case 3: Lt.display();
                                break;
                        case 4: default: exit(0);
                }
        }
}
```

OUTPUT:



```
F:\TCWIN45\BIN\LAB4.EXE
1.insert 2.delete 3.display 4.exit
Enter your option :1
enter the data item to be inserted:12
1.insert 2.delete 3.display 4.exit
Enter your option :1
enter the data item to be inserted:132
1.insert 2.delete 3.display 4.exit
Enter your option :1
enter the data item to be inserted:14
1.insert 2.delete 3.display 4.exit
Enter your option :2
deleted item is: 14
1.insert 2.delete 3.display 4.exit
Enter your option :3
data present is
132
12
1.insert 2.delete 3.display 4.exit
Enter your option :_
```

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

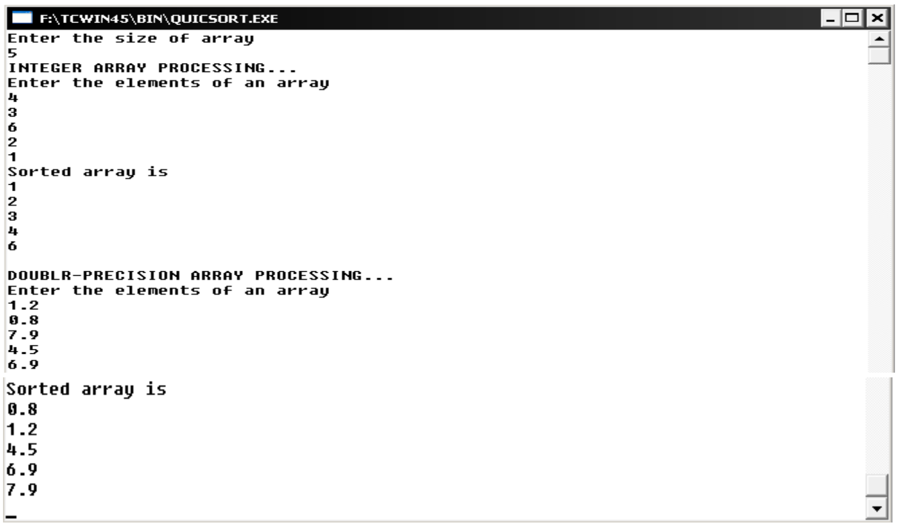
Program:

```
#include <iostream.h>
#include <conio.h>
#define SIZE 10
template <class T>
class QSORT
{
        private: T Array[SIZE];
        int len;
        public: QSORT(int); //constructor
        void getArray(void);
        void quickSort(int,int);
        int partition(int, int);
        void printArray(void);
}; //End of QSORT class
template <class T>
QSORT <T> :: QSORT(int length)
{
        len = length;
        //Initialize all array elements to 0
        for (int k = 0; k < len; k++)
        {
        Array[k] = 0;
        }
} //End of the constructor function
template <class T>
void QSORT <T> :: getArray(void)
{
        cout<<"Enter the elements of an array"<<endl;
        for (int k = 0; k < len; k++)
        {
                cin>>Array[k];
        }
} //End of getArray function

template <class T>
void QSORT <T> :: quickSort(int low, int high)
{
        int pos;
```

```
          if(low < high)
          {
                    pos = partition(low,high);
                    quickSort(low, pos-1);
                    quickSort(pos+1,high);
          }
} //End of quickSort function
template <class T>
int QSORT <T> :: partition(int low, int high)
{
          T key, temp;
          int left,right, true = 1;
          left = low;
          right= high;
          key = Array[low]; // Assume that the first element is a pivot value
          while(true)
          {
                    while(left <high && (key >= Array[left]))
                    {
                              left++;
                    }
                    while(key < Array[right])
                    {
                              right--;
                    }
                    if (left < right)
                    {
                              temp = Array[left];
                              Array[left] = Array[right];
                              Array[right]=temp;
                    }
                    else
                    {
                    temp = Array[low];
         Array[low] = Array[right];
         Array[right]=temp;
         return (right);
                    }
          } //End of while
}
```

```
template <class T>
void QSORT <T> :: printArray(void)
{
        cout<<"Sorted array is "<<endl;
        for (int k = 0; k < len; k++)
        {
                cout<<Array[k]<<endl;
        }
} //End of printArray function
void main()
{
        int size,low, high;
        clrscr();
        cout<<"Enter the size of array"<<endl;
        cin>>size;
        low = 0;
        high = size - 1;
        QSORT<int> IA(size);
        QSORT<double> DA(size);
        cout<<"INTEGER ARRAY PROCESSING..."<<endl;
        IA.getArray();
        IA.quickSort(low, high);
        IA.printArray();
        cout<<"\nDOUBLR-PRECISION ARRAY PROCESSING..."<<endl;
        DA.getArray();
        DA.quickSort(low, high);
        DA.printArray();
        getch();
}
```

OUTPUT:



```
Enter the size of array
5
INTEGER ARRAY PROCESSING...
Enter the elements of an array
4
3
6
2
1
Sorted array is
1
2
3
4
6

