INDEX

Appendixes:

PROGRAM No. – 1

Objective : Write a program to print the name of your Institute.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
void main()
{
        cout<<"Rustamji Institute of Technology";
        getch();
}
```

Exercise :

i. Write a program to display your name.

PROGRAM No. – 2

Objective : Write a program to add two integer numbers.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
void main()
{
        int a = 10;
        int b = 20;
        int c;
        c = a + b;
        cout<<a<<" + "<<b<<" = "<<c;
        getch();
}
```

Exercise :

i. Write a program to multiply three integer numbers.

ii. Write a program to add two integers and subtract the sum into third integer.

PROGRAM No. – 3

Objective : Write a program to take input of two numbers and print their multiplication.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
void main()
{
        int a,b,c;
        clrscr();
        cout<<"Enter First Number : ";
        cin>>a;
        cout<<"Enter Second Number : ";
        cin>>b;
        c = a * b;
        cout<<a<< " * " <<b<<" = "<<c;
        getch();
}
```

Precautions : Since an integer variable may have value between -32768 to +32767, so if any mathematical operation results an integer number less than -32768 or greater than +32767, then the result will be a garbage value.

Exercise :

i. Write a program to take input of **maximum marks** and **marks obtained** and print the **percentage marks**.

PROGRAM No. – 4

Objective : Write a program to find largest of three integers given by user.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
void main()
{
      int a, b, c;
      clrscr();
      cout<<"Enter First Number : ";
      cin>>a;
      cout<<"Enter Second Number : ";
      cin>>b;
      cout<<"Enter Third Number : ";
      cin>>c;
      if (a > b)
      {
            if (a > c)
                  cout<<a<<" is the greatest."<<endl;
            else if (a < c)
                  cout<<c<<" is the greatest."<<endl;
            else
                  cout<<a<<" and "<<c<<" are equal."<<endl;
      }
      else if (a < b)
      {
            if (b > c)
                  cout<<b<<" is the greatest."<<endl;
            else if (b < c)
                  cout<<c<<" is the greatest."<<endl;
            else
                  cout<<b<<" and "<<c<<" are equal"<<endl;
      }
      else
      {
            cout<<a<<" and "<<b<<" are equal"<<endl;
            if (a > c)
                  cout<<a<<" and "<<b<<" are the greatest"<<endl;
            else
                  cout<<c<<" is the greatest."<<endl;
      }
      getch();
}
```

Precautions : The input given by user should between -32768 and +32767.

Exercise :

i. Write a program to find the largest number from ten integers given by user.

PROGRAM No. – 5

Objective : Write a program to print table of a number given by user.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
void main()
{
      int x, i;
      clrscr();
      cout<<"Enter a Number : ";
      cin>>x;
      i = 1;
      while ( i <= 10)
      {
            cout<<endl<<i<<"  *  "<<x<<"  =  "<<i*x;
            i = i + 1;
      }
      getch();
}
```

Exercise :

i. Write a program to print table of a number given by user using **do-while loop**.

ii. Write a program to find the factorial of a number given by user.

PROGRAM No. – 6

Objective : Write a program to print star triangle of user-defined size.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
void main()
{
      int x, i, j;
      clrscr();
      cout<<"Enter Number of Rows : ";
      cin>>x;
      for (i = 1; i <= x; i++)
      {
            for (j = 1; j <= i; j++)
                  cout<<"*";
            cout<<endl;
      }
      getch();
}
```

Exercise :

i. Write a program to print **pyramid** of user-defined size.

