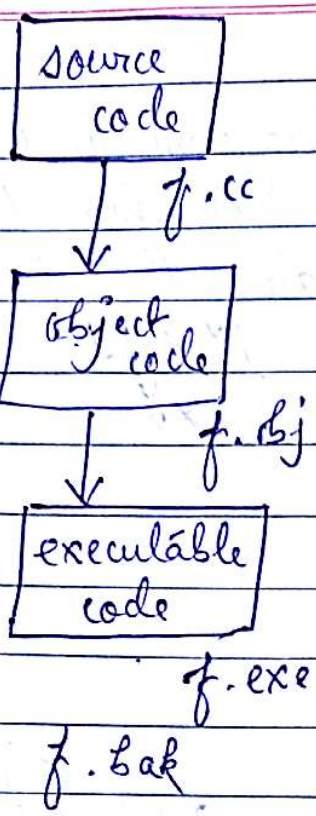
Q= Write a function program in C to find factorial of a number. Use this function to calculate the following expression — $Y = \frac{!x + !y}{x^y}$

→
```
#include <stdio.h>
#include <conio.h>
void main()
{
int x, y, float Y;
clrscr(); int fact(a) → function prototype
printf("Enter the values");
scanf("%d %d %d", &x, &y);
Y = (fact(x) + fact(y)) / (Pow(x,y));

printf("\n Y=", Y); getch(); m! + n! / m^n + n^n
}
int fact (int a)
{ int i, y=1;
for (i=1; i<=a; i++)
  {
    f = f * i;
  }
return ();
}
```

((fact(m) + fact(n)) / (sum(m) + sum(m));

*




Q= Write a function program to find –
(i) The area of circle
(ii) Area of rectangle.
(iii) Surface area of cuboid.

→
```
#include<stdio.h>
#include<conio.h>
void main()
{
int r, l, b, h;
int cir(int)
int rec(int, int);
int cub(int, int, int);
float A;
printf("Enter the radius of circle");
scanf("%d", &r);
```

```
A = cir(r);
printf("Area is %f", A);
printf("Enter l, b of rectangle");
scanf("%d %d", &l, &b);
A = rec(l,b);
printf("Area of rect %f", A);
printf("Enter l, b, h of cuboid");
scanf("%d %d %d", &l, &b, &h);
A = cub(l,b,h);
printf("Area of cuboid is %f", A);
float cir(int a)
{
float Area;
Area = 3.14 * a * a;
return Area;
}
int rec(int e, int f)
{
return e*f;
}
int cub(int g, int h, int i).
{
return 2*(l*b + b*h + h*l);
}
```

Q= Rewrite the area calculation program using function overloading.

Q= Using function overloading write function program to calculate the SI and compound interest on a given principle amount.

Q:- Create a class student to define the telephone directory of the students. Write main() to create the object of the class, give same input to the class and

→ class class_name
{
Attributes } Data members ← private
↑
Member function ← public

Class encapsulation

```
class student ← class
{
  int roll
  char name [50]
public :
    void input()
    {- - - - -
    - - - - -
    }
    void show()
    {- - - -
     - - - -
    }
};
```

void main() objects

{

student S1, S2; , or student S1, S2[50];

S1 input();

S1 roll = 5;

S1 show();

}

for(int i=0; i<50)

S2[1] input();

Q= ~~WAP~~ create a class inventory equipment to record the no. of students in a computer lab.

→ #include <iostream.h>

#include <conio.h>

#include <iomanip.h> (for 10 space gap = setw)

int name;

float quan, cost;

Class Lab

1)
```
// Area calculation
#include <stdio.h>
#include <conio.h>
void main()
{
int x,y,z;
float A;
clrscr();
float area(int);
int area(int,int);
int area(int, int, int);
printf("Enter the radius of circle");
scanf("%d", &x);
  A= area(x)
printf("Area of circle %d", A);
printf("Enter length and breadth of
                    rectangle ");
scanf("%d %d", &x, &y);
  A= Area(x,y);
printf("Area of rectangle =%d", A);
printf("Enter the length, breadth
and height of cuboid");
scanf("%d %d %d", &x, &y, &z);
A= Area(x,y,z);
printf("Area of cuboid = %d", A);
getch();
}
float area(int a)
```

```
{
return 3.14 * a * a;
}
int area (int a, int b)
{
return a * b;
}
int area (int a, int b, int c)
{
return 2 * (a*b + b*c + c*a);
}
```

2)

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
int P,R,T,n;
float A;
float interest (int, int, int);
float interest (int, int, int, int);
printf ("Enter P,R, T and n");
scanf ("%d %d %d %d", &P, &R, &T, &n);
A= Interest (P,R,T);
printf ("The Simple Interest is %f", A);
getch ();
float interest (int P, int R, int T);
{
```

```
return (P*R*T/100);
}
float interest (int P, int R, int n, int T);
{
return (P* Pow[(1+r/n), n*T)]);
}
```

Q= Rewrite the program in C++

→

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
void main()
{
float interest (int, int, float);
float interest (int, int, double);
int P,t;
float I, r;
clrscr();
cout << "Enter value of R,P,t";
cin >> P>> t >> r;
I= Interest (P,t, (double) r);
cout << " Compound Interest = "<<I;
getch();
}
float interest (int P, int t, float r)
{
return (P* t * r)/100;
```

```
}
float interest(int P, int t, double r, int n)
{
return P* Pow[(1+r/n), t];
}
```

## Compound Interest Formula

$$a = p * (1 + r/n)^{n*t}$$

$p \rightarrow$ principle amount
$r \rightarrow$ rate of interest in decimal
$n \rightarrow$ number of years
$t \rightarrow$ no. of times interest is calculated in a year.
$r \rightarrow$ should be passed in decimal like for 20% = 0.2
or
it should be written as = $\frac{20}{100}$

Q:- Define the structure student with useful data. Write two functions input() and show() to get the input and print the value of structure elements.

