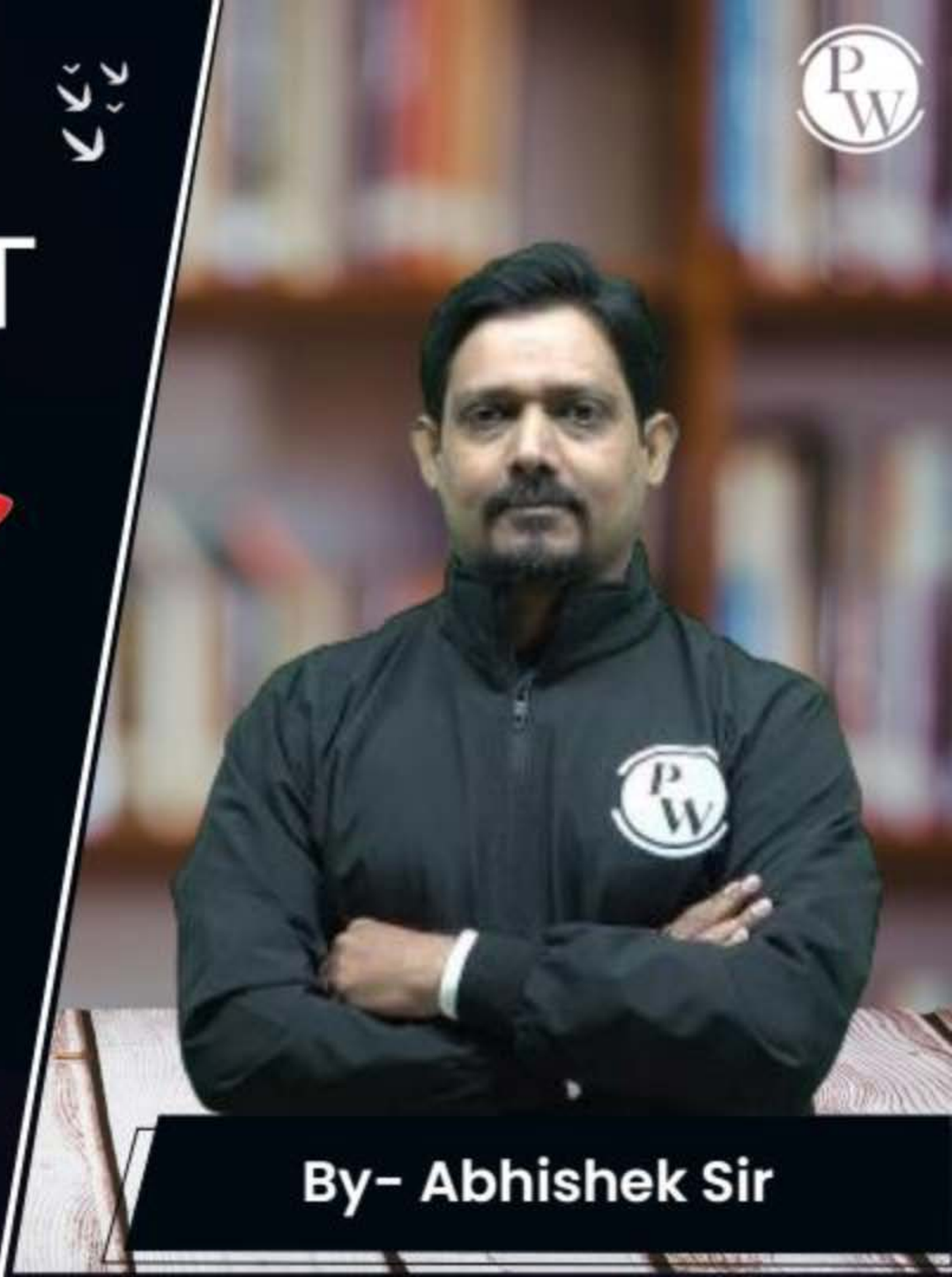
Computer Science & IT

C Programming

Function & Storage Class

Lecture No. 02



By- Abhishek Sir

Recap of Previous Lecture

Topic — function

Topic — Activation Record

Topic — Run time stack

Topic — Memory Layout

Topic

Slide

Topics to be Covered

- Topic — Storage class
- Topic — Auto
- Topic — Static
- Topic — Register
- Topic — Extern

Slide

Storage Class

Storage class (Memory)

defines. Memory where variable will be allocated

initial value of variable, Scope of variable, and Lifetime of a variable.

Scope : the part of program where variable can be Accessed.

Lifetime: How Long variable will be allocated.