PROGRAM No. – 7

Objective : Write a program to print menu for mathematical operations, i.e., Addition, Subtraction, Multiplication, Division and Exit.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
# include <process.h>
void main()
{
	int a,b,c,choice;
	while (1)
	{
		clrscr();
		cout<<endl<<"MATH MENU";
		cout<<endl<<"=========="<<endl;
		cout<<endl<<"1. Addition";
		cout<<endl<<"2. Subtraction";
		cout<<endl<<"3. Multiplication";
		cout<<endl<<"4. Division";
		cout<<endl<<endl<<"0. Exit"<<endl;
		cout<<endl<<endl<<"Enter Your Choice (0-5) : ";
		cin>>choice;
		if (choice >= 1 && choice <= 4)
		{
			cout<<endl<<"Enter First Number : ";
			cin>>a;
			cout<<endl<<"Enter Second Number : ";
			cin>>b;
		}
		switch(choice)
		{
			case 1:	c = a + b;
				cout<<a<<" + "<<b<<" = "<<c;
				break;
			case 2:	c = a - b;
				cout<<a<<" - "<<b<<" = "<<c;
				break;
			case 3:	c = a * b;
				cout<<a<<" * "<<b<<" = "<<c;
				break;
			case 4:	c = a / b;
				cout<<a<<" / "<<b<<" = "<<c;
				break;
			case 0:	cout<<endl<<"Thanks for using Math Menu.";
				cout<<endl<<"Press Any Key to Exit......";
				getch();
				exit(1);
			default:cout<<endl<<"Invalid Choice Entered.";
				cout<<endl<<"Please Enter Choice Between 0-4";
				cout<<endl<<"Press Any Key to Exit......";
				break;
		}
		getch();
	}
}
```

Exercise :

i. Write a program to print a menu which gives two choices, i.e., Factorial or Table of a number given by user using switch statement.

PROGRAM No. – 8

Objective : Write a program to take input of ten numbers and print them in ascending order.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
void main()
{
      int a[10],temp,i,j;
      clrscr();
      for(i = 0; i <= 9; i++)
      {
            cout<<"Enter Number"<<i+1<<" : ";
            cin>>a[i];
      }
      clrscr();
      cout<<"Numbers given by user : "<<endl;
      for(i = 0; i <= 9; i++)
            cout<<a[i]<<endl;
      cout<<endl<<"Sorted List : "<<endl;
      for(i = 0 ; i < 9; i++)
      {
            for(j = i; j <= 9; j++)
            {
                  if (a[i] > a[j])
                  {
                        temp = a[i];
                        a[i] = a[j];
                        a[j] = temp;
                  }
            }
      }
      for(i = 0; i <= 9; i++)
            cout<<a[i]<<endl;
      getch();
}
```

Exercise :

i. Write a program to take input of ten numbers and then search another number (given by user) in that list and print all the numbers less than and greater than to that number.

ii. Write a program to take input of minimum and maximum temperature of Gwalior for a week and print the weekly temperature in tabular form. An example of output is shown below:

Weekly Temperature Report

Day	Minimum Temperature	Maximum Temperature
Sunday	25.0	38.0
Monday	23.2	38.1
Tuesday	24.0	37.2
Wednesday	24.4	38.2
Thursday	25.6	39.9
Friday	24.8	42.2
Saturday	25.3	41.7

PROGRAM No. – 9

Objective : Write a program to add two 3x3 matrix.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
# include <iomanip.h>
void main()
{
        int a[3][3], b[3][3], c[3][3], i, j;
        clrscr();
        cout<<endl<<"Input of First Matrix : "<<endl;
        for(i = 0; i <= 2; i++)
        {
                for(j = 0; j <= 2; j++)
                {
                        cout<<"Enter "<<i+1<<"x"<<j+1<<" element : ";
                        cin>>a[i][j];
                }
        }
        clrscr();
        cout<<endl<<"Input of Second Matrix : "<<endl;
        for(i = 0; i <= 2; i++)
        {
                for(j = 0; j <= 2; j++)
                {
                        cout<<"Enter "<<i+1<<"x"<<j+1<<" element : ";
                        cin>>b[i][j];
                }
        }
        for(i = 0; i <= 2; i++)
                for(j = 0; j <= 2; j++)
                        c[i][j] = a[i][j] + b[i][j];
        clrscr();
        for(i = 0; i <= 2; i++)
        {
                for(j = 0; j <= 2; j++)
                        cout<<setw(7)<<a[i][j];
                cout<<endl;
        }
        cout<<endl<<"      +        "<<endl;
        for(i = 0; i <= 2; i++)
        {
                for(j = 0; j <= 2; j++)
                        cout<<setw(7)<<b[i][j];
                cout<<endl;
        }
        cout<<endl<<"      =        "<<endl;
        for(i = 0; i <= 2; i++)
        {
                for(j = 0; j <= 2; j++)
                        cout<<setw(7)<<c[i][j];
                cout<<endl;
        }
        getch();
}
```

Exercise :

i. Write a program to multiply two 3x3 matrix.

PROGRAM No. – 10

Objective : Write a program to check a number weather it is an Armstrong number or not.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
void arms(int);
void main()
{
      int num;
      clrscr();
      cout<<"Enter a number : ";
      cin>>num;
      arms(num);
      getch();
}

void arms(int number)
{
      int a, sum=0, digit;
      a=number;
      while(a > 0)
      {
            digit = a % 10;
            sum = sum + (digit * digit * digit);
            a = a / 10;
      }
      if(sum == number)
            cout<<number<<" is an armstrong number.";
      else
            cout<<number<<" is not an armstrong number.";
}
```

Exercise :

i. Write a program to check a number weather it is prime or not using function.

ii. Write a program to print the square of given number using function.

