PRACTICAL 1

1a) Write C++ program to create a simple calculator.

→

```
#include <iostream>
using namespace std;
int main()
{
    int a,b,res,ch;
    cout<<"Enter a:";
    cin>>a;
    cout<<"Enter b:";
    cin>>b;
    cout<<"Your choices are:\n";
    cout<<"1.Addition\n";
    cout<<"2.Subtraction\n";
    cout<<"Enter your choice:";
    cin>>ch;
    switch(ch)
    {
        case 1:
        res = a+b;
        cout<<"Addition is "<<res<<endl;
        break;
```

```
        case 2:
        res = a-b;
        cout <<" Substraction is " << res << endl;
        break;

        default:
        cout<< "Invalid choice";
    }
    return 0;
}
```

1b) Write C++ program to convert seconds into hours, minutes & seconds

→

```
#include<iostream>
using namespace std;
int main()
{
    int sec, min, hour;
    cout<<"Enter seconds:";
    cin>>sec;
    min = sec/60;
    hour = min/60;
    cout<<"Time is "<<hour<<"hour"<<int(min%60)<<
    "minutes"<<int(sec%60)<<"seconds"<<endl;
    return 0;
}
```

1c) Write c++ program to find the volume of a square, cone & rectan

→
```
#include <iostream>
using namespace std;
int main()
{
    float s, v1, r, h, v2, l, w, h1, v3;
    cout<<"Enter side of square:";
    cin>>s;
    v1=s*s*s;
    cout<<"Volume of square is "<<v1<<endl;
    cout<<"In Enter radius of cone:";
    cin>>r;
    cout<<"Enter height of cone:";
    cin>>h;
    v2=0.33*3.14*r*r*h;
    cout<<"Volume of cone is "<<v2<<endl;
    cout<<"In Enter length of rectangle:";
    cin>>l;
    cout<<"Enter width of rectangle:";
    cin>>w;
    cout<<"Enter height of rectangle:";
    cin>>h1;
    v3=l*w*h1;
    cout<<"Volume of rectangle is "<<v3;
    return 0;
}
```

2a) →

PRACTICAL 2

2a) Write a c++ program to find the greatest of three numbers.

→

```
#include <iostream>
using namespace std;
int main()
{
    int a,b,c,x,y,z;
    cout<<"Enter a:";
    cin>>a;
    cout<<"Enter b:";
    cin>>b;
    cout<<"Enter c:";
    cin>>c;
    x=(a>b)?a:b;
    y=(b>c)?b:c;
    z=(x>y)?x:y;
    cout<<"The greatest number is "<<z;
    return 0;
}
```

2b) Write C++ program to find the sum of even & odd n natural

→
```
#include<iostream>
using namespace std;
int main()
{
    int n,i, even=0, odd=0;
    cout<<"Enter n:";
    cin>>n;
    for(i=1; i<=n; i++)
    {
        if(i%2==0)
        {
            even = even + i;
        }
        else
        {
            odd = odd + i;
        }
    }
    cout<<"Addition of even no: "<<even<<endl;
    cout<<"Addition of odd no: "<<odd;
    return 0;
}
```

2c) Write C++ program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.

→

```
#include<iostream>
using namespace std;
int main()
{
    int n,i,j,count;
    cout<<"Enter n:";
    cin>>n;
    for(i=1;i<=n;i++)
    {
        count=0;
        for(i=1;i<=n;i++)
        {
            if(i%j==0)
            {
                count++;
            }
        }
        if(i==1||count==2)
        {
            cout<<i<<endl;
        }
    }
    return 0;
}
```

PRACTICAL - 3

3a) Write a C++ program using classes and object Student to print name of the student, roll no. Display the same.

→

```
#include<iostream>
#include<conio.h>
using namespace std;
class student
{
   public:
    int rno;
    char name[50];
    void getdata()
     {
        cout<<"Enter roll no:";
        cin>>rno;
        cout<<"Enter name:";
        cin>>name;
     }
     void display()
      {
        cout<<"Roll no is "<<rno<<endl;
        cout<<"Name is "<<name;
      }
};
```

```
void main()
{
    student obj;
    obj.getdata();
    obj.display();
}
```

3b.) Write a C++ program for structure bank employee to print name of the employee, account_no and balance. Display the same, also display the balance after withdrawal & deposit.

