Page No.
Date

Experiment -5

a) WAP to find sum of nos b/w 1 to n using a constructor where value of n will be passed to the constructor

-> • using default constructor:

```
#include <iostream>
using namespace std;
{
 int n,s;
 Public:
 sum()
{
  n=10;
}
void calculate()
{
 for (int i=1 ; i<=n; i++)
 {
     s = s+i;
  }
}
void display()
{
cout<<" the sum from 1 to n is :" <<s;
 }
};
int main ()
{
sum s1;
s1. calculate()
s1. display ()
```

```
return 0;
}
```

- Output :

The sum from 1 to n is : 55

- by using parameterized contructor :

```
#include <iostream>
using namespace std;
class sum
{
int n, s = 0;
Public :
sum (int num)
{
    n = num;
}
void calculate()
{
for (int i=1 ; i<=n ; i++)
{
    s=s+i;
}
}
Void display()
{
cout << " the sum from 1 to n is :" << s ;
}
};
```

```
int main()
{
 sum s1(10);
 s1.calculate();
 s1.display();
 return 0;
}
```

- Output:

The sum from 1 to 10 is :55

c) using copy constructor

```
#include <iostream>
using namespace std;
class sum
{
 int n,s=0;
public:
 sum(int num)
{
 n=num;
}
 sum(sum &s1)
{
 n=s1.n;
 for(int i=1; i<=n; i++)
{
  s=s+n;
}
 cout<<"the sum from 1 to n is:"<<s;
}
```

```
};

int main()
{
  sum s1(10);
  sum s2(s1);
  return 0;
}
```

- Output :

the sum from 1 to n is : 55

b) WAP to delare a class "student" having data members as name & percentage. Write a constructor to initialize these data members. Accept & display data for one student.

-> - by default constructor :

```
#include <iostream>
using namespace std;
class student{
  string name;
  float percentage;
  Public:

student()
{
  name = " ";
  percentage = 0.0;
}
```

```
void accept()
 {
cout << "enter name:";
cin >> name;
cout << " enter percentage:";
cin >>percentage;
}
void display()
{
cout <<"\n student Name:" <<name;
cout <<"\n Percentage:" << percentage << "%.";
}
};
int main()
{
student s;
s.accept();
s.display();
 return 0;
}
```

- Output:

```
Enter name: Aseeda
Enter percentage: 85.6

Student name: Aseeda
 Percentage : 85.6%.
```

- Parameterized constructor:

```
#include <iostream>
using namespace std;
class student{
 string name;
 float percentage;

Public :
     student (string n, float p)
  {
     name = n;
     percentage = p;
  }
 void display()
{
cout<<" \n student Name :" << name;
cout<<" \n Percentage  :" << percentage <<"%.";
}
};

int main ()
{
 student s (" Aseeda", 85.6);
 s.display();

 return 0;
}
```

- Output:

```
student Name: Aseeda
 Percentage : 85.6%.
```

- Copy constructor:

```
# include <iostream>
using namespace std;
class student
{
string name;
float percentage;
public:

student (string n, float p)
{
 name = n;
 percentage = p;
}
 student (const student &s)
 {
   name = s.name;
   percentage = s.percentage;
 }
 void display()
 {
 cout << " student Name:" << name;
 cout << " Percentage:" << percentage << "%.";
}
};
 int main()
{
 student s1 ("Aseeda", 85.6);
 student s2 (s1);

 s1. display();
 s2. display();
```

```
return 0;
}
```

- output :

```
student Name : Aseeda
percentage : 85.6%.

student Name : Aseeda
Percentage : 85.6%.
```

c) Define a class 'college' variables as roll-no, name, course. WP using constructor with default value as "computer engineering" for course. Accept this data for 2 objects of class & display the data.

→ • default constructor :

```
#include <iostream>
using namespace std;

class college {
   string name;
   string course;

Public:
   college() {
    course = "computer Engineering:"

void accept()
  {
```

```
cout << " enter student name:";
cin >> name;
}

void display()
{
cout << " Name:" <<name;
cout << " course:" <<course << endl;
}
};
int main ()
{
college c1,c2;

c1. accept();
c2. accept();

c1. display();
c2. display();
return 0;
}
```

output:

enter student name: Aseeda
enter student name: Tehsin

Name: Aseeda
course: computer Engineering

Name: ~~Return~~ Tehsin
course: computer Engineering.

- Parameterized

```
#include <iostream>
using namespace std;

class college {
    string name;
    string course;

public:
    college( string n, string c = "computer Engineering")
    {
    name = n;
    course = c;
}
void display()
{
cout << " Name: " << name;
 cout << " course : " << course << endl;
}
};
int main() {
 college c1 ("Aseeda")
 college c2("Tehsin");

 c1.display();
 c2.display();
}
```

Output:

Name: Aseeda
course: computer engineering

Name: tehsin
course: computer engineering.

- copy constructor:

```
#include <iostream>
using namespace std;

class college
{
 string name;
 string course;

public:
 college (string n, string c = "computer Engineering")
 {
  name = n;
  course = c;
}
college (const college &cl)
 {
  name = cl.name;
  course = cl.course;
 }
void display()
{
cout<<" name :"<<name;
cout << " course:"<<course<< endl;
}
};
int main()
{
```

```
college c1("Aseeda");
college c2(c1); (Tehsin) = (c1);

c1.display();
c2.display();

}
```

- Output:

```
Name: Aseeda
course: computer Engineering

Name: Tehsin Aseeda
course: computer engineering
```

d Write a program to demonstrate constructor overloading.