PROGRAM No. – 11

Objective : Write a program to print an array given by user using function.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
void print_array(int [],int);
void main()
{
      int a[10], i;
      clrscr();
      for (i = 0; i < 10; i++)
      {
            cout<<"Enter element no. "<<i+1<<" of array: ";
            cin>>a[i];
      }
      print_array(a, 10);
      getch();
}


void print_array(int arr[], int size)
{
      int i;
      cout<<endl<<"You Entered "<<endl;
      for(i = 0; i < size; i++)
      {
            cout<<arr[i]<<endl;
      }
}
```

Exercise :

i. Write a program to calculate the average of ten numbers using function.

PROGRAM No. – 12

Objective : Write a program to take input of two numbers and then print them.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class number
{
            int number1, number2;
      public:
            void input()
            {
                  cout<<"Enter First Number : ";
                  cin>>number1;
                  cout<<"Enter Second Number : ";
                  cin>>number2;
            }
            void print()
            {
                  cout<<endl<<"First Number : "<<number1;
                  cout<<endl<<"Second Number : "<<number2;
            }
};

void main()
{
      number a;
      clrscr();
      a.input();
      a.print();
      getch();
}
```

Exercise :

i. Write a program to take input of two numbers and then print the sum of them using class.

PROGRAM No. – 13

Objective : Write a program to take input of Name, Class, Max. Marks (5 Subjects), Marks Obtained in each subject of a student and then calculate the percentage and print the whole data.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <stdio.h>
# include <conio.h>
class student
{
            char name[20]; char subject[5][30];
            int max_marks[5];
            int marks_obtained[5];
            int passing_marks;
            float total_marks_obtained;
            float total_max_marks;
            float percentage;
            int chk;
      public:
            void input();
            void calc_percent();
            void print();
};
void student::input()
{
      chk = 0;
      cout<<"Enter Name : ";
      gets(name);
      for(int i = 0; i < 5; i++)
      {
            cout<<"Enter Name of Subject "<<i + 1<<" : ";
            fflush(stdin);
            gets(subject[i]);
      }
      for(i = 0; i < 5; i++)
      {
            cout<<"Enter Max. Marks of "<<subject[i]<<" : ";
            cin>>max_marks[i];
            cout<<"Enter Marks Obtained for "<<subject[i]<<" : ";
            cin>>marks_obtained[i];
      }
      cout<<"Enter the minimum Passing Marks : ";
      cin>>passing_marks;
}
void student::calc_percent()
{
      total_marks_obtained = 0;
      total_max_marks = 0;
      for(int i = 0;i < 5; i++)
      {
            if(marks_obtained[i] < passing_marks)
                  chk = 1;
            total_marks_obtained += marks_obtained[i];
            total_max_marks += max_marks[i];
      }
      if(chk == 1)
```

```
            percentage = 0;
      else
            percentage = (total_marks_obtained * 100) / total_max_marks;
}

void student::print()
{
      clrscr();
      gotoxy(10,2);
      cout<<"Name   : "<<name;
      gotoxy(10,4);
      cout<<"Subject";
      gotoxy(45,4);
      cout<<"Max.Marks";
      gotoxy(58,4);
      cout<<"Marks Otb.";
      for(int i = 0; i < 5; i++)
      {
            gotoxy(10, 5 + i);
            cout<<subject[i];
            gotoxy(45, 5 + i);
            cout<<max_marks[i];
            gotoxy(58, 5 + i);
            cout<<marks_obtained[i];
      }
      gotoxy(10,12);
      cout<<"Total Marks Obtained : "<<total_marks_obtained;
      gotoxy(10,13);
      cout<<"Total Max. Marks : "<<total_max_marks;
      gotoxy(10,15);
      cout<<"% marks : "<<percentage;
      gotoxy(10,14);
      if (percentage == 0)
            cout<<"Result : Fail";
      else
      {
            if (percentage >= 60)
                  cout<<"Result : Pass in First Division";
            else if (percentage >= 45 && percentage < 60)
                  cout<<"Result : Pass in Second Division";
            else
                  cout<<"Result : Pass in Third Division";
      }
}

void main()
{
      student x;
      clrscr();
      x.input();
      x.calc_percent();
      x.print();
      getch();
}
```

Exercise :

i. Convert above program for input of 10 students and printing only % and result in tabular format.

PROGRAM No. – 14

Objective : Program to demonstrate the use of static member variable.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class test
{
            int x;
            static int y;
      public:
            void input()
            {
                  cout<<"Enter The Value of X : ";
                  cin>>x;
            }
            void display()
            {
                  y++;
                  cout<<endl<<"X = "<<x;
                  cout<<endl<<"Y = "<<y<<endl;
            }
};

int test::y = 0;

void main()
{
      test a, b, c;
      clrscr();
      cout<<"INPUT"<<endl;
      a.input();
      a.display();
      b.input();
      b.display();
      c.input();
      c.display();
      cout<<"VALUE AFTER INPUT"<<endl;
      a.display();
      b.display();
      c.display();
      getch();
}
```

Exercise :

i. Write a program to print Fibonacci Series using Static Variable.

