**Static**

**What:**

Static is a keyword in C++ used to give special characteristics to an element. Static elements are allocated storage only once in a program lifetime in static storage area. And they have a scope till the program lifetime.

**Where:**

Static Keyword can be used with following,

1. Static variable in functions
2. Static Class Objects
3. Static member Variable in class
4. Static Methods in class

**How:**

Static variable in functions

These variable are initialized only once, they are stored in intialized data segment if not initialized they are stored in the uninitialized data segment.

**#01\_Pgm**

**void counter()**

{

static int count=0;

cout << count++;

}

int main(0

{

for(int i=0;i<5;i++)

{

counter();

}

}

**Output:**

0 1 2 3 4

**#02\_Pgm**

**void counter()**

{

int count=0;

cout << count++;

}

int main(0

{

for(int i=0;i<5;i++)

{

counter();

}

}

**Output:**

0 0 0 0 0

Note:

Reinitialization of the variable is been avoided in the static variable.

Static Class Objects

**#03\_Pgm**

**class Abc**

{

int i;

public:

Abc()

{

i=0;

cout << "constructor";

}

~Abc()

{

cout << "destructor";

}

};

void f()

{

static Abc obj;

}

int main()

{

int x=0;

if(x==0)

{

f();

}

cout << "END";

}

Output;

constructor END destructor

Note:

static lifetime is till the end of the program. Hence the destructor is called at the end of the program.

**#04\_Pgm**

class Abc

{

int i;

public:

Abc()

{

i=0;

cout << "constructor";

}

~Abc()

{

cout << "destructor";

}

};

void f()

{

Abc obj;

}

int main()

{

int x=0;

if(x==0)

{

f();

}

cout << "END";

}

Output:

constructor destructor END

Note:

Here the object is not static hence they get removed from the stack once they move out of the scope.

Static member Variable in class   
 Static data members of class are those members which are shared by all the objects. Static data member has a single piece of storage, and is not available as separate copy with each object, like other non-static data members.

**#05\_Pgm**

class X

{

public:

static int i;

X()

{

// construtor

};

};

int X::i=1;

int main()

{

X obj;

cout << obj.i; // prints value of i

}

Output:

1

Static member function:

These functions work for the class as whole rather than for a particular object of a class.

It can be called using an object and the direct member access . operator. But, its more typical to call a static member function by itself, using class name and scope resolution :: operator.

### #06\_Pgm

class X

{

public:

static void f()

{

// statement

cout << “Hello\n”;

}

};

int main()

{

X::f(); // calling member function directly with class name

}

Output:

Hello

Static functions are used to call the function directly via the class name, they cannot be used to modify the data member or call other member functions in the class.