→
```
#include <iostream.h>
#include <conio.h>
struct stud { char name[20];
              float marks;
              int Roll no;
              } S1;
void input();
{
 cout << "Enter the name of student";
cin.getline(S1.name, 50);
cout << "Enter marks";
cin >> S1.marks;
cout << "Enter Roll no.";
cin >> S1.Roll no.;
}
Void show()
{
cout << "Roll no = " << S1.Roll No;
cout << "Name = " << S1.Name;
cout << "Marks = " << S1.Marks;
  }
void main()
```

```
{
clrscr();
S1.input();
S1.show();
getch();
}
```

17th Aug-'16

## Constructors

→ Special func" within a class
→ Have the same name as class name.
→ No return value, no return type
→ Used to initialize the data members
→ Executed automatically when object of a class is created.
→ Can be overloaded.

## Destructors

→ Special function within class
→ Have same name as class name preceeded by ~ (tilda)
→ No return value, no return type
→ Not used to initialize
→ Executed when class object is destroyed.
→ Cannot be overloaded.

Example-

```
Class SI
{
  int p;
 int r;
 int t;
public:
 SI ()
 {
  p=100,  r=10, t=1;
 }
~SI()
 {
 cout << p * r * t /100;
 }
void main ()
 {
  SI. simple ;
 }
```

```
/* Friend Function */

class n1;
class n2
{
int x,y;
Public:
n1()
n1(int a, int b)
void input();
void show();
friend float mean(n1, n2);
};
class n2
{
int x,y;
public:
n1()
n2(int a, int b)
void input();
void show()
friend float mean(n1, n2)
};
float mean(n1 a, n2 b)
{float t
t= a.x + a.y + b.x + b.y;
return (t/4);
}
```

```
void main()
{
n1 01(10,20);
n2 02(25.5, 30);
cout << "Mean =" << Mean(01, 02);
}
```

Q= Create two classes DMC and DFI to maintain the distance in ~~XX~~ USN English system. DMC stores distance in metre and centimetres and DFI stores distance in feet and inches. Find the sum of distances of ~~two~~ the objects of DMC and DFI class using friend function. Can we overload the plus operator to do the same functionality

{

→

```
class DMC();
Class DFI
{
  int f, i;
  public:
  DFI()
  { f=0; i=0; }
  DFI (int a, int b)
  {
    f=a;
    i=b;
  }
```

29/8/16

```
Class student
    {
    int rollno;
    ==
    ==
    public:
    student() { rollno=20; }
    student (int a)
    {
     rollno=a; }
    student (int a=6)
    {
    rollno. =a;
    }
    student (student a)
    {
     a = a. rollno;
    }
```

# Operator overloading

Syntax –

```
return type operator + ()
{
    ======
    ======
}
```

eg. –

```
class num
{
int x, y;
public:
num () {----}
num (int a, int b) {-----}
void input () {------}
void show () {------}
num operator + (num a)
{
 int i, j;
 i = x + a.x;
 j = y + a.y;
return num (i, j);
}
void main()
{
```

```
num A, B(6), C;
A.input();
C= A+B;
C.show();
}
```

Q= Create a class time to maintain time in hours, minutes and seconds. Overload + operator to add objects of classes. Write main function to show useability of a class.

→

```
time operator + (time a)
{
  float as = s+a.s;
  if(s>60)
  {
    as -= 60;
    am++;
  }
int am = m+a.m;
  if(m>60)
  {
    am -= 60;
    ah++;
  }
int ah = h+a.h;
return time (ah, am, as)
}
```

Q= Create a class time to maintain time in hours, minutes and seconds. Overload operator to add objects of classes.

→ time operator — (time $t_1$, $t_2$)

```
{
  int y, z;
  float x;
  x = s - a.s;
  if (s > 60)
  {
    s - = 60;
    x++;
  }
m = m + a.m;
if (m > 60)
{
  m - = 60;
  y++;
}
h = h + a.h;
return (x, y, z)
```

Q: Using the class time write a program to input the start time and finish time of an event and find the total time taken in the event. Ensure that the input finish time should be greater than the start time.

→
```
class time
{
float t1, t2;
public:
void start time()
{
cout << " Enter the finish time";
```

```
enum bool { False, True };
class time
{
int h, m; float s;
public:
 time() { h = m = s = 0; }
 time ( int a, int b, float c)
 {
 h = a ; m = b ; s = c ; }
void input()
{
 cout << "\n Input h, m and seconds";
```

```
cin>> h >> m >> s;
}
void show()
{
cout << "H = " << h << "M = " << m << "S = "
                                    << s;
}
time operator - (time x)
{
 time temp;
 int ah = h;
 int am = m;
 float as = s;
 if (as < x.s)
 {
  as = as + 60;
  am = am - 1;
 }
 temp.s = as - x.s;
 if (am < x.m)
 {
  am = am + 60;
  ah --;
 }
 temp.m = am - x.m;
 temp.h = ah - x.h;
 return (temp);
}
```

```
bool operator > (time t)
{
   float s2, s1;
s2 = ft.h * 3600 + ft.m * 60 + ft.s;
s1 =  h * 3600 + m * 60 + s;
if (s2 > s1)
{
   f1.input();
   return (true);
}
 else
 {
  return (false);
 }
};
void main()
{
 time t, s, f;
 s.input();
 f.input();
 while (s>f)
{
cout << Finish time
 f.input();
}
 t = f - s;
 t.show()
}
```

Q:- Create a class DB to maintain the english distance in feet and inches. Overload operator to print the difference between two distances.