PROGRAM No. – 15

Objective : Program to demonstrate the use of static function.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class test
{
            static int x;
      public:
            static void display()
            {
                  x++;
                  cout<<x;
            }
};

int test::x = 0;

void main()
{
      test a;
      clrscr();
      cout<<endl<<"The Value at the Starting of Program is : ";
      a.display();
      cout<<endl<<"The Value at the Middle of Program is : ";
      a.display();
      cout<<endl<<"The Value at the End of Program is : ";
      a.display();
      getch();
}
```

Exercise :

i. Write a program to print Fibonacci Series using Static Function.

PROGRAM No. – 16

Objective : Program to demonstrate the use of constructor.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class percent
{
            float total_marks;
            float max_marks;
      public:
            percent()
            {
                  cout<<endl<<"Enter Marks Obtained : ";
                  cin>>total_marks;
                  cout<<endl<<"Enter Max. Marks : ";
                  cin>>max_marks;

            }
            void display()
            {
                  cout<<endl<<"Marks Obtained   : "<<total_marks;
                  cout<<endl<<"Out Of           : "<<max_marks;
                  cout<<endl<<"Percentage Marks : "<<(total_marks*100)/max_marks;
            }
};

void main()
{
      clrscr();
      percent a;
      a.display();
      getch();
}
```

Exercise :

i. Write a program to take input of a 2x2 matrix using constructor and then print it.

PROGRAM No. – 17

Objective : Program to demonstrate the use of multiple constructors.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class percent
{
            float total_marks;
            float max_marks;
      public:
            percent(float tm, float mm)
            {
                  total_marks = tm;
                  max_marks = mm;
            }
            percent()
            {
                  cout<<"Enter Marks Obtained : ";
                  cin>>total_marks;
                  cout<<"Enter Max. Marks : ";
                  cin>>max_marks;
            }
            percent(percent &p)
            {
                  total_marks = p.total_marks;
                  max_marks = p.max_marks;
            }
            void display()
            {
                  cout<<endl<<"Marks Obtained   : "<<total_marks;
                  cout<<endl<<"Out Of           : "<<max_marks;
                  cout<<endl<<"Percentage Marks : "<<(total_marks*100)/max_marks;
            }
};

void main()
{
      clrscr();
      cout<<"First Student :"<<endl;
      percent a;
      a.display();
      cout<<endl<<endl<<"Second Student :";
      percent b(432,450);
      b.display();
      percent c(a);
      cout<<endl<<endl<<"Third Student :";
      c.display();
      getch();
}
```

Exercise :

i. Write a program to take input of a 2x2 matrix using constructor and then copy it to another matrix and then print the sum of them.

PROGRAM No. – 18

Objective : Program to demonstrate the use of destructors.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class percent
{
            float total_marks;
            float max_marks;
      public:
            percent(float tm, float mm)
            {
                  total_marks = tm;
                  max_marks = mm;
            }
            percent()
            {
                  cout<<"Enter Marks Obtained : ";
                  cin>>total_marks;
                  cout<<"Enter Max. Marks : ";
                  cin>>max_marks;
            }
            percent(percent &p)
            {
                  total_marks = p.total_marks;
                  max_marks = p.max_marks;
            }
            ~percent()
            {
                  cout<<endl;
                  cout<<endl<<"Marks Obtained   : "<<total_marks;
                  cout<<endl<<"Out Of           : "<<max_marks;
                  cout<<endl<<"Percentage Marks : "<<(total_marks*100)/max_marks;
            }
};

void main()
{
      clrscr();
      percent a;
      percent b(432,450);
      percent c(a);
}
```

Exercise :

i. Write the program of matrix multiplication using destructor.

PROGRAM No. – 19

Objective : Program to overload unary operator (-).