DOUBLR-PRECISION ARRAY PROCESSING...
Enter the elements of an array
1.2
0.8
7.9
4.5
6.9
Sorted array is
0.8
1.2
4.5
6.9
7.9
_
```

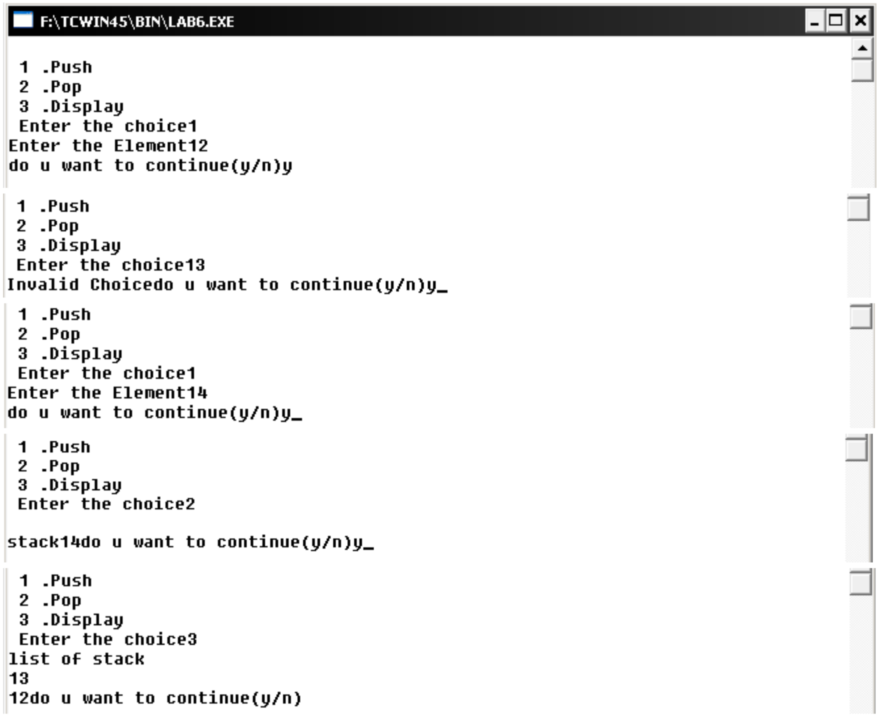
6. Write a C++ program to create a class called STACK using an array of integers. Implement the following operations by overloading the operators '+' and '-':
(i) s1 = s1 + element; where s1 is an object of the class STACK and element is an integer to be pushed on the top of the stack
(ii) s1 = s1- ; where s1 is an object of the class STACK. '-'operator pops the element.
Handle the STACK empty and full conditions. Also display the contents of the stack after each operation, by overloading the << operator.

Program:

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
# define max 5
class stack
{
        int sta[max];
        int top;
        public:
        stack()
        {
                top=-1;
        }
        friend stack operator+(stack s1,int item );
        friend stack operator-(stack s1);
        friend ostream &operator<<(ostream&,stack);
};
stack operator+(stack s1, int item)
{
        if(s1.top==max-1)
        {
                cout<<"\n stack overflow";
                return(s1);
        }
        else
        {
                s1.top++;
                s1.sta[s1.top]=item;
                return(s1);
        }
}
```

```
stack operator- (stack s1)
{
        if(s1.top==-1)
        {
                cout<<"\nstack underflow";
                return (s1);
        }
        else
        {
                cout<<"\nstack"<<s1.sta[s1.top];
                s1.top--;
                return(s1);
        }
}
ostream& operator<<(ostream &dout,stack s1)
{
        if(s1.top==-1)
                dout<<"\n Stack Empty";
        else
        {
                dout<<"\list of stack";
        for(int i=s1.top;i>=0;i--)
                dout<<"\n"<<s1.sta[i];
        }
        return(dout);
}
void main()
{
        stack s1;
        int ch,item;
        char ch1;
        clrscr();
        do
        {
                cout<<"\n 1 .Push";
                cout<<"\n 2 .Pop";
                cout<<"\n 3 .Display";
                cout<<"\n Enter the choice";
                cin>>ch;
                switch(ch)
                {
                        case 1:
                                cout<<"Enter the Element";
                                cin>>item;
                                s1=s1+item;
                                break;
                        case 2:
                                s1=-s1;
                                break;
                        case 3:
                                cout<<s1;
                                break;
                        default:
                                cout<<"Invalid Choice";
                }
        cout<<"do u want to continue(y/n)";
        cin>>ch1;
        clrscr();
        }
        while(ch1=='Y'||ch1=='y');
}
```

OUTPUT:



F:\TCWIN45\BIN\LAB6.EXE

```
 1 .Push
 2 .Pop
 3 .Display
 Enter the choice1
Enter the Element12
do u want to continue(y/n)y

 1 .Push
 2 .Pop
 3 .Display
 Enter the choice13
Invalid Choicedo u want to continue(y/n)y_

 1 .Push
 2 .Pop
 3 .Display
 Enter the choice1
Enter the Element14
do u want to continue(y/n)y_

 1 .Push
 2 .Pop
 3 .Display
 Enter the choice2

stack14do u want to continue(y/n)y_

 1 .Push
 2 .Pop
 3 .Display
 Enter the choice3
list of stack
13
12do u want to continue(y/n)
```

7. Create a class called MATRIX using 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 + and – operators respectively. Display the results by overloading the operator <<.
If (m1==m2) then m3 = m1+m2 and m4 = m1-m2 else display error.

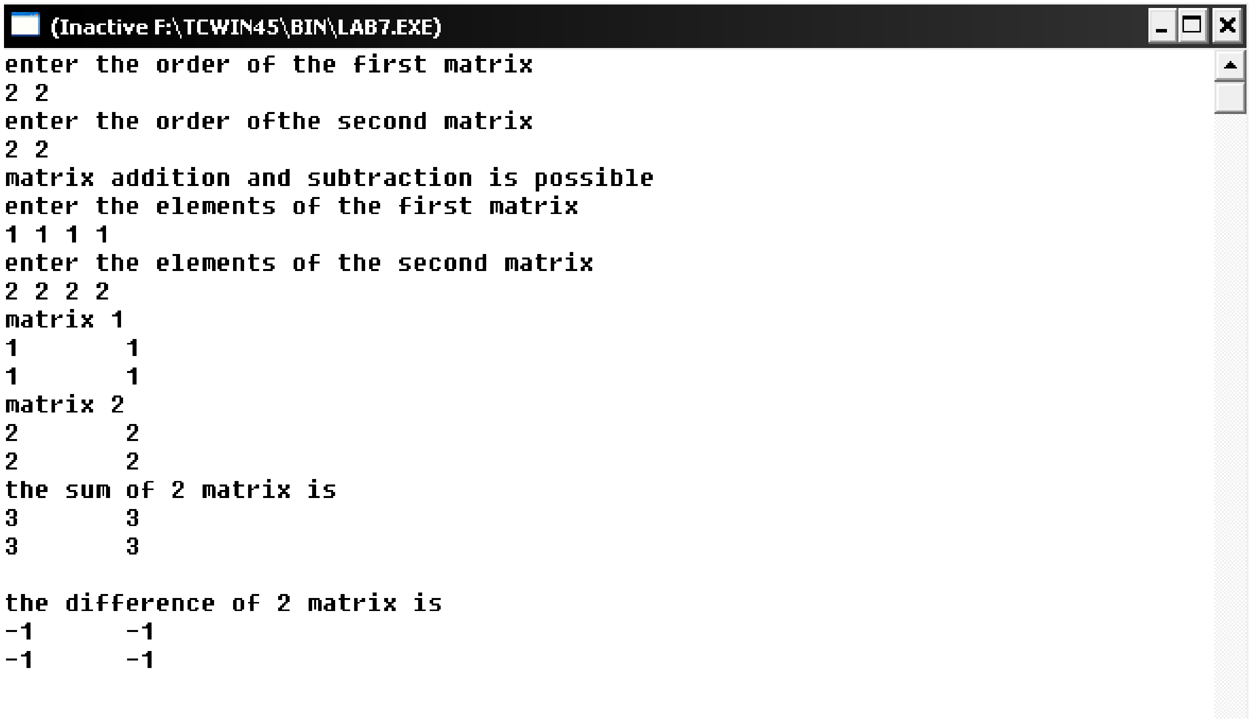
Program:

```
#include <iostream.h>
#include<conio.h>
class matrix
{
        int a[5][5],row,col;
        public:
        void read_order()
        {
                cin>>row>>col;
        }
        void read_matrix()
        {
                int i,j;
                for(i=0; i<row; i++)
                        for(j=0; j<col; j++)
                                cin>>a[i][j];
        }
        int operator ==(matrix mt)
        {
                if(row == mt.row && col == mt.col)
                {
                        cout<<"matrix addition and subtraction is possible"<<endl;
                        return 1;
                }
                return 0;
        }
        matrix operator +(matrix mp)
        {
                matrix temp;
                int i,j;
                for(i=0; i<row; i++)
                        for(j=0; j<col; j++)
                                temp.a[i][j]=a[i][j] + mp.a[i][j];
```

```
		temp.row= row;
		temp.col= col;
		return temp;
	}
	matrix operator -(matrix ms)
	{
		matrix temp;
		int i,j;
		for(i=0; i<row; i++)
			for(j=0; j<col; j++)
				temp.a[i][j] = a[i][j] - ms.a[i][j];
		temp.row = row;
		temp.col = col;
		return temp;
	}
	friend void operator <<(ostream &, matrix);
};
void operator <<(ostream &ct, matrix mb)
{
	int i,j;
	for(i=0; i<mb.row; i++)
	{
		for(j=0; j<mb.col; j++)
		{
			ct<<mb.a[i][j]<<'\t';
		}
		ct<<endl;
	}
}
void main()
{
	matrix m1,m2,m3,m4;
	clrscr();
	cout<<"enter the order of the first matrix"<<endl;
	m1.read_order();
	cout<<"enter the order ofthe second matrix"<<endl;
	m2.read_order();
	if(m1==m2)
	{
		cout<<"enter the elements of the first matrix"<<endl;
```

```
                m1.read_matrix();
                cout<<"enter the elements of the second matrix"<<endl;
                m2.read_matrix();
                cout<<"matrix 1"<<endl;
                cout<<m1;
                cout<<"matrix 2"<<endl;
                cout<<m2;
                m3=m1+m2;
                m4=m1-m2;
                cout<<"the sum of 2 matrix is ";
                cout<<endl<<m3;

                cout<<endl<<"the difference of 2 matrix is ";
                cout <<endl<<m4;
        }
        else
                cout<<"matrix addition and subtraction is not possible"<<endl;
}
```

