Computer Science & IT

C Programming

Practice Classes

Lecture No. 02

By- Abhishek Sir

Recap of Previous Lecture

- Topic — practice problem
- Topic
- Topic
- Topic
- Topic

Slide

Topics to be Covered

- Topic
- Topic
- Topic
- Topic
- Topic

practice problem

Slide

Question

Consider the following C program:

```
#include <stdio.h>
int jumble(int x, int y){
    x=2*x+y;
    return x;
}
int main(){
    int x=2, y=5;
    y= jumble(y,x);
    x= jumble(y,x);
    printf("%d \n", x);
    return 0;
}
```

1 x=5, y=2 – first call

x = 2*5+2 = x=12

2. x=12, x=2

x = 2*12+2 = 24+2 = 26

x	~~2~~ 26	y	~~5~~ 12

y = jumble(y, x) = 12

x = jumble(y, x) = 26

final value of x is 26

The value printed by the program is 26

Slide

Question

Consider the C program shown below.

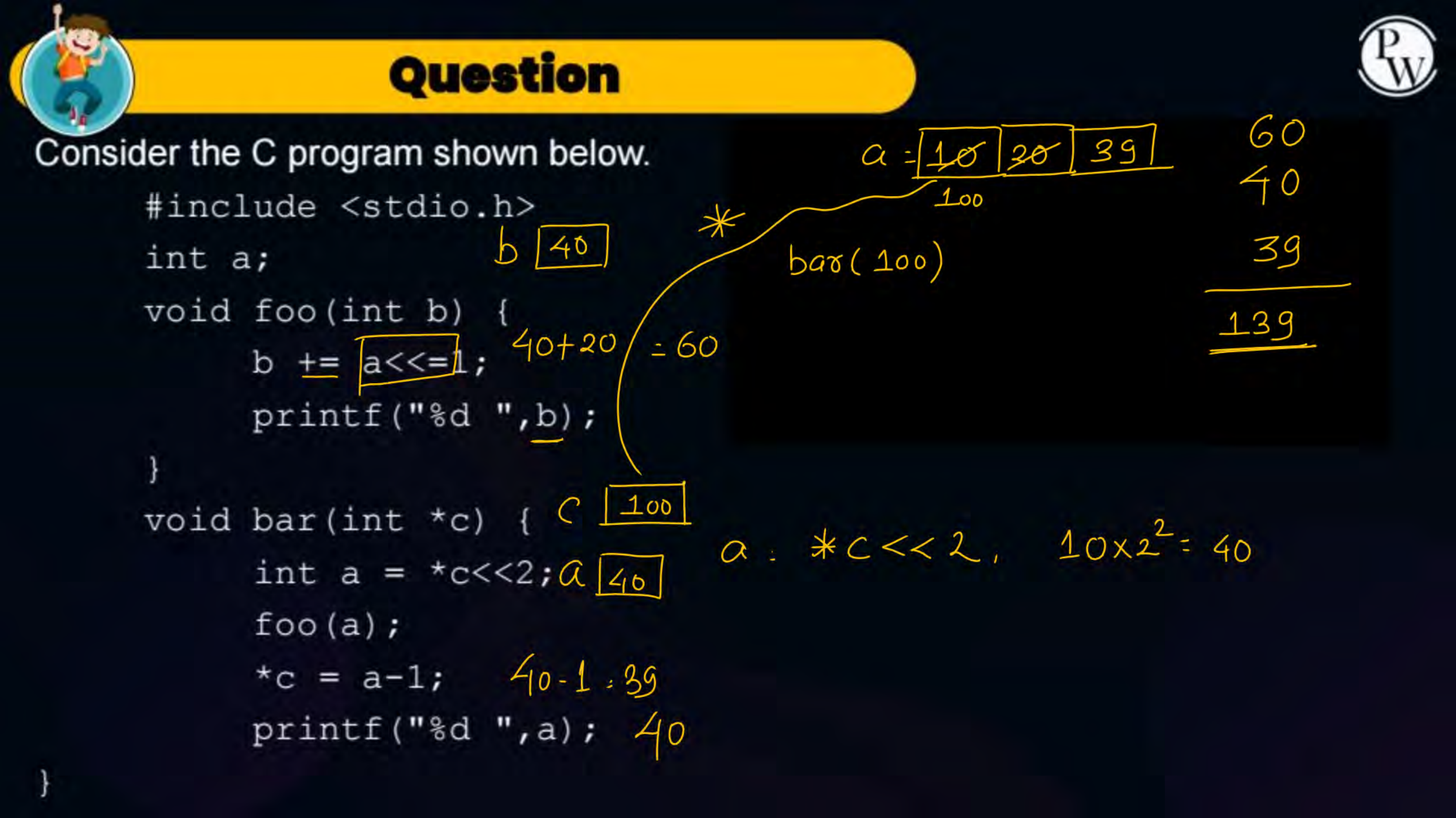
```
#include <stdio.h>
int a;
void foo(int b) {
        b += a<<=1;
        printf("%d ",b);
}
void bar(int *c) {
        int a = *c<<2;
        foo(a);
        *c = a-1;
        printf("%d ",a);
}
void main(void) {
        a = 10;
        bar(&a);
        printf("%d ",a);
}
```