->
```
#include <iostream>
using namespace std;

class student
{
  string name;
  string course;

Public:
    Student()
    {
      name = "Tehsin";
      course = "computer Engineering";
    }
```

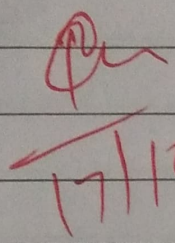
```
student (string n, string c)
{
 name=n;
 course=c;
}

student (const student &s)
{
 name=s.name;
 course = s.course;
}

void display()
{
cout << "Name:" <<name;
cout<<" course:" << course <<endl;
}
};
int main()
{
student s1;
student s2 ("Aseeda", "IT");
student s3 (s2);

  s1.display();
  s2.display ();
  s3.display();
}
```

- Output:

Name : tehsin
course: computer Engineering

Name : Aseeda
course : IT

Name : Aseeda
course : IT



Experiment - 6

a) WAP to implement multilevel inheritance. Assume suitable

→

```
#include <iostream>
using namespace std;
class person {
public:
  string name;
  void getName() {
    cout<<" enter name:";
    cin>>name;
  }
};
class student: public person {
public:
  int RollNo;
  void getRollNo() {
  cout<<" enter roll number:";
  cin>>roll no;
  }
};
class Marks : public student {
  public:
    int marks;
    void getmarks() {
    cout << " enter marks:";
    cin>>marks;
  }
  void display() {
  cout<<" \n Name :" <<name;
  cout<<" \n Roll no:"<<roll No;
  cout<<" \n marks:" <<marks<<" %." <<endl;
  }
```

```
};

int main () {
    marks m;
    m.getName ();
    m.getRollNo ();
    m.getMarks();
    m.display();
}
```

- Output :

```
Enter name :    Aseeda
Enter Roll number : 32
Enter marks   : 85

Name    : Aseeda
Roll No  :  32
Marks :  85%.
```

b) WAP to implement multiple inheritance. Assume suitable data.

→

```
# include <iostream>
using namespace std;
class student {
 Public:
   void getstudentInfo ()
{
 cout<<" Name: ";
 string name;
 int rollno;
 Public:
```

```
void getstudentinfo()
{
 cout<<" enter name:";
 cin>>name;
 cout<<" enter roll number:";
 cin>>rollno;
 }
};
class marks {
 Protected:
  int m1, m2, m3;
 Public:
  void getmarks()
 {
 cout<<" enter marks in 3 subjects:";
 cin >>m1 >>m2 >>m3;
 }
};
class Result : Public student, Public marks {
  Public:
   void displayResults() {
    int total = m1+m2+m3;
    float Percentage = total / 3.0;
     cout<<"\n --- Result ";
     cout<<"Name:"<<name<<endl;
     cout<<" RollNo :"<<rollno<<endl;
     cout<<" total marks:"<<total<<endl;
     cout<<"percentage:"<<percentage<<"%."<<endl;
    }
 };
int main() {
 Result r;
```

```
r. getstudentinfo();
r. get marks();
r. displayResult();
return 0;
}.
```

- Output:

```
Enter name: Aseeda
Enter roll number: 101
Enter marks in 3subjects : 85 90 88


Name : Aseeda
Roll No : 101
total marks: 263
Percentage: 87.6677.
```

c) by heirarchical inheritance.

→

```
# include <iostream>
using namespace std;

Class student {
 Public:
 string name;
 int roll;
 void getdata () {
  cout << "enter name and roll no:";
  cin >> name >> roll;
  }
};
```

```
class Marks : Public student {
Public:
  int m1, m2;
  void getMarks() {
  cout << "enter 2 subject marks:";
  cin >> m1 >> m2;
  cout << "total marks =" m1 + m2 << endl;
  }
};
class sports : Public student {
Public :
  int score;
  void getscore() {
   cout << "enter sports score:";
   cin >> score;
   cout << "sports score =" << score << endl;
   }
};
int main () {
 marks m;
 sports s;

m.getData();
m.getMarks();

s.getData();
s.getMarks();
}
```

Output :

```
Enter name and roll no  :  Aseeda  32
Enter 2 subject marks   :  87  65
Total Marks             :  152
Enter name and roll no  :  abc  56
Enter sports score      :  87
sports score            :  87.
```

d. By hybrid inheritance.

→

```
#include <iostream>
using namespace std;
class student{
 Public:
   int roll;
   void input (){
      cout << " enter Roll NO:";
      cin >> roll;
     }
};
class marks: Public student{
 Public:
   int m1, m2;
   void getmarks(){
     cout<<" enter two subject marks:";
     cin >> m1 >>m2;
    }
};
class sports {
Public:
   int score;
```

```
void getscore() {
    cout << "enter sports score:";
    cin >> score;
    }
};

class Result : Public Marks , Public sports {
Public:
    void display() {
        cout << "\n Roll NO :" << roll;
        cout << "\n total =" << (m1 + m2 + score) << endl;
        }
};
int main() {
 Result r;
 r.input();
 r.getMarks();
 r.getscore();
 r.display();
}
```

- Output :

```
Enter roll NO          : 5
Enter two subject marks : 78  85
Enter sports score     : 10

Roll no                : 5
total                  : 173
```

e WAP to demonstrate virtual base class. Assume suitable data with figures.

→

```
#include <iostream>
using namespace std;

class person {
 Public:
    string name;
};

class student : virtual Public person {
   Public:
     int roll;
};

class Teacher : virtual Public person {
 Public:
    string subjects;
};

class TA : public student, Public teacher { };
int main() {
  TA ta;
  ta.name = "Aseeda";
  ta.roll = 32;
  ta.subject = "Maths";

  cout << " TA Details: \n";
  cout << "Name:" << ta.name << endl;
  cout << "Roll no:" << ta.roll << endl;
  cout << "subject:" << ta.subject << endl;
}
```

TA details :

Name : Aseeda

Roll NO : 32

Subject : Maths.

17/10