→
```
#include<iostream>
#include<conio.h>
using namespace std;
class bank
{
    public:
     int acc_no, bal, amt, ch;
     char name[50];
     void getdata()
      {
        cout<<"Enter account number:";
        cin>>acc_no;
        cout<<"Enter name:";
        cin>>name;
        cout<<"Enter balance:";
        cin>>bal;
      }
     void display()
      {
        cout<<"Account no.:"<<acc_no<<endl;
        cout<<"Username:"<<name<<endl;
        cout<<"Account balance:"<<bal<<endl;
      }
```

```
void transaction()
{
cout<<"Select options given below\n";
cout<<" 1. Deposit "<<endl;
cout<<" 2. Withdraw "<<endl;
cout<<" Enter your choice (1/2) ";
cin>>ch;
switch(ch)
{
case 1:
cout<<" Enter amount to be deposited: ";
cin>>amt;
bal = bal + amt;
cout<<" Balance = "<<bal<<endl;
break;

case 2:
cout<<" Enter amount to be withdrawn: ";
cin>>amt;
if(amt > bal)
{
cout<<" Insufficient balance\n";
}
else
{
bal = bal - amt;
cout<<" Balance = "<<bal<<endl;
}
```

```
                    break;

                    default:
                    cout << " Option not available";
                }
            }
};
void main()
{
    bank obj;
    obj.getdata();
    obj.display();
    obj.transaction();
}
```

3d) Write a program to find maximum out of two numbers using friend function. Here one number is a member of one class & the other number is member of other class.

→

```
#include<iostream>
#include<conio.h>
using namespace std;
class B;
class A
{
    public:
     int a;
     void get1()
      {
          cout<<"Enter a:";
          cin>>a;
      }
     friend void compare(A obj1, B obj2);
};
class B
{
   public:
    int b;
    void get2()
     {
        cout<<"Enter b:";
        cin>>b;
     }
```

```
        friend void compare (A obj1, B obj2);
};
void compare (A obj1, B obj2)
{
   if (obj1.a > obj2.b)
   {
      cout<<"A is greater";
   }
   else
   {
      cout<<"B is greater";
   }
}
void main()
{
   A obj1;
   B obj2;
   obj1.get1();
   obj2.get2();
   compare (obj1, obj2);
}
```

3f) Write a C++ program to allocate memory dynamically for an object of a given class using class' constructor.

→
```
#include<iostream>
#include<conio.h>
using namespace std;
class A
{
  public:
   char *p;
   A()
   {
      p= new char[10];
      p="Nehal";
   }
   void display()
   {
      cout<<"Name is "<<p<<endl;
   }
};
void main()
{
  A obj;
  obj.display();
}
```

PRACTICAL - 4

4a) Write a c++ program to design a class representing complex numbers and having the functionality of performing addition and multiplication of two complex numbers using operator overloading.

→

```
#include<iostream>
#include<conio.h>
using namespace std;
class A
{
    public:
    int a,b;
    A(int a1, int b1)
    {
        a=a1;
        b=b1;
    }
    A(){}
    A operator + (A obj2)
    {
        A obj3;
        obj3.a= a+obj2.a;
        obj3.b= b+obj2.b;
        return obj3;
    }
```

```
int display1()
{
    cout << "A = " << a << "\t B = " << b << endl;
}
A operator * (A obj2)
{
    A obj4;
    obj4.a = a * obj2.a;
    obj4.b = b * obj2.b;
    return obj4;
}
int display2()
{
    cout << "A = " << a << "\t B = " << b << endl;
}
};
int main()
{
    A obj1(1,2);
    A obj2(3,4);
    A obj3;
    obj3 = obj1 + obj2;
    obj1.display1();
    obj2.display1();
    cout << "After After addition" << endl;
    obj.display1();
    A obj4;
```

```
    obj4 = obj1 * obj2;
    cout << " After multiplication" << endl;
    obj4.display2();
}
```

4c) Write a C++ program to access members of a student class using pointer to object members.

→
```
#include<iostream>
#include<conio.h>
using namespace std;
class student
{
    public:
    int rno;
    void get()
    {
        cout<<"Enter roll no:";
        cin>>rno;
    }
    void display()
    {
        cout<<"Roll no is "<<rno;
    }
};
void main()
{
    student obj, *s;
    s= &obj;
    s->get();
    s->display();
}
```

4d) Write a c++ program to generate fibonacci series by using constructor to intitialize the data members.

→
```
#include<iostream>
#include<conio.h>
using namespace std;
class A
{
  public:
   int a,b,c,i;
   A()
   {
     a=0;
     b=1;
   }
   void fibonacci(int n)
   {
     cout<<"Fibonacci series is as follows:\n";
     cout<<a<<"\n"<<b<<"\n";
     for(i=1;i<=n;i++)
     {
       c=a+b;
       cout<<c<<"\n";
       a=b;
       b=c;
     }
   }
{
```

~~void main(■)~~
~~{~~

```
void main()
{
    A obj;
    obj.fibonacci(10);
}
```

Write a C++ program to illustrate single inheritence.

```
#include<iostream>
#include<conio.h>
using namespace std;
class A
{
  public:
   int a;
   void get1()
    {
      cout<<"Enter a:";
      cin>>a;
    }
};
class B : public A
{
  public:
   int b;
   void get2()
    {
      cout<<"Enter b:";
      cin>>b;
    }
   void add()
    {
      cout<<"A+B="<<(a+b);
    }
};
```

```
void main()
{
    B obj;
    obj.get1();
    obj.get2();
    obj.add();
}
```

4f) Write a C++ program that illustrates multiple inheritence.