The Sum of the value printed by the program is _____

Question

Consider the C program shown below.

```
#include <stdio.h>
int a;
void foo(int b) {
      b += a<<=1;
      printf("%d ",b);
}
void bar(int *c) {
      int a = *c<<2;
      foo(a);
      *c = a-1;
      printf("%d ",a);
}
```



Question

Consider the C program shown below.

```
#include <stdio.h>
int a;
void foo(int b) {
        b += a<<=1;
        printf("%d ",b);
}
void bar(int *c) {
        int a = *c<<2;
        foo(a);
        *c = a-1;
        printf("%d ",a);
}
void main(void) {
        a = 10;
        bar(&a);
        printf("%d ",a);
}
```

The Sum of the value printed by the program is __139____

Question

2017

Consider the C functions foo and bar given below:

```
int foo (int val ) {
  int x = 0;
  while (val > 0) {
        x = x + foo ( val --);
  }
  return val ;
}
```

```
int bar (int val ) {
  int x = 0;
  while (val > 0) {
    x = x + bar (val - 1) ;
  }
  return val ;
}
```

Abnormal termination

Invocations of foo (3) and bar (3) will result in:

(A) Return of 6 and 6 respectively.

(B) Infinite loop and abnormal termination respectively.

(C) Abnormal termination and infinite loop respectively.

(D) Both terminating abnormally

Slide

Question

Consider the C functions foo and bar given below:

```
int foo (int val ) {
  int x = 0;
  while (val > 0) {
        x = x + foo ( val --);
  }
  return val ;
}
```

post decrement

foo(3)

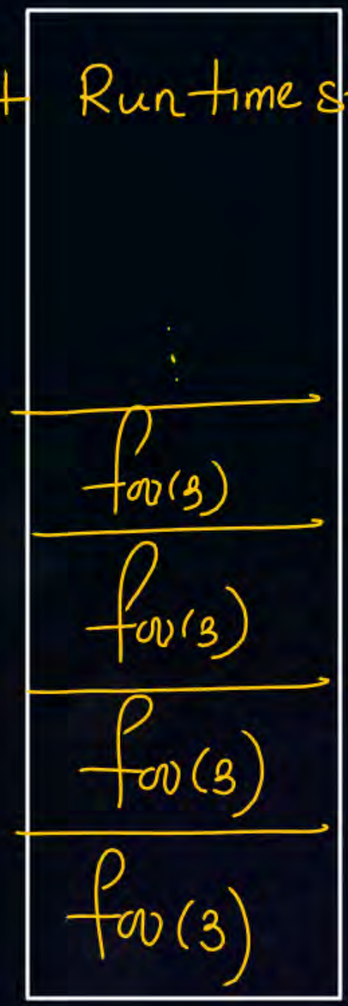
3>0

x = 0 + foo(3)

3>0

x = x + foo(3)

exaust Run time stack



Slide

Question

```
int bar (int val ) {
  int x = 0;
  while (val > 0) {
    x = x + bar (val - 1) ;
  }
  return val ;
}
```

Infinite loop program

while (3>0)

x = x + bar(2)

while(2>0)

x = x + bar(1)

while(1>0) {

val is update??