OUTPUT:



(Inactive F:\TCWIN45\BIN\LAB7.EXE)

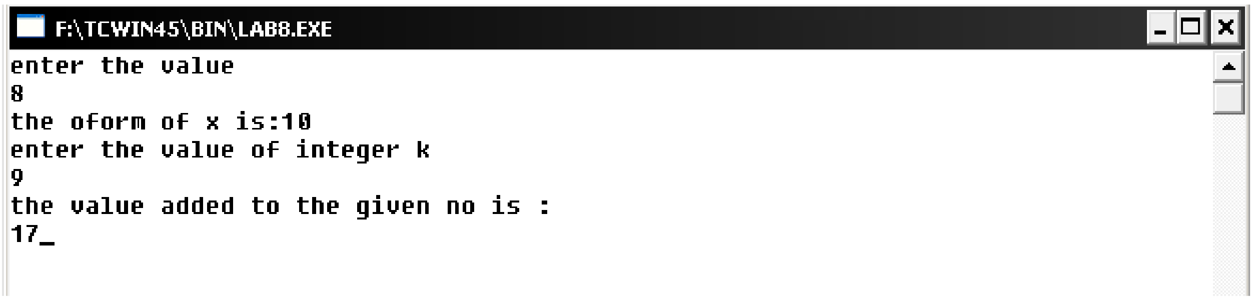
```
enter the order of the first matrix
2 2
enter the order ofthe second matrix
2 2
matrix addition and subtraction is possible
enter the elements of the first matrix
1 1 1 1
enter the elements of the second matrix
2 2 2 2
matrix 1
1       1
1       1
matrix 2
2       2
2       2
the sum of 2 matrix is
3       3
3       3

the difference of 2 matrix is
-1      -1
-1      -1
```

8. Write a C++ 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 result by overloading the operator <<. Also display the values of h and y.

Program:
```
#include <iostream.h>
#include <conio.h>
class octal
{
        int oform;
        public:
        octal (int x);
        int operator + (int x);
        friend void operator <<(ostream &print,octal x);
};
octal :: octal(int x)
{
        int rem ,c=1;
        oform = 0;
        while (x>0)
        {
                rem = x % 8;
                x = x/8;
                oform = oform + rem *c;
                c = c * 10;
        }
}
int octal :: operator +(int x)
{
        int r,temp=oform;
        int oform=0,c=1;
        while(temp)
        {
                r = temp %10;
                temp = temp / 10;
                oform = oform +r *c;
```

```
                    c = c *8;
          }
          return oform +x;
}
void operator <<(ostream &print,octal x)
{
          print<<x.oform;
}
void main()
{
          int x,y,k;
          clrscr();
          cout<<"enter the value"<<endl;
          cin >>x;
          octal h = x;
          cout <<"the oform of x is:"<<h;
          cout <<endl<<"enter the value of integer k"<<endl;
          cin>>k; y=h+k;
          cout <<"the value added to the given no is :"<<endl;
          cout<<y;
          getch();
}
```

OUTPUT:



```
enter the value
8
the oform of x is:10
enter the value of integer k
9
the value added to the given no is :
17_
```

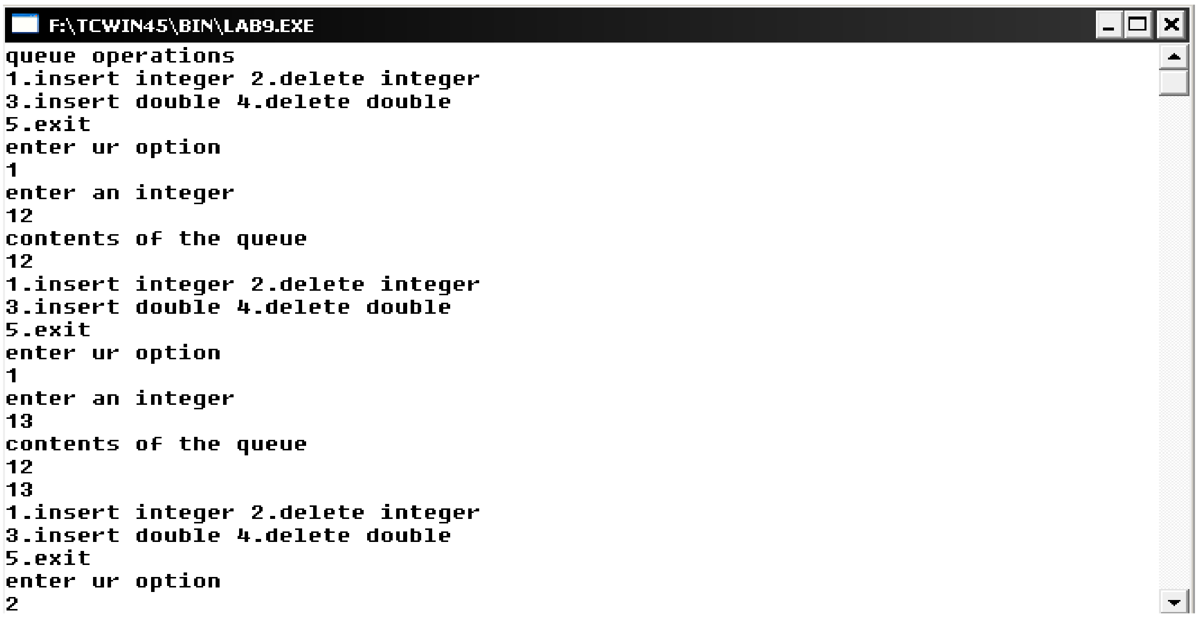
9. Write a C++ program to create a class called QUEUE with member functions to add an element and to delete an element from the queue. Using the member functions, implement a queue of integers and double. Demonstrate the operations by displaying the contents of the queue after every operation.

