Raja N.L.Khan Women's College(Autonomous)
Department of BCA
4th Semester
Paper: BCA2296
Prepared By: Moumita Mondal
C++ LAB.

1. **<u>Write a program to implement hybrid inheritance</u>.**

```
#include<iostream>
using namespace std;
class student
      {
protected:
int roll;
public:
void get_roll(int x)
{
roll=x;
}
void put_roll()
{
cout<<"Roll No. :"<<roll<<endl;
 }
    };
class test: public student
{
protected:
int paper1,paper2;
public:
void get_marks(int a,int b)
 {
        paper1=a;
        paper2=b;
 }
void put_marks()
 {
cout<<"Marks in paper1:"<<paper1<<endl;
cout<<"Marks in paper2:"<<paper2<<endl;
 }
};
class activity
{
protected:
int score;
public:
void get_score(int a)
{
score=a;
}
void put_score()
  {
cout<<"Score="<<score<<endl;
```

```
   }
};
class result:public test,public activity
{
int total;
public:
void display();
};
void result:: display()
{
total=paper1+paper2+score;
put_roll();
put_marks();
put_score();
cout<<"Total Marks="<<total;
}
int main()
{
result p;
p.get_roll(10);
p.get_marks(50,60);
p.get_score(9);
p.display();
}
```

2. **Write a program to implement virtual base class.**

```
#include<iostream>
using namespace std;
class student
         {
protected:
int roll_no;
public:
void get_roll();
void put_roll();
       };
void student:: get_roll()
       {
cout<<"enter the roll\n";
cin>>roll_no;
        }
void student:: put_roll()
          {
cout<<"Roll is:"<<roll_no<<endl;
          }
class test: virtual public student
          {
protected:
int paper1,paper2;
```

```
public:
void get_marks();
void put_marks();
          };
void test:: get_marks()
           {
cout<<"enter the marks\n";
cin>>paper1>>paper2;
           }
void test:: put_marks()
           {
cout<<"Marks in paper1:"<<paper1<<endl;
cout<<"Marks in paper2:"<<paper2<<endl;
           }
class sports: public virtual student
             {
protected: int score;
public:
void get_score();
void put_score();
             };
void sports:: get_score()
              {
cout<<"enter the score\n";
cin>>score;
}
void sports:: put_score()
{
cout<<"Score="<<score<<endl;
}
class result: public test,public sports
{
private:
int total;
public:
void display();
};
void result:: display()
  {
total=paper1+paper2+score;
put_roll();
put_marks();
put_score();
cout<<"Total Score="<<total<<endl;
}
int main()
{
```

```
result p;
p.get_roll();
p.get_marks();
p.get_score();
p.display();
return 0;
}
```

3. **<u>Write a program to find the volume of cube ,cylinder , rectangular box using function overloading</u>.**

```
#include<iostream>
using namespace std;
int  volume(int);
float  volume(float , int);
long  volume(float, int, int);
int main()
     {
cout<<"volume of cube="<<volume(5)<<endl;
cout<<"volume of cylinder="<<volume(8.2,7)<<endl;
cout<<"volumn of rectangular box="<<volume(5.2,25,15)<<endl;
      }
int  volume(int  p)
     {
return(p*p*p);
     }
float  volume(float r, int  h)
     {
return(3.14*r*r*h);
     }
 long volume(float l, int  b, int  h)
     {
return(l*b*h);
     }
```

4. **<u>Write a program to add two complex number using friend function.</u>**

```
#include<iostream>
 using namespace std;
 class complex
 {
 int  real , imag;
 public:

void input()
 {
  cout<<"Enter real and imag part:";
  cin>>real>>imag;
```

```
    }

    friend complex  sum(complex, complex);
     void display();
    };

    void complex::display()

       {
      cout<<"The sum of complex number is:"<<real<<"+i"<<imag;
        }
     complex sum(complex a, complex b)
    {
     complex t;
     t.real=a.real+b.real;
     t.imag=a.imag+b.imag;
     return t;
    }
     int main()
    {
   complex a,b,c;
   a.input();
   b.input();
   c=sum(a,b);
   c.display();
  return(0);
}
```

5. **<u>Write a program to use common friend function to swap private data of two classes</u>.**

```
#include<iostream>
using namespace std;
class Y;
class X
  {
int a;
public:
void input(int i)
   {
     a=i;
   }
void display(void)
   {
cout<<"a="<<a<<"\n";
   }
friend void swap(X&,Y&);
};
class Y
  {
```

```
int b;
public:
void input(int i)
  {
    b=i;
  }
void display(void)
  {
cout<<"b="<<b<<"\n";
  }
friend void swap(X&,Y&);
};
void swap(X&  m ,Y& n)
 {
int  temp;
temp=m.a;
m.a=n.b;
n.b=temp;
}
int main()
 {
 X c1;
Y c2;
c1.input(20);
c2.input(40);
cout<<"Values before exchange"<<endl;
c1.display();
c2.display();
exchange(c1,c2);
cout<<"Values after exchange"<<endl;
c1.display();
c2.display();
return 0;
}
```

6. Write a program to generate Fibonacci series using constructor.

```
#include<iostream>
#include<iomanip>
using namespace std;
class fibo
{
        int f1,f2;
public:
        fibo()
        {
                f1 = 0;
                f2 = 1;
        }
        void increment(int);
};

void fibo::increment(int n)
{
        int i,f3;
        cout<< setw(4)<<f1<<setw(4)<< f2;
        for(i=1; i <= n-2; i++)
              {
                f3 = f1 + f2;
                cout << setw(4) <<f3;
                f1 = f2;
                f2 = f3;
        }
}

int main()
{
        fibo fib;
        int n;
        cout << "Enter how many terms:";
        cin >> n;
        cout << "Fibonacci series are:\n";
        fib.increment(n);
}
```

7. **<u>Write a program to implement copy constructor</u>.**

```
#include<iostream>
using namespace std;
class integer
{
private:
    int m,n;
public:
integer ( int a , int  b)      //parameterized constructor
     {
      m=a;
      n=b;
     }
integer (integer &i)    //copy constructor
     {
      m=i.m;
      n=i.n;
     }
void display ()
     {
    cout<<"\nValue of m="<<m;
```

```
     cout<<"\nValue of n="<<n;
      }
};
int main()
{
integer x(5,10);
integer y(x);
x.display();
cout<<"\ncopy the same value";
y.display();
}
```

8. **Write a program to implement overloaded constructor.**

```
#include <iostream>
using namespace std;
class Area
{
  private:
  int area;
  public:
  Area()          // constructor with no argument
  {
    area =0;
  }
  Area(int a)     // constructor with one  argument
  {
    area = a * a;
  }

  Area(int a, int  b)    // constructor with two arguments
  {
    area = a * b;
  }
  void display()
  {
    cout << "The area is: " << area << endl;
  }
};

int main()
{
  Area a1;
  Area a2(6);
  Area a3(4,6);
  a1.display();
  a2.display();
  a3.display();
}
```