1>0

x = 0

x = x + bar(0) 0

bar(0) 0>0

}

Slide

#Q. Consider the following C program.

```
# include <stdio.h>
int main (){
int a[4][5] = {{1, 2, 3, 4, 5},
               {6, 7, 8, 9, 10},
               {11, 12, 13, 14, 15},
               {16, 17, 18, 19, 20}};
printf("%d\n", *(*(a+**a+ 1)+4)+ *(*(a+**a)+4));
return (0);
}
```

A 15

B 10

C 25

D 20

15 + 10 ✓ = 25

a[2][4] + a[1][4]

*xa : a[0][0]

((a+2)+4)

((a+1)+4)

: a[1][4]

#Q. Consider the following C program.

```
# include <stdio.h>
int main (){
int a[4][5] = {{1, 2, 3, 4, 5},
               {6, 7, 8, 9, 10},
               {11, 12, 13, 14, 15},
               {16, 17, 18, 19, 20}};

printf("%d\n", *(*(a+**a+ 1)+4)+ *(*(a+**a)+4));
return (0);
}
```

A 15

B 10

C **25**

D 20

#Q. What is the output printed by the following program?

```
# include <stdio.h>
int f(int n, int k) {
    if (n == 0) return 0;
    else
        if (n%2)
            return f(n/2, 2*k) + k;
    else
            return f(n/2, 2*k) - k;
}
int main () {
        printf("%d",f(34,1));
        return 0;
    }
```

(A)5 (B) 8 (C) 9 (D) 20

#Q. What is the output printed by the following program?

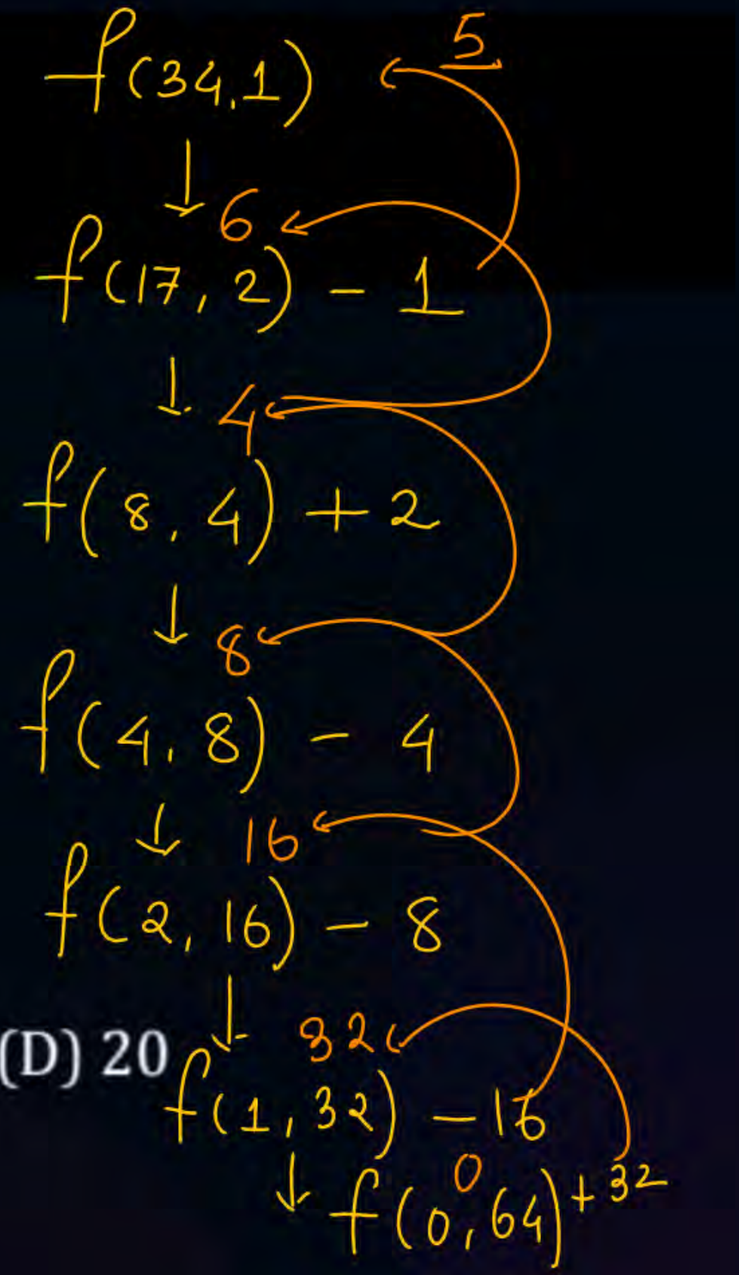
```
# include <stdio.h>
int f(int n, int k) {
    if (n == 0) return 0;
    else
            if (n%2)
                    return f(n/2, 2*k) + k;
    else
                    return f(n/2, 2*k) - k;
}
```