Program:

```
#include <iostream.h>
#include <conio.h>
#include<stdlib.h>
#include <process.h>
#define max 5
template <class Q>
class QUEUE
{
	int f,r;
	Q q[max];
	public:
	QUEUE()
	{
		f=-1;
		r=-1;
	}
	void insert(Q);
	void del();
};
template <class Q>
void QUEUE<Q>::insert(Q item)
{
	if (r==max-1)
	{
		cout<<"queue is full"<<endl;
		return;
	}
	r++;
	q[r]=item;
	if(f==-1)
		f++;
	if(f==-1)
	{
		cout<<"queue is empty"<<endl;
```

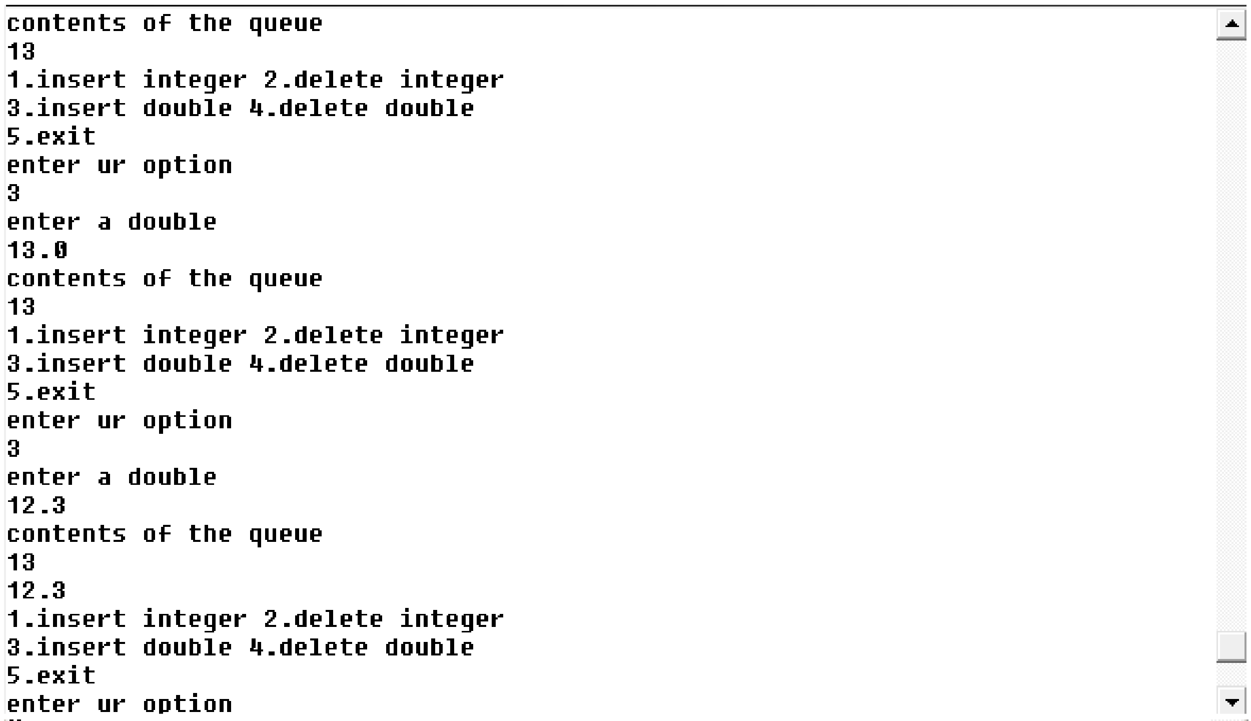
```
                return;
        }
        cout<<"contents of the queue"<<endl;
        for(int i=f;i<=r;i++)
                cout<<q[i]<<endl;
        }
template<class Q>
void QUEUE<Q>::del()
{
        Q item;
        if(f==-1)
        {
                cout<<"queue is empty"<<endl;
                return;
        }
        item =q[f];
        cout<<"deleted element is"<<'\n'<<item<<endl;
        if(f==r)
                f=r=-1;
        else
                f++;
        if(f==-1)
        {
                cout<<"queue is empty"<<endl;
                return;
        }
        cout<<"contents of the queue"<<endl;
        for(int i=f;i<=r;i++)
                cout<<q[i]<<endl;
}
void main()
{
        QUEUE<int>x;
        QUEUE<double>y;
        int ch,i;
        double d;
        clrscr();
        cout<<"queue operations"<<endl;
        for(;;)
        {
                cout<<"1.insert integer 2.delete integer"<<endl;
                cout<<"3.insert double 4.delete double "<<endl;
                cout<<"5.exit"<<endl; cout<<"enter ur option"<<endl;
                cin>>ch;
                switch(ch)
                {
                        case 1 : cout<<"enter an integer"<<endl;
                                cin>>i;
                                x.insert(i);
                                break;

                        case 2 : x.del();
                                break;
                        case 3 : cout<<"enter a double"<<endl;
                                cin>>d;
                                y.insert(d);
                                break;
                        case 4 : y.del();
                                break;
                        case 5 : default: exit(0);
                }
        }
        getch();
}
```

OUTPUT:

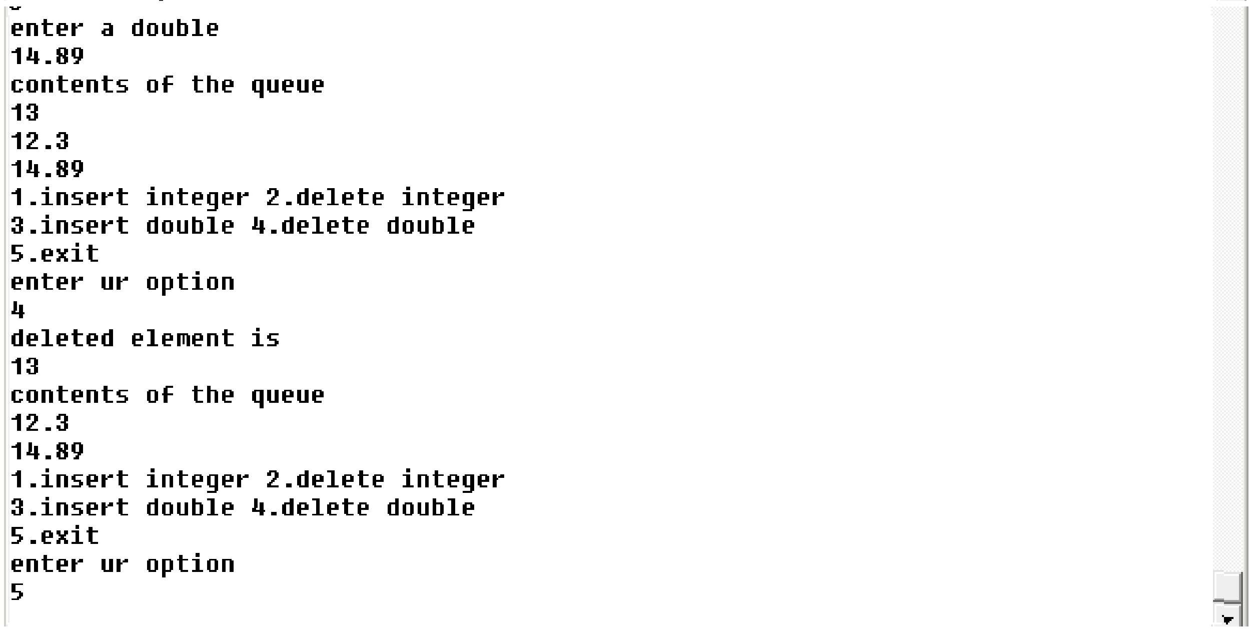


F:\TCWIN45\BIN\LAB9.EXE

```
queue operations
1.insert integer 2.delete integer
3.insert double 4.delete double
5.exit
enter ur option
1
enter an integer
12
contents of the queue
12
1.insert integer 2.delete integer
3.insert double 4.delete double
5.exit
enter ur option
1
enter an integer
13
contents of the queue
12
13
1.insert integer 2.delete integer
3.insert double 4.delete double
5.exit
enter ur option
2
```



```
contents of the queue
13
1.insert integer 2.delete integer
3.insert double 4.delete double
5.exit
enter ur option
3
enter a double
13.0
contents of the queue
13
1.insert integer 2.delete integer
3.insert double 4.delete double
5.exit
enter ur option
3
enter a double
12.3
contents of the queue
13
12.3
1.insert integer 2.delete integer
3.insert double 4.delete double
5.exit
enter ur option
```



```
enter a double
14.89
contents of the queue
13
12.3
14.89
1.insert integer 2.delete integer
3.insert double 4.delete double
5.exit
enter ur option
4
deleted element is
13
contents of the queue
12.3
14.89
1.insert integer 2.delete integer
3.insert double 4.delete double
5.exit
enter ur option
5
```

10. Write a C++ program to create a class called STUDENT with data members USN, Name and Age. Using inheritance, create the classes UGSTUDENT and PGSTUDENT having fields as Semester, Fees and Stipend. Enter the data for at least 5 students. Find the semester-wise average age for all UG and PG students separately.

Program:

```
#include <conio.h>
#include <stdio.h>
#include <iostream.h>
#define MAX 2 //You can change to the requirement
//Base class with member fields and member function
class STUDENT
{
        private: char *name;
        char *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;
}
/*Derived class for ugstudent with member fields and functions*/
class UGstudent:public STUDENT
{
        private: int semester;
        float fees;
        float stipend;

        public:
        void putugdata();
```

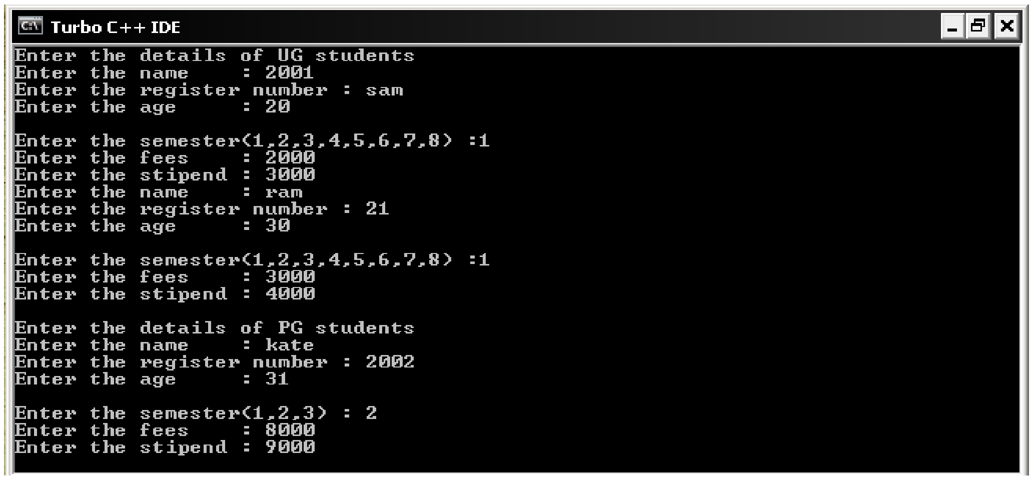
```
	int getsemester()
	{
		return(semester);
	}
};
void UGstudent::putugdata()
{
	STUDENT::putdata();
	cout<<"\nEnter the semester(1,2,3,4,5,6,7,8) :";
	cin >> semester;
	cout<<"Enter the fees : ";
	cin >> fees;
	cout<<"Enter the stipend : ";
	cin >> stipend;
}
/*Derived class for pgstudent with member fields and functions*/
class PGstudent:public STUDENT
{
	private: int semester;
	float fees;
	float stipend;
	public:
	void putpgdata();
	int getsemester()
	{
		return(semester);
	}
};
void PGstudent::putpgdata()
{
	STUDENT::putdata();
	cout<<"\nEnter the semester(1,2,3) : ";
	cin >> semester;
	cout<<"Enter the fees : ";
	cin >> fees;

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

