Member ( ) defining outside the class :-

FUNCTIONS can be written inside [or] outside the

→ Using namespace std;

```
#include <iostream>
class complex
{
    private : int a, b;
    public : void get ( );
             void add (complex, complex);
             void display ( );
};
void complex :: get ( )
  {
    cout << "enter a, b"; cin >> a >> b;
  }
void complex :: add (complex p, complex q)
{
    x = p.x; x + q.
    a = p.a + q.a ; b = p.b + q.b;
}
void complex :: display ( )
{
    cout << a << "+i" << b;
}
main ( )
{
    complex p, q, r;
    p.get ( ); q.get ( );
    r.add (p, q);
    r.display ( );
}
```

o/p :-

enter a, b: 1 2
enter a, b: 3 4
4 + i6

1 2
3 4
4+i6

higher order call back

complex complex
1+i2
5+i6
Shortcut

CONSTRUCTOR :- Object creates. It constructs an object. No return type.

CONSTRUCTOR :- It constructs an object & initialises values. It should have class name.
It shouldnot have return type atleast void.
It is called Implicit as well as Explicit.

Ex:-
class complex
{
private: int
It we donot write any construct compiler substitutes a default constructor.
we may have any number of constructors within a class.
i.e a constructors can be overloaded.

TYPES OF CONSTRUCTOR =>

1. Default constructor :-

SYNTAX :-

```
classname ( )
{
}
```

2. Parameterized constructor :-

SYNTAX :-

```
classname (type arg1 ---)
{
  body
}
```

Yes DESTRUCTOR:-

It destructs an object. It should have class name. It should have tilda sign '~' before it.

The order of the destructor execution is reverse order of the constructor execution.

We have only 1 destructor for an entire class i.e a destructor can't be overloaded.

Ex:-

```
-> using namespace std;
#include <iostream>
class complex
{
  private: int a, b;
  public: complex()
    {
      a=1, b=2;
    }
    complex(int x, int y)
    {
      a=x; b=y;
    }
    void display()
    {
      cout << a << "+i" << b << "\n";
    }
    ~complex()
    {
      cout << "destructor called \n";
    }
};
```

```
main ()
{
    complex p, q(5, 6);
    p.display ();
    q.display ();
}
```

O/P :- 1 + i 2
5 + i 6
Destructor called
Destructor called

25-06-2024

Harshad No:-

Ex:- 24

2+4 = 6

24/6 = 4 → Divisible of the sum is Harshad No

```
-> main()
{
   int n, sum=0, temp=num
   printf
   cout << "Enter n: ";
   cin >> n;
   for (i=0; i<n; i++)
```

24 => 2 | 4

2+4 %.10 /10

```
int num;
Enter number:-
("%d", &num
```

NUMBERS:-
18; 54; 120;
1729
666

```
-> #include <stdio.h>
int checkHarshad(int num)
{
   int sum=0;
   int temp = num;
   while (temp != 0)
   {
      sum = sum + temp %.10;
      temp /= 10;
   }
   return num %. sum == 0;
}
int main()
{
   int num;
   printf("Enter number: ");
   scanf("%d", &num);
   if (checkHarshad(num))
      printf("%d is Harshad's No: ", num);
   else printf("%d is not Harshad's No: ", num);
}
```

METHOD-2 FOR HARSHAD NO:-

EX:- 24

2+4=6

24/6=4 -> Divisible of the sum is Harshad No

```
-> #include <stdio.h>
   int checkHarshad (int num)
   {
      int sum=0;
      int temp = num;
      while (temp != 0)
      {
        sum = sum + temp %10;
          temp /= 10;
      }
      return num %. sum == 0;
   }
   int main()
   {
      int num;
      char choice;
      do {
            printf ("Enter number: ");
            scanf ("%d", &num);
```

```
if (checkHarshad(num))
    printf("%d is a Harshad's No.\n", num);
else
    printf("%d is not a Harshad's No\n", num);

// Clear the input buffer
fflush(stdin);

printf("\n\nDo you want to check another
        number? (y/n): ");
scanf("%c", &choice);
}
while (choice == 'y' || choice == 'Y');
return 0;
printf("\n\nThank You for running the code
        :)\n");
printf("
}
```