(A) 5 (B) 8 (C) 9 (D) 20

odd

odd

even



#Q. Output of the program is

```
# include <stdio.h>
int tmp = 20;
int func() {
    static int tmp = 10;
    return tmp++*3;
}
int main () {
    printf("%d", func ()+func ()+tmp);
}
```

A 82

B 83

C 79

D 56

#Q. Output of the program is

```
# include <stdio.h>
int tmp = 20;
int func() {
    static int tmp = 10;
    return tmp++*3;
}
int main () {
    printf("%d", func ()+func ()+tmp);
}
```

temp: 10 11 12

10 * 3

30 + 33 + 20 = 83

#Q. Output of the program is

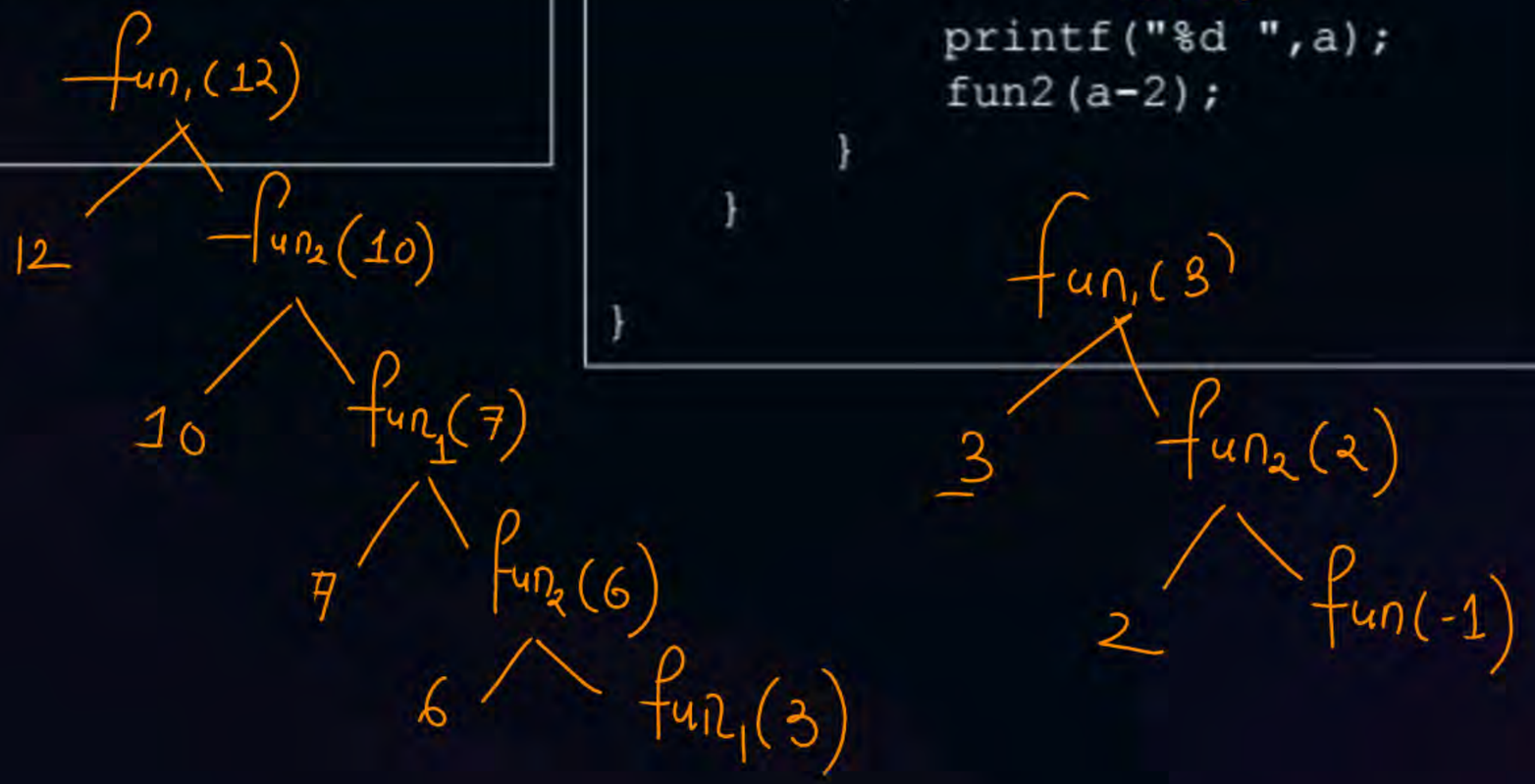
```c
#include
<stdio.h>
void fun1(int);
void fun2(int);

int main(){
    int v=12;
    fun1(12);

    return 0;
}
```

```c
void fun2(int a){
    if (a>0){
        printf("%d ",a);
        fun1(a-3);
    }
}
```

```c
void fun1(int a){
    if(a>0){
        if (a%2){   odd
            printf("%d ",a);
            fun2(a-1);
        }
        else
        {          even
            printf("%d ",a);
            fun2(a-2);
        }
    }
}
```