→
```
#include<iostream>
#include<conio.h>
using namespace std;
class A
{
  public:
   int a;
   void get1()
    {
      cout<<"Enter a:";
      cin>>a;
    }
};
class B
{
   public:
    int b;
    void get2()
     {
       cout<<"Enter b:";
       cin>>b;
     }
};
```

```
class C : public A, public B
{
  public :
  void add()
  {
    cout << "A+B=" << (a+b);
  }
};
void main().
{
  C obj;
  obj.get1();
  obj.get2();
  obj.add();
}
```

4q) Write a C++ program that illustrates multi level inheritence.

→
```
#include <iostream>
#include<conio.h>
using namespace std;
class A
{
  public:
  int a;
  void get1()
  {
    cout<<"Enter a:";
    cin>>a;
  }
};
class B : public A
{
  public:
  int b;
  void get2()
  {
    cout<<"Enter b:";
    cin>>b;
  }
};
```

```
class C : public B
{
    public:
    void add()
    {
        cout << "A+B= " << (a+b);
    }
};
void ~~ad~~ main()
{
    C obj;
    obj.get1();
    obj.get2();
    obj.add();
}
```

4h) Write a program that illustrates hierarchical inheritence.

→
```
#include<iostream>
#include<conio.h>
using namespace std;
class A
{
  public:
   int a;
   void get1()
   {
     cout<<"Enter a:";
     cin>>a;
   }
};  ~~voi~~
class B: public A
{
  public:
   int b;
   void get2()
   {
     cout<<"Enter b:";
     cin>>b;
   }
   void add()
   {
     Cout<<"A+B="<<(a+b);
   }
};
```

```
class C: public A
{
  public:
  int c;
  void get3()
  {
    cout<<"Enter c:";
    cin>>c;
  }
  void sub()
  {
    cout<<"A-C="<<(a-c);
  }
};
void main()
{
  B obj1;
  obj1.get1();
  obj.get2();
  obj1.add();
  C obj2;
  obj2.get1();
  obj2.get3();
  obj2.sub();
}
```

4) Write a C++ program illustrating how the constructors are implemented and the order in which they are called when the classes are inherited

```
#include<iostream>
#include<conio.h>
using namespace std;
class alpha
{
  public:
   int a;
   alpha(int a1)
   {
      a= a1;
   }
   void disp1()
   {
      cout<<"Alpha class"<<endl;
      cout<<"A= "<<a<<endl;
   }
};
class beta
{
  public:
   int a,b;
   beta(int a1, int b1)
   {
      a=a1;
      b=b1;
   }
```

```
    void disp2()
    {
      cout<<"Beta class"<<endl;
      cout<<"A="<<a<<"\t B="<<b<<endl;
    }
};
class gamma : public alpha, public beta
{
  public:
    int c;
    gamma(int a, int b, int c1) : alpha(a), beta(a,b)
    {
      c = c1;
    }
    void disp3()
    {
      cout<<"Gamma class"<<endl;
      cout<<"c="<<c<<endl;
    }
};
void main()
{
  gamma obj(10,20,30);
  obj.disp1();
  obj.disp2();
  obj.disp3();
}
```

PRACTICAL - 5

5a) Write a c++ program to design a student class representing student roll no, & a test class (derived class of student) representing the scores of the students in various subjects sports class representing the score in sports. The sports & test class should be inherited by a result class having the functionality to add the scores & display the final result for a student.

→
```
#include <iostream>
#include <conio.h>
using namespace std;
class student
{

   public;
   int rno;
   void getI()
    {

      cout << "Enter roll no:";
      cin >> rno;
    }
};
class test : public student
{

   public :
   int m1, m2;
    void get2()
    {

      cout << "Enter m1:";
```

```
        cin>>m1;
        cout<<"Enter m2:";
        cin>>m2;
    }
};
class sports
{
    public:
    int sc;
    void get3()
    {
        cout<<"Enter score:";
        cin>>sc;
    }
};
class result : public test, public sports
{
    public:
    void display()
    {
        cout<<"Final result of roll no "<<rno<<" is "<<
        (m1+m2+sc);
    }
};
```

```
void main()
{
    result obj;
    obj.get1();
    obj.get2();
    obj.get3();
    obj.display();
}
```

PRACTICAL - 7

7a) Write a c++ program illustrating the use of virtual functions in class.

→
```
#include <iostream>
#include <conio.h>
using namespace std;
class A
{
    public :
    virtual void show()
     {
        cout << " Class A show" << endl;
     }
};
class B : public A
{
    public :
     void show()
      {
        cout << "Class B show" << endl;
      }
};
```

```
void main()
{
    A obj1, *ptr;
    B obj2;
    ptr = &obj1;
    ptr -> show();
    ptr = &obj2;
    ptr -> show();
}
```