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
# include <stdio.h>
# include <string.h>
class string
{
            char s[10];
            char rs[10];
      public:
            string()
            {
                  cout<<"Enter a String (max. 10 char.) ";
                  gets(s);
            }
            void operator -()
            {
                  int l = strlen(s);
                  int i, j;
                  for (j = 9; j > l - 1; j--)
                        rs[j] = ' ';
                  for (i = 0; i < l; i++)
                  {
                        rs[j] = s[i];
                        j--;
                  }
            }
            void display()
            {
                  cout<<endl<<"Reverse String : ";
                  puts(rs);
            }
};
void main()
{
      clrscr();
      string a;
      -a;
      a.display();
      getch();
}
```

Exercise :

i. Write a program to reverse a number by overloading the unary operator minus (-).

PROGRAM No. – 20

Objective : Program to overload the plus operator (+) to add two matrixes.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
# include <iomanip.h>
class matrix
{
        int val[3][3];
    public:
        void input();       void display();
        matrix operator +(matrix b)
        {
            int i, j;     matrix c;
            for (i = 0; i < 3; i++)
                for (j = 0; j < 3; j++)
                    c.val[i][j] = val[i][j] + b.val[i][j];
            return c;
        }
};
void matrix :: input()
{
    int i, j;
    for ( i = 0; i < 3; i++)
    {
        for ( j = 0 ; j < 3; j++)
        {
            cout<<"Enter Value of Row "<<i+1<<" & Col. "<<j+1<<" : ";
            cin>>val[i][j];
        }
    }
}
void matrix :: display()
{
    int i, j;
    for ( i = 0; i < 3; i++)
    {
        for ( j = 0 ; j < 3; j++)
            cout<<setw(6)<<val[i][j];
        cout<<endl;
    }
}
void main()
{
    matrix a, b, c;
    clrscr();
    cout<<"Input of Matrix A : "<<endl;     a.input();
    clrscr();
    cout<<"Input of Matrix B : "<<endl;     b.input();
    c = a + b;
    clrscr();
    cout<<"Sum of "<<endl;     a.display();
    cout<<endl<<"And"<<endl;   b.display();
    cout<<endl<<"is"<<endl;    c.display();
    getch();
}
```

Exercise :

i. Write a program to add two strings using + operator.

PROGRAM No. – 21

Objective : Program to inherit all public members of one class to other class.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
      class parent
      {
            int a;
      protected:

            float b;
      public:
            void input()
            {
                  cout<<"Enter value of a : ";
                  cin>>a;
                  cout<<"Enter value of b : ";
                  cin>>b;
            }
            void display()
            {
                  cout<<endl<<"The value of a : "<<a;
                  cout<<endl<<"The value of b : "<<b;
            }
      };
      class child : private parent
      {
      public:
            void print()
            {
                  input();
                  display();
            }
      };

      void main()
      {
            child a;
            clrscr();
            a.print();
            getch();
      }
```

Exercise :

i. Write a program to inherit members of two classes and printing them by using only one function.

PROGRAM No. – 22

Objective : Program to demonstrate the use of Constructors and Destructors in Derived Class.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class parent
{
            int a;
      public:
            parent()
            {
                  cout<<"Enter a number : ";
                  cin>>a;
            }
            ~parent()
            {
                  cout<<endl<<"The number in Parent Class : "<<a;
            }
};

class child : private parent
{
            int b;
      public:
            child()
            {
                  cout<<"Enter another number : ";
                  cin>>b;
            }
            ~child()
            {
                  cout<<endl<<"The number in Child Class : "<<b;
            }
};

void main()
{
      clrscr();
      child x;
}
```

Exercise :

i. Write a program to inherit members of more than one class. Each class should have its own constructor and destructor.

PROGRAM No. – 23

Objective : Program to demonstrate the use of this pointer.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class complex
{
            float real;
            float img;
      public:
            void input()
            {
                  cout<<"Enter Real Part : ";
                  cin>>real;
                  cout<<"Enter Imag. Part : ";
                  cin>>img;
            }
            void display()
            {
                  cout<<"["<<real<<" + "<<img<<"i]";
            }
            complex operator +(complex z)
            {
                  this->real = real + z.real;
                  this->img = img + z.img;
                  return *this;
            }
};
void main()
{
      complex x, y, z;
      clrscr();
      x.input();
      y.input();
      z = x + y;
      x.display();
      cout<< " + ";
      y.display();
      cout<<" = ";
      z.display();
      getch();
}
```

Exercise :

i. Write a program to demonstrate the use of ***this*** pointer (other than the program given above).

PROGRAM No. – 24

Objective : Program to demonstrate the use of pointers to object.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class test
{
            int n;
      public:
            test()
            {
                  cout<<"Enter a number : ";
                  cin>>n;
            }
            void display()
            {
                  cout<<endl<<"Entered number : "<<n;
            }
};
void main()
{
      clrscr();
      test *a = new test;
      a->display();
      getch();
}
```

Exercise :

i. Write a program to take input of a complex number and then subtract that number from another complex number using pointer to object.