Slide

Storage Class

Storage Specifier	Memory	Initial value	Scope	Lifetime
auto	Stack	Garbage	Within block	End of block
Extern	Data segment	Zero	Global / multiple files	Till the program
Static	Data Segment	Zero	Global – within program Local- within block	Till the program
Register	CPU register	Garbage	Within block	End of Block

Storage Class

Auto default storage class for Local variable

```
{
  int a;
}
```

```
{
auto int a,
}
```

Memory : stack

Initial value :- garbage

Scope : within block

Lifetime : until block is in execution

Slide

Storage Class

Static is default storage class for global variable

```
int a;
int main(){

}
```

```
Static int a;
int main(){

}
```

Memory : Data Segment

Initial value : 0

Scope : Entire program

Life time :- Until program is Running

Slide

Storage Class

Local variable can be declared as static

but global can't declared as auto.

```
auto int a,
int main(){
}
```

```
void fun(){
  static int a,
}
int main(){
}
```

if Local declared as static

Memory : Data Segment

Initial value : 0

Scope : within the block

Life time :- until program is Running

Slide

Storage Class

```
#include<stdio.h>
void fun( int a) {
    static int i;
    i++;
    printf("%d", a+i);
}
```

Not allowed

```
int main(){
fun(10);   // 11
fun(20);   // 22
fun(30);   // 33
return 0;
}
```

i 0 1 2 3

Static variable remember old value during multiple call for the function

Slide

Storage Class

4 5 6 7

```
#include<stdio.h>
void fun( int a) {
    static int i=4;
    i++;
    printf("%d", a+i);
}
```

Not allowed

```
int main(){
fun(10); //15
fun(20); //26
fun(30); //37
return 0;
}
```

Static variable initialized only once

Assignment can be done multiple times

Slide

Question

#Q

The value of j at the end of the execution of the following C program

```
int incr (int i)
{
        static int count = 0;
        count = count + i;
        return (count);
}
main () {
        int i,j;
        for (i = 0; i <=4; i++)
          j = incr(i);
}
```

is

(a) 10

(b) 4

(c) 6

(d) 7

count 0 ~~1~~ 3 6 10

j = incr(i)

i = 0 j = 0

i = 1 j = 1

i = 2 j = 3

i = 3 j = 6

i = 4 j = 10

Slide

GATE ~~2004~~ 2021

#Q
```
#include<stdio.h>
int funcp(){
static int x = 1;
x++;
return x;
}
int main(){
int x,y;
x = funcp();
y = funcp()+x;
printf("%d\n", (x+y));
return 0;
}
```

The value printed by program ✗ ~~1~~ ~~2~~ 3

x = funcp()

x = 2

y = funcp() + x ; = 3 + 2 = 5

Slide

Question

Consider the following C program:

```
#include <stdio.h>
int foo(int x, int y){
    static int a;
    x=2*x+y;
    y =2*y+x;
    a = x+y;
    return a;
}

int main(){
int x=2, y=5;
y= foo(y,x);
x= foo(y,x);
printf("%d \n", x);
return 0;
}
```

The value printed by the program is 120

Slide

Question

```
#Q # include <stdio.h>
void print1 (void){
    static int x = 10;
    x+= 5;
    printf ("%d\t", x);
}
void print2 (void){
    static int x;
    x= 10;
    x+= 5;
    printf ("%d\t", x);
}
int main (){
print1(); print1(); print2(); print2();
return 0;
}
```