```
// main program
void main()
{
        UGstudent ugstudent[MAX]; /*for MAX ugstudent*/
        PGstudent pgstudent[MAX]; /*for MAX pgstudent*/
        clrscr();
        cout<<"Enter the details of UG students"<<endl;
        for(int i=0;i<MAX;i++) //accepting MAX ugstudent data
        {
                ugstudent[i].putugdata();
        }
        /* 8 variables to hold total age of each semister*/
        int total1=0,total2=0,total3=0,total4=0,total5=0,
        total6=0,total7=0,total8=0;
        /* 8 variables to hold no. of student in each semister*/
        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].getsemester();
                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();
```

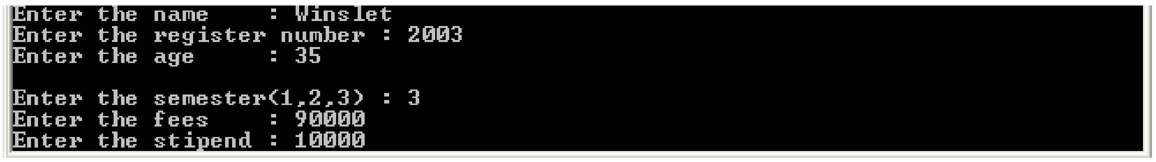
```
                    total5 += tempage;
                    ++tco5;
                    break;
            case 6 : tempage=ugstudent[i].getage();
                    total6 += tempage;
                    ++tco6;
                    break;
            case 7 : tempage=ugstudent[i].getage();
                    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++) /*accept the MAX pgstudents data*/
{
        pgstudent[i].putpgdata();
}
/*3 variables to hold total age of each semister*/
int tot1=0,tot2=0,tot3=0;
/*3 variables to hold no. of student in each semister*/
int tc1=0,tc2=0,tc3=0;
int temsem=0,temage=0;
for(i=0;i<MAX;i++)
{
        temsem=pgstudent[i].getsemester();
        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(); /*displaying the result on the screen*/
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();
}
```

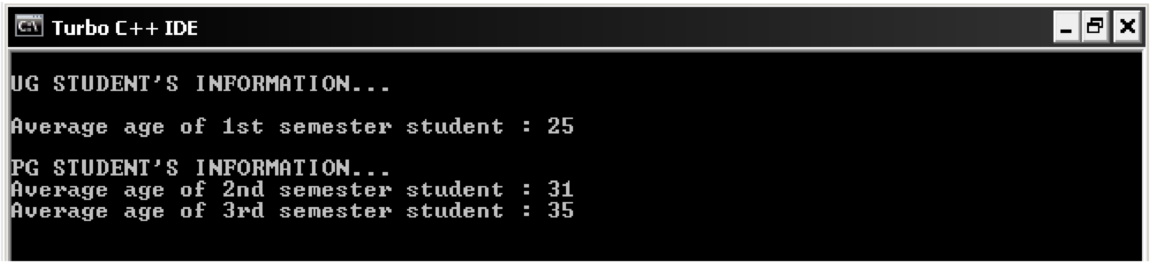
OUTPUT:



```
Turbo C++ IDE
Enter the details of UG students
Enter the name     : 2001
Enter the register number : sam
Enter the age      : 20

Enter the semester(1,2,3,4,5,6,7,8) :1
Enter the fees     : 2000
Enter the stipend : 3000
Enter the name     : ram
Enter the register number : 21
Enter the age      : 30

Enter the semester(1,2,3,4,5,6,7,8) :1
Enter the fees     : 3000
Enter the stipend : 4000

Enter the details of PG students
Enter the name     : kate
Enter the register number : 2002
Enter the age      : 31

Enter the semester(1,2,3) : 2
Enter the fees     : 8000
Enter the stipend : 9000
```



```
Enter the name     : Winslet
Enter the register number : 2003
Enter the age      : 35

Enter the semester(1,2,3) : 3
Enter the fees     : 90000
Enter the stipend : 10000
```



```
UG STUDENT'S INFORMATION...

Average age of 1st semester student : 25

PG STUDENT'S INFORMATION...
Average age of 2nd semester student : 31
Average age of 3rd semester student : 35
```

11. Write a C++ program to create a class called DLIST (doubly Linked List) with member functions to insert a node at a specified position and delete a node from a specified position of the list. Demonstrate the operations by displaying the content of the list after every operation.

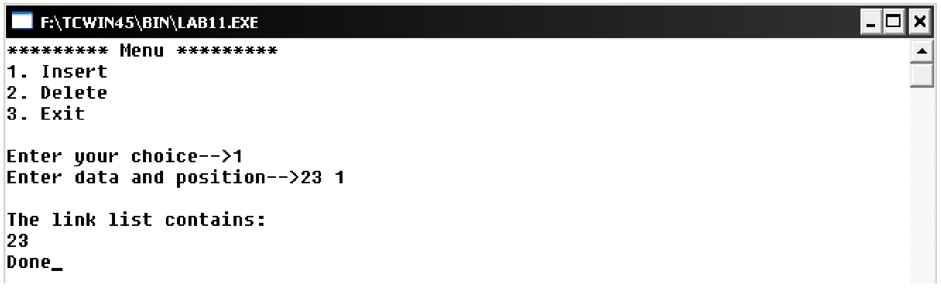
Program:

```
#include <iostream.h>
#include <conio.h>
#include <process.h>
class node
{
        public:
        node *right;
        int data;
        node *left;
};
class dlinklist
{
        public:
        dlinklist() :first(0),size(0) {}
        void add(int,int);
        void del(int);
        void display(void);
        private:
        node *first;
        int size;
};
void main()
{
        dlinklist l;
        int c,i,p;
        while(1)
        {
                clrscr();
                cout<<"********* Menu *********"<<endl;
                cout<<"1. Insert"<<endl;
                cout<<"2. Delete"<<endl;

                cout<<"3. Exit"<<endl<<endl;
                cout<<"Enter your choice-->";
                cin>>c;
                switch (c)
                {
                        case 1:
                                cout<<"Enter data and position-->";
                                cin>>i>>p;
                                l.add(i,p);
                                break;
                        case 2:
                                cout<<"Enter position-->";
                                cin>>p;
                                l.del(p);
                                break;
                        case 3:
                                exit(1);
                        default:
                                cout<<"Wrong choice";
                }
                cout<<endl;
                l.display();
                cout<<endl<<"Done";
                getch();
        }
}
void dlinklist::display(void)
{
        node* add;
        cout<<"The link list contains:"<<endl;
        add=first;
        while(add)
        {
                cout<<add->data<<" ";
                add=add->right;
        }
        if (first==NULL)
                cout<<"Empty";
}

void dlinklist::add(int ele,int pos)
{
        if (pos>size+1 || pos<1)
                cout<<"Invalid Position";
```

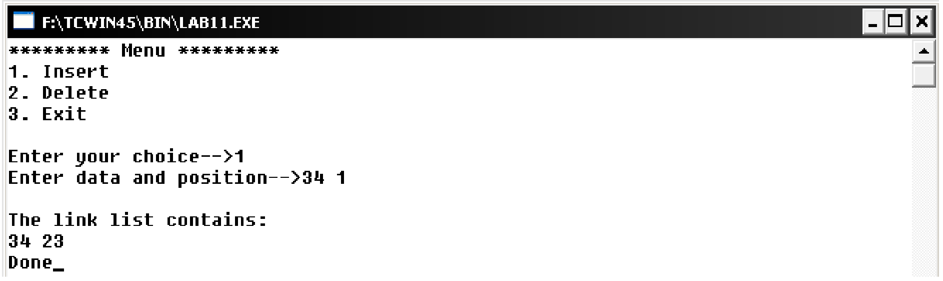
```
	else
	{
		node *newn=new node;
		newn->data=ele;
		if (pos==1)
		{
			newn->right=first;
			newn->left=0;
			if (first)
				first->left=newn;
			first=newn;
		}
		else
		{
			node *add=first;
			for (int i=1;i<pos-1;i++)
				add=add->right;
			newn->right=add->right;
			newn->left=add;
			if (newn->right)
				newn->right->left=newn;
			add->right=newn;
		}
		size++;
	}
}
void dlinklist::del(int pos)
{
	if (pos>size || pos<1)
		cout<<"Invalid Position";
	else
	{
		node *add;

		if (pos==1)
		{
			add=first;
			first=first->right;
			delete add;
		}
		else
```