PROGRAM No. – 25

Objective : Program to demonstrate the use of Virtual Function.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
class A1
{
            int x,y;
      public:
            virtual void input()
            {
                  cout<<"Enter x : ";
                  cin>>x;
                  cout<<"Enter y : ";
                  cin>>y;
            }
};
class A2:public A1
{
            int z;
      public:
            void input()
            {
                  cout<<"Enter z : ";
                  cin>>z;
            }
};
void main()
{
      A1 a;
      A2 b;
      A1 *c = &a;
      clrscr();
      c->input();
      c = &b;
      c->input();
      getch();
}
```

Note: *The above program also illustrate the example of Polymorphism since the same call c->input() invokes more than one function.*

PROGRAM No. – 26

Objective : Program to open and file and printing the contents of that.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
# include <fstream.h>
void main()
{
      ifstream infile("input.txt");
      char *c;
      clrscr();
      while(infile >> c && *c != NULL)
      {
            cout<<c;
            infile.get(*c);
      }
      cout<<endl<<"End of File ";
      getch();
}
```

Exercise :

i. Write a program to read a file and print the contents of that in capital letters.

PROGRAM No. – 27

Objective : Program to write in the file and then read it.

S/W Requirements : Turbo C++ 3.0

Program :

```
# include <iostream.h>
# include <conio.h>
# include <fstream.h>
void main()
{
      ofstream outfile("output.txt");
      char *c;
      clrscr();
      cout<<"Enter a string : ";    cin>>c;
      outfile<<c;
      outfile.close();
      ifstream infile("output.txt");
      cout<<endl<<"Reading File "<<endl;
      while(infile >> c && *c != NULL)
      {
            cout<<c;
            infile.get(*c);
      }
      cout<<endl<<"End of File ";
      getch();
}
```

Exercise :

i. Modify the above program to enter more than one string in the file and then print the contents of that file.

APPENDIX A

C++ Keywords

asm	auto	break	case	cdecl	char
class	const	continue	_cs	default	delete
do	double	_ds	else	enum	_es
extern	_export	far	_fastcall	float	for
friend	goto	huge	if	inline	int
interrupt	_loadds	long	near	new	operator
pascal	private	protected	public	register	return
_saveregs	_seg	short	signed	sizeof	_ss
static	struct	switch	template	this	typedef
union	unsigned	Virtual	void	volatile	while

C++ Operators

::	Global/Class scope resolution	.	Direct member selection	->	Indirect member selection
--	Pre/Post-decrement	sizeof	Size of object or type	~	Bitwise complement
*	Dereference/Multiplication	&	Address	new	Allocation
/	Division	%	Remainder	<<	Bit shift left
>	Greater than	>=	Greater than or equal to	==	Equal to
&&	Logical AND	||	Logical OR	?:	Conditional expression
*=	Multiplication assignment	/=	Division assignment	%=	Remainder assignment
<<=	Bit shift left assignment	>>=	Bit shift right assignment	,	Comma
[]	Subscript	()	Function call	++	Pre/Post-increment
!	Logical NOT	+	Addition/Unary Plus	-	Subtraction/Unary Minus
delete	De-allocation	.*	Direct member selection	->*	Indirect member selection
>>	Bit shift right	<	Less than	<=	Less than or equal to
!=	Not equal to	^	Bitwise XOR	|	Bitwise OR
=	Assignment	+=	Addition assignment	-=	Subtraction assignment
&=	Bitwise AND assignment	^=	Bitwise XOR assignment	|=	Bitwise OR assignment

APPENDIX B

Data Types: integer, float, double and character

Integer : To store integer type values.
Float : To store values containing decimal (floating-point numbers).
Double : To store large values containing decimal (double precision floating-point numbers).
Character : To store a single character. The array of character is called string.

The **long** and **short** are the keywords which can be combined with all above data types as per the requirement. For example an integer may be short or long.

A numeric data type may also be **signed** or **unsigned**.

The maximum value stored by a data type and memory used by the same is given in the following table:

Data Type	Memory Used		Data Storing Capacity
	In bits	In bytes	
unsigned char	8 bits	1 byte	0 to 255
char	8 bits	1 byte	-128 to 127
unsigned int	16 bits	2 bytes	0 to 65,535
short int	16 bits	2 bytes	-32,768 to 32,767
int	16 bits	2 bytes	-32,768 to 32,767
unsigned long	32 bits	4 bytes	0 to 4,29,49,67,295
long	32 bits	4 bytes	-2,14,74,18,31,61,418 to 2,14,74,83,647
float	32 bits	4 bytes	3.4×10^{-38} to $3.4 \times 10^{+38}$
double	64 bits	8 bytes	1.7×10^{-308} to $1.7 \times 10^{+308}$
long double	80 bits	10 bytes	3.4×10^{-4932} to $3.4 \times 10^{+4932}$

There are to more data types in C++:

enum : It is called *enumeration* type data. This data type can be defined by user with following syntax:

enum *typename* {enumerator-list}

Here **enum** is a C++ keyword, *typename* stands for an identifier that names the type being defined, and *enumerator-list* stands for a list of names for integer constants. For example, the following defines the enumeration type Weekdays, specifying the seven possible values that a variable of that type can have:

```
enum Weekdays {SUN, MON, TUE, WED, THU, FRI, SAT}
```

User can declare the variable of this type as:

```
Weekdays w1, w2;
```

and those variable can be used as:

```
w1 = TUE;
w2 = FRI;
if (w1 == w1)
      cout<<"Same Day";
```

Boolean : A *Boolean type* is an integral type whose variables can have only two values: **false** and **true**. These values are stored as the integers 0 and 1. The Boolean type in C++ is named **bool**.