15 20 , 15, 15

(A) 15 20 25 30

(B) 15 20 15 20

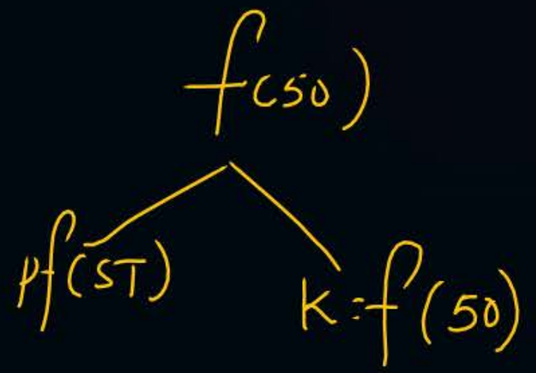
(C) 15 15 15 15

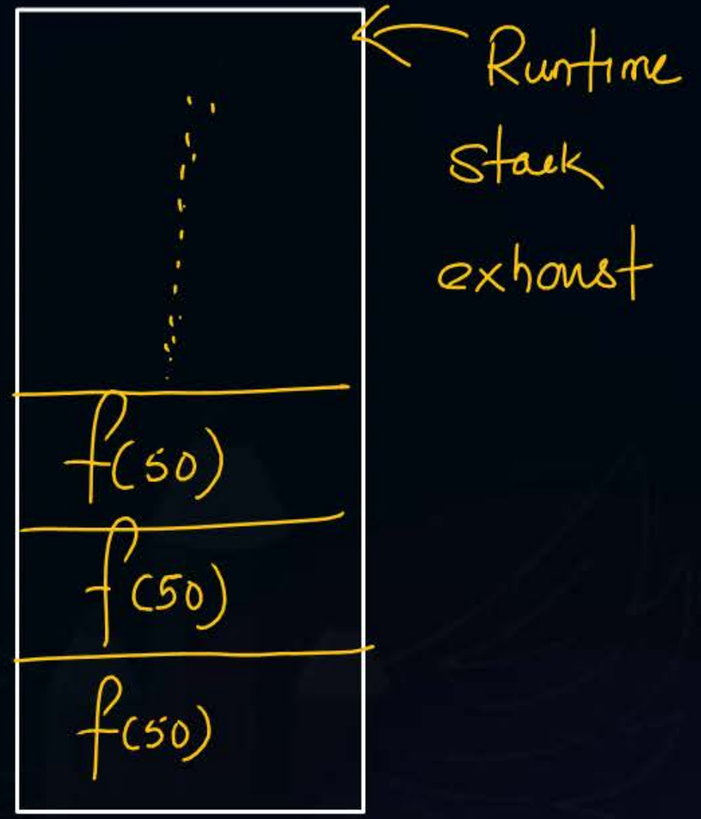
(D) 15 20 15 15

Slide

#Q Consider the C function given below

```
int f(int j){
static int i = 50;
int k;
if (i == j){
        printf("something");
        k = f(i);
        return 0;
}
else
        return 0;
}
```





Which one of the following is TRUE?

(A) The function returns 0 for all values of j. ✓

(B) The function prints the string something for all values of j. ✗ (i != j)

(C) The function returns 0 when j = 50.

(D) The function will exhaust the runtime stack or run into an infinite loop when j = 50. ✓

lide

Question

Consider the following program

```
#include<stdio.h>
int a,b,c;
int print(void){
    static int a=10;
    return a++;
}
int main() {
    int a=0,b,c;
    a+=print();
    a++;
    print();
    a+=print();
    printf("%d",a);
}
```

Output of the program is

a ~~10~~ ~~11~~ 12

a = a + print(),
= 0 + 10 = 10

11

a = a + print
11 + 12 = 23

(A) 20

(B) 21

(C) 22

(D) 23

Slide

Question

#Q The integer value printed by the ANSI-C program given below is .

```
#include<stdio.h>
int bar(int y){
    static int x = 20;
    x-=y;
    return x;
}
int foo(){
static int x = 1;
x+=10;
return bar(x);
}
int main(){
int x,y;
x = foo();
y = bar(x)+x;
printf("%d\n", (x+y));
return 0;
}
```

Slide

GATE 2004

#Q

```
#include <stdio.h>
int foo(int x);
int bar(int y);
int main ()      {
        int x = 1, y = 2, count;
        for (count = 1; count <=2; ++count){
                y+=foo(x) + bar(x);
        printf("\n %d",y);
        }
}
```

```
int foo (int x) {
        int y;
        y=bar(x);
        return(y);
}
int bar (int x) {
        static int y = 10;
        y+=1;
        return (y+x);
}
```

Count = 1

y = 2 + 12 + 13 = 27

y = bar(1)

y = 12

y | 10 | 11 | 12 | 13

27 + 14 + 15 = 27 + 29 = 56

(A) 27 56 (B) 42 74 (C) 33 37 (D) 32 32

Slide

GATE 2017

#Q The output of executing the following C program is _____.

```
# include <stdio.h>
int  total (int v) {
while (v)    {
      count + = v & 1;
      v>> = 1;
}
      return count;
}
void main ( ) {
static int x = 0;
int  i = 5;
for (; i> 0; i--)  {
      x=x + total (i) ;
}
printf ("%d\n", x)  ;
}
```

static int count = 0

(A) 20

(B) 21

(C) 23

(D) None of the above

Slide

GATE 2017

+=

count = 2 3 5 6 7

#Q The output of executing the following C program is _____.

```
# include <stdio.h>
int  total (int v) {
while (v)    {
      count + = v & 1;
      v>> = 1;
}
      return count;
}
void main ( ) {
static int x = 0;
int  i = 5;
for (; i> 0; i--)   {
      x=x + total (i) ;
}
printf ("%d\n", x)   ;
}
```

Counting No. of 1's in Binary Representation of v

static int count = 0

0101 ← MSB, LSB
& 0001
(1)

0010 LSB
0001
0000

0001
0001
0001

x = x + total(i)

i = 5 : 0 + total(5) = x = 2
i = 4 : 2 + total(4) = x = 5 (total(4) = 3)
i = 3 : 5 + total(3) = x = 10 (total(3) = 5)
i = 2 : 10 + 6 = x = 16
i = 1 : 16 + 7 = x = 23

0100	0010	0001
0001	0001	0001
0000	0000	0001

Slide

2 mins Summary

Topic storage class

Topic auto

Topic Static

Topic Local static

Topic

Slide

THANK - YOU