```
                {
                        node *add=first;
                        for (int i=1;i<pos;i++)
                                add=add->right;
                        add->left->right=add->right;
                        if (add->right->left)
                                add->right->left=add->left;
                        delete add;
                }
                size--;
        }
}
```

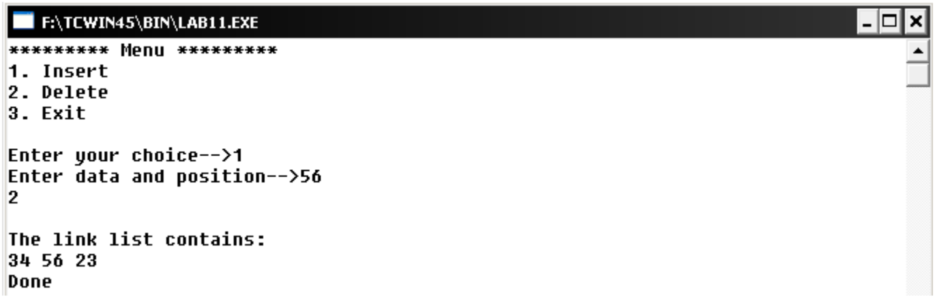
OUTPUT:



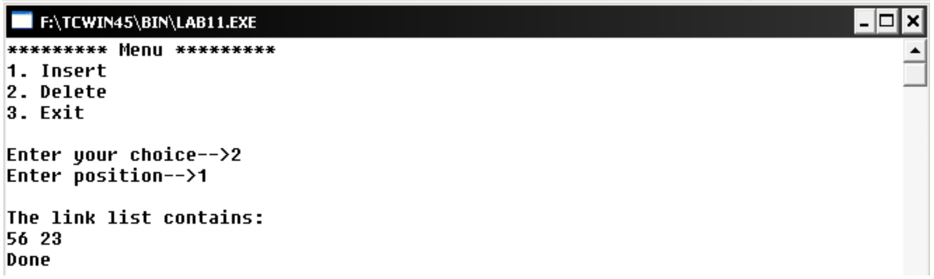
```
F:\TCWIN45\BIN\LAB11.EXE
********* Menu *********
1. Insert
2. Delete
3. Exit

Enter your choice-->1
Enter data and position-->23 1

The link list contains:
23
Done_
```



```
F:\TCWIN45\BIN\LAB11.EXE
********* Menu *********
1. Insert
2. Delete
3. Exit

Enter your choice-->1
Enter data and position-->34 1

The link list contains:
34 23
Done_
```



```
F:\TCWIN45\BIN\LAB11.EXE
********* Menu *********
1. Insert
2. Delete
3. Exit

Enter your choice-->1
Enter data and position-->56
2

The link list contains:
34 56 23
Done
```



```
F:\TCWIN45\BIN\LAB11.EXE
********* Menu *********
1. Insert
2. Delete
3. Exit

Enter your choice-->2
Enter position-->1

The link list contains:
56 23
Done
```

12. Write a C++ program to create a class called EXPRESSION. Using appropriate member functions convert a given valid Infix expression into postfix form. Display the infix and postfix expressions.

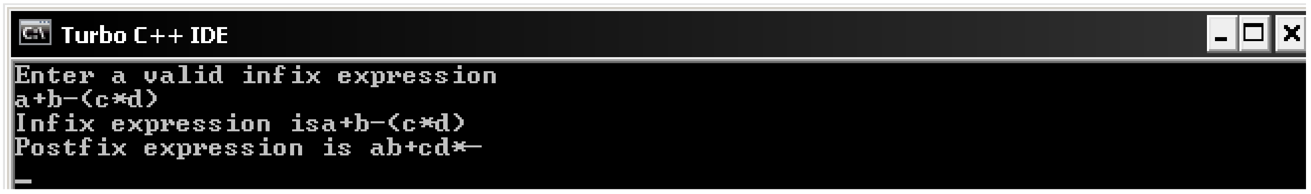
Program:

```
#include <iostream.h>
#include <string.h>
#include <conio.h>
#include <stdio.h>
class EXPRESSION
{
        private: char infix[20];
        char postfix[20];
        char stack[20];
        int top;
        public : EXPRESSION() //constructor
                {
                        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)
{
        clrscr();
        cout<<"Infix expression is ";
        cout<<infix<<endl;
        cout<<"Postfix expression is ";
        cout<<postfix<<endl;

}
```

```
void main ()
{
        EXPRESSION e;
        clrscr();
        e.getINFIX();
        e.infix_to_postfix ();
        e.putPOSTFIX();
        (flushall());
        getch();
}
```

OUTPUT:



```
Enter a valid infix expression
a+b-(c*d)
Infix expression isa+b-(c*d)
Postfix expression is ab+cd*-
_
```

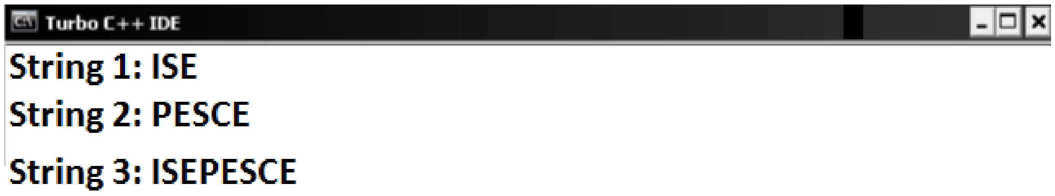
13. Write a C++ program to create a class called STRING and implement the following operations. Display the results after every operation by overloading the operator <<.
(i) STRING s1 = "ISE"
(ii) STRING s2 = "PESCE"
(iii) STRING s3 = s1 + s2 (Use copy constructor)

Program:

```
#include <iostream.h>
#include <string.h>
#include <conio.h>
class string
{
        private:
        char str[20];
        public:
        string() // constructor
        {
                strcpy(str," ");
        }
        string(string& x) // copy constructor
        {
                strcpy(str,x.str);
        }
        string(char *s) //constructor with parameter
        {
                strcpy(str, s);
        }
        string operator +(string ss)
        {
                string temp;
                strcpy(temp.str, str);
                strcat(temp.str, ss.str);
                return temp;
        }

        friend ostream& operator <<(ostream &os,string &ss)
        {
                os<< ss.str;
                return os;
        }
}; //End of class string
```

```
void main()
{
	clrscr();
	string s1 = "ISE"; // copy constructor called
	string s2 = "PESCE";
	cout<< "String 1: ";
	cout<< s1<<"\n"; // << overloaded
	cout<< "String 2: ";
	cout<<s2<<"\n\n"; // << overloaded
	string s3 = s1 + s2; // + is overloaded
	cout<< "String 3: ";
	cout<< s3;
	getch();
}
```

OUTPUT:



```
String 1: ISE
String 2: PESCE

String 3: ISEPESCE
```

14. Write a C++ program to create a class called BIN_TREE (Binary Tree) with member functions to perform in-order, preorder and post-order traversals. Create a BIN_TREE object and demonstrate the traversals.

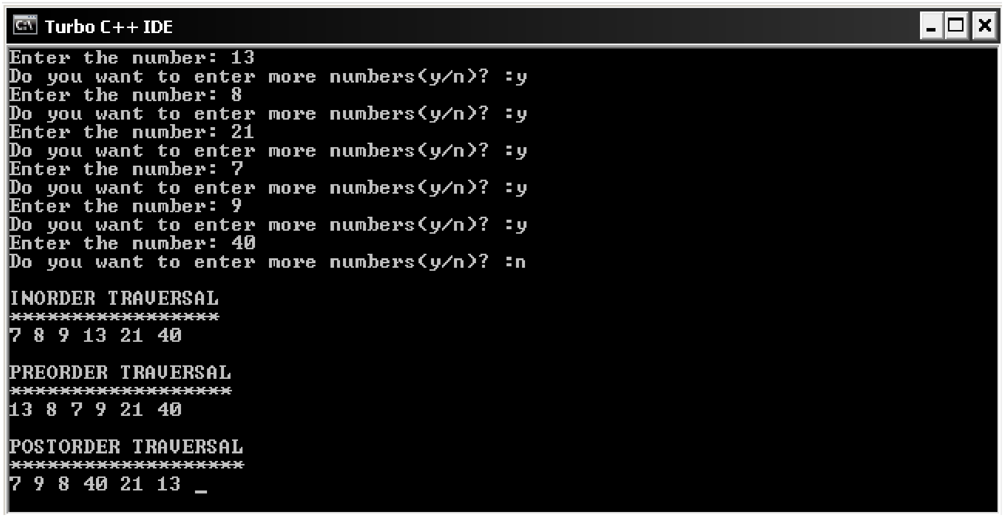
Program:

```
#include <iostream.h>
#include <conio.h>
#define true 1
#define false 0
#define maxnode 10
class node
{
        public:
        int item;
        node *left;
        node *right;
};
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;
                }

                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;
                    }
                } //End of else if
        } //End of while
} //End of create
//member function to inorder traversal
void bin_tree::inorder(node *base)
{
        if(base != NULL)
        {
                inorder(base->left);
                cout<<base->item<<" ";
                inorder(base->right);
                return;

        }
}
```

```
//member function to preorder traversal
void bin_tree::preorder(node *base)
{
        if(base != NULL)
        {
                cout<<base->item<<" ";
                preorder(base->left);
                preorder(base->right);
                return;
        }
}
//member function to postorder traversal
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();
} // End of main()
```

OUTPUT:



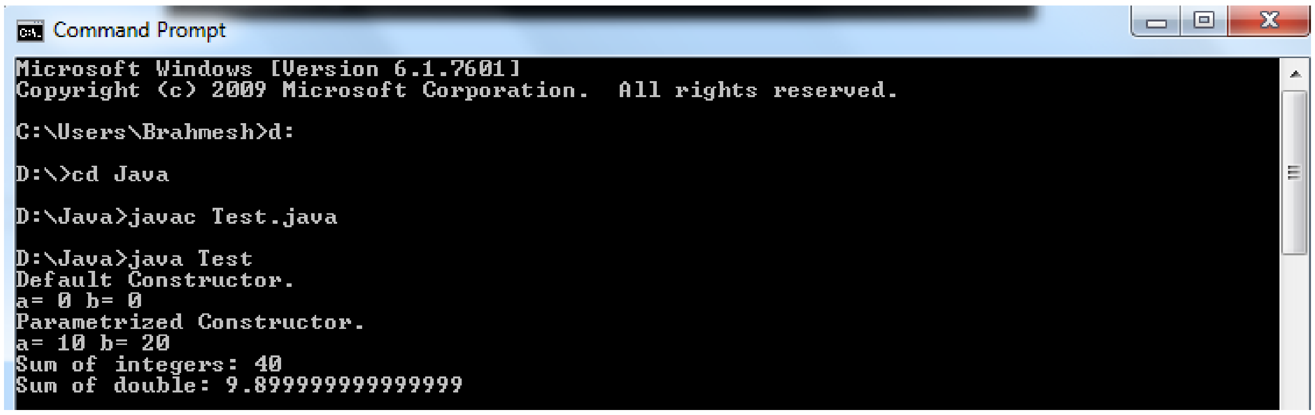
```
Enter the number: 13
Do you want to enter more numbers(y/n)? :y
Enter the number: 8
Do you want to enter more numbers(y/n)? :y
Enter the number: 21
Do you want to enter more numbers(y/n)? :y
Enter the number: 7
Do you want to enter more numbers(y/n)? :y
Enter the number: 9
Do you want to enter more numbers(y/n)? :y
Enter the number: 40
Do you want to enter more numbers(y/n)? :n

INORDER TRAVERSAL
*****************
7 8 9 13 21 40

PREORDER TRAVERSAL
******************
13 8 7 9 21 40

POSTORDER TRAVERSAL
*******************
7 9 8 40 21 13 _
```

15. Write a JAVA Program to demonstrate Constructor overloading and Method Overloading.

```
class example
{
	int a,b;
	public example()
	{
		a=0; b=0;
		System.out.println("Default Constructor.");
		System.out.println("a= " + a + " " + "b= " + b);
	}
	public example(int x,int y)
	{
		a=x;b=y;
		System.out.println("Parametrized Constructor.");
		System.out.println("a= " + a + " " + "b= " + b);
	}
	public void add(int x,int y)
	{
		int c=x+y;
		System.out.println("Sum of integers: "+c);
	}
	public void add(double x, double y)
	{
		double c=x+y;
		System.out.println("Sum of double: "+c);
	}
}
```

```
class Test
{
        public static void main(String args[])
        {
                example e1=new example();
                example e2=new example(10,20);
                e1.add(20,20);
                e1.add(5.6,4.3);
        }
}
```

OUTPUT:



Command Prompt

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Brahmesh>d:

D:\>cd Java

D:\Java>javac Test.java

D:\Java>java Test
Default Constructor.
a= 0 b= 0
Parametrized Constructor.
a= 10 b= 20
Sum of integers: 40
Sum of double: 9.899999999999999
```

Viva Questions:

1. What is a Variable?
2. Define Class.
3. Which are Primitive data types?
4. Compare and contrast method overloading with method overriding.
5. What are inner class and anonymous class?
6. When you will decide to use if statement?
7. Need of Constructor?
8. What is OOPs?
9. State the purpose of Encapsulation.
10. Give an example for Polymorphism.
11. What is the difference between procedural and object-oriented programs?
12. What is an Object and how do you allocate memory to it?
13. Differentiate between constructor and method.
14. How C++ Differs From Java?
15. What are methods and how are they defined?
16. What is the difference between an argument and a parameter?
17. What are different types of access modifiers?
18. What are the differences between references and pointers?
19. What are virtual functions – Write an example?
20. What is this pointer?