Example:

```
# include <iostream.h>
void main()
{     bool flag = false;
      cout << "flag = "<<flag<<endl;
      flag = true;
      cout << "flag = "<<flag<<endl;
}
```

APPENDIX C

Important mathematics functions.

Syntax	Type	Use
abs(int i)	Integer	Returns the absolute value of i
acos(double d)	Double	Returns the arc cosine of d (in the range of 0 to pi)
acosl(long double l)	Long	Returns the arc cosine of l (in the range of 0 to pi)
asin(double d)	Double	Returns the arc sine of d (in the range of –pi/2 to pi/2)
asinl(long double l)	Long	Returns the arc sine of l (in the range of –pi/2 to pi/2)
atan(double d)	Double	Returns the arc tangent of d (in the range of –pi/2 to pi/2)
atan(long double l)	Long	Returns the arc tangent of l (in the range of –pi/2 to pi/2)
atan2(double d1, double d2)	Double	Returns the arc tangent of d1/d2 (in the range of –pi to pi)
atan2l(long l1, long l2)	Long	Returns the arc tangent of l1/l2 (in the range of –pi to pi)
atof(char *s)	Double	Converts a string to a floating point
cabs (complex z)	Double	Returns the absolute value of complex number z
ceil(double d)	Double	Round up to the next integer value (the smallest integer that is greater than or equal to d)
ceill (long double d)	Double	Round up to the next integer value (the smallest integer that is greater than or equal to d)
cos(double d)	Double	Return the cosine of d
cos (long double l)	Double	Return the cosine of d
cosh(double d)	Double	Return the hyperbolic cosine of d
coshl(long double d)	Double	Return the hyperbolic cosine of d
exp(double d)	Double	Raise *e* to the power d (*e*=2.7182818…… is the base of natural system of logarithms)
exp(long double d)	Double	Raise *e* to the power d (*e*=2.7182818…… is the base of natural system of logarithms)
fabs(double d)	Double	Return the absolute value of d
fabls(long double d)	Double	Return the absolute value of d
floor(double d)	Double	Round down to the next integer value (the largest number that does not exceed d)
floor(long double d)	Double	Round down to the next integer value (the largest number that does not exceed d)

fmod(double d1, double d2)	Double	Return the remainder (i.e., the non-integer part of the quotient) of d1/d2, with same sign as d1
fmodl(long double d1, long double d2)	Double	Return the remainder (i.e., the non-integer part of the quotient) of d1/d2, with same sign as d1
frexp(double d, int *exponent)	Double	Splits a double number into mantissa and exponent
frexp(long double (d), int *(exponent))	Double	Splits a long double number into mantissa and exponent
hypot(double x, double y)	Double	Calculates the hypotenuse of right triangle where $z^2=x^2+y^2$ and z>=0
hypotl(long double x, long double y)	Double	Calculates the hypotenuse of right triangle where $z^2=x^2+y^2$ and z>=0
labs(long int x)	Long int	Returns the absolute value of a long number
log(double d)	Double	Return the natural logarithm of d
logl(long double l)	Long	Return the natural logarithm of l
log10(double d)	Double	Return the base 10 logarithm of d
log10l(long double l)	Double	Return the base 10 logarithm of l
modf(double x, double *ipart)	Double	Splits a double number into integer and fraction part
modfl(long double (x), long double *(ipart))	Double	Splits a long double number into integer and fraction part
poly(double x, int degree, double coeffs[])	Double	Generates a polynomial in x with degree ***degree*** with coefficients coeffs[0], coeffs[1], ……
poly(long double x, int degree, double coeffs[])	Double	Generates a polynomial in x with degree ***degree*** with coefficients coeffs[0], coeffs[1], ……
pow(double d1, double d2)	Double	Return d1 raised to the d2 power
powl(long double d1, long double d2)	Double	Return d1 raised to the d2 power
pow(int p)	Double	Compute the 10^p
sin(double d)	Double	Return the sine of double d
sinl(long double d)	Double	Return the sine of long double d
sinh(double d)	Double	Returns the hyperbolic sine of d
sinhl(long double d)	Double	Returns the hyperbolic sine of d
sqrt(double d)	Double	Return the square root of double number d
sqrt(long double d)	Double	Return the square root of long double number d
tan(double d)	Double	Return the tangent of d
tanl(long double d)	Double	Return the tangent of d
tanh(double d)	Double	Return the hyperbolic tangent of d
tanhl(long double d)	Double	Return the hyperbolic tangent of d

APPENDIX D

Important String functions

Syntax	Type	Use
setmem(void *dest, unsigned length, char value)	Void	Sets a block of **length** bytes, ***dest**, to the byte **value**
strcpy(char *dest, char *source)	Void	Copies the string **source** to string **dest**
strcat(char *dest, char *source)	Void	Appends the string **source** to string **dest**
strchr(char *s, int c)	Char	Scans the first occurrence of given character c in string s
strcmp(char *s1, char *s2)	Integer	Compare two strings s1 and s2 and returns value <0 if s1 < s2 =0 if s1 = s2 >0 if s1 > s2
strcspn(char *s1, char *s2)	Integer	Scan a string for the segment that does not contain a subset of a set of character
strdup(char *s)	Void	Makes a duplicate string of s, obtaining space with a call to **malloc**
strlen(char *s)	Integer	Returns the length of given string s
strlwr(char *s)	Void	Converts string s to lowercase
strset(char *s, char c)	Void	Sets all characters of s to c
strrev(char *s)	Void	Reverses all characters in s
strpbrk(char *s1, char *s2)	Char	Scans first string for the first occurrence of any character in second string
strspn(char *s1, char *s2)	Integer	Scan a string for the segment that is a subset of a set of character
strstr(char *s1, char *s2)	Integer	Finds the first occurrence of a substring into second string.
strupr(char *s)	Void	Converts all characters in s to uppercase

APPENDIX E

Important Character functions

Syntax	Type	Use
isalnum(char c)	Integer	Returns 0 if c is not an alphabet or numeric
isalpha(char c)	Integer	Returns 0 if c is not a letter (A to Z or a to z)
isascii(char c)	Integer	Returns 0 if c is not the low order byte of range 0 to 127
iscntrl(char c)	Integer	Returns 0 if c is not a delete character or ordinary control character
isdigit(char c)	Integer	Returns 0 if c is not a digit
isgraph(char c)	Integer	Returns 0 if c is not a printing character, like isprint, except that a space character is excluded
islower(char c)	Integer	Returns 0 if c is not a lowercase letter (a to z)
isprint(char c)	Integer	Returns 0 if c is not a printing character
ispunct(char c)	Integer	Returns 0 if c is not a punctuation character
isspace(char c)	Integer	Returns 0 if c is not a space or tab or carriage return or new line or vertical tab or form feed character.
isupper(char c)	Integer	Returns 0 if c is not an uppercase letter (A to Z)
isxdigit(char c)	Integer	Returns 0 if c is not a hexadecimal digit (0 to 9, A to F, a to f)
toascii(char c)	Void	Converts c (in the range EOF to 255) to its uppercase value (A to Z; if it was lowercase, a to z). All others are left unchanged
tolower(char c)	Void	Converts c (in the range EOF to 255) to its lowercase value (a to z; if it was uppercase, A to Z). All others are left unchanged
toupper(char c)	Void	Converts the c to ASCII by clearing all but the lower 7 bits. This gives a value in the range 0 to 127.

APPENDIX F

Inline & Friend function

Inline Functions : A function call involves substantial overhead. Extra time and space have to be used to invoke the function, pass parameters to it, allocate storage for its local variables, store the current variables and the location of execution in the main program, etc. In some cases, it is better to avoid all this by specifying the function to be **inline**. This tells the compiler to replace each call to the function with explicit code for the function. A function can be declared inline function by using the keyword **inline** at the starting of function header.

Example :

```
#include <iostream.h>
inline int cube(int x)
{
            return x*x*x;
}
void main()
{
            int a, b;
            cout<<"Enter a number : ";
            cin>>a;
            b=cube(a);
            cout<<"Cube = " <<b;
}
```

Friend Functions : A **friend** of a class X is a function or class that, although not a member of that class, has full access rights to the private and protected members of class X.

A function can be made friend using friend keyword at the starting of declaration as under:

```
friend <identifier>;
```

In all other respects, the friend function is a normal function in terms of scope, declarations, and definitions.

Example :

```
class stars
{
            friend galaxy;
            int magnitude;
            int starfunc(void);
};
class galaxy
{
            long int number_of_stars;
            void stars_magnitude(stars&);
            void stars_func(stars*);
}
```