(A) 12 10 8 6 4 2

(B) 12 10 7 6 3 2

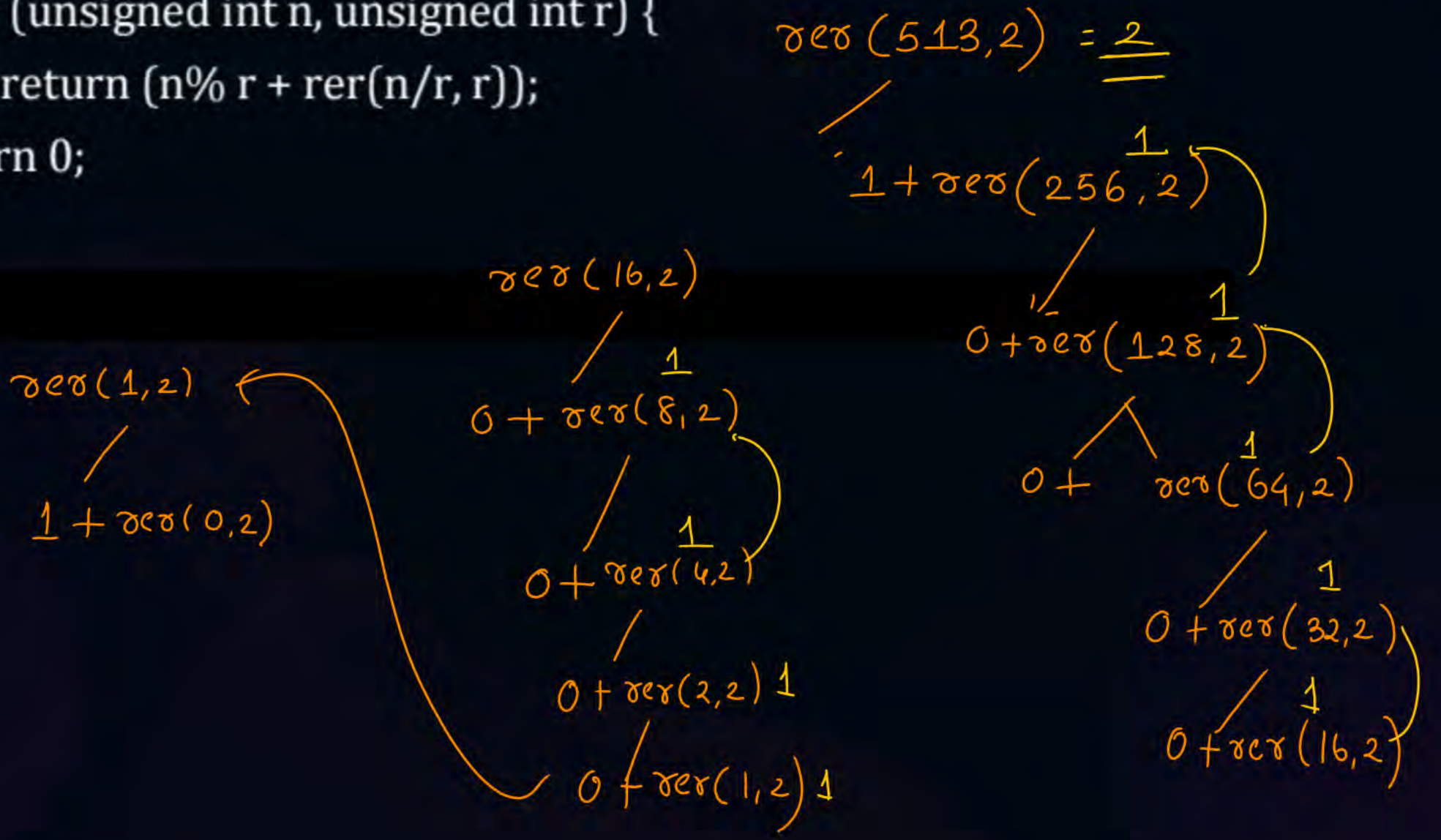
(C) 12 10 7 6 4 2

(D) 12 10 8 6 3 2

#Q Consider the following recursive C function that takes two arguments

```
unsigned int rer (unsigned int n, unsigned int r) {
        if (n > 0) return (n% r + rer(n/r, r));
        else return 0;
}
```

(a) 9
(b) 8
(c) 5
(d) 2



#Q Consider the following recursive C function that takes two arguments

```
unsigned int rer (unsigned int n, unsigned int r) {
        if (n > 0) return (n% r + rer(n/r, r));
        else return 0;
}
```

What is the return value of the function rer when it is called as rer (513, 2)?

(a) 9

(b) 8

(c) 5

(d) **2**

#Q Consider the following program is pseudo-Pascal syntax.

```
program main;
var x: integer;
procedure Q [z:integer);
begin
    z: z + x;
    writeln(z)
end;
procedure P (y:integer);
var x: integer;
begin
    x: y + 2;
    Q(x);
    writeln(x)
    end;
begin
    x:=5;
    P(x);
    Q(x);
    writeln(x)
end.
```

(II) z = 5
z = 5+5 = 10

(I) z = 7

7+5 = 12 = 12 static

7+7 = 14 ← Dynamic

y = 5

x = 5+2 = 7

Q(7)

12, 7, 10, 5

← global

Dynamic scoping we search in funchon p

#Q Consider the following program is pseudo-Pascal syntax.

```
program main;
var x: integer;
procedure Q [z:integer);
begin
        z: z + x;
        writeln(z)
end;
procedure P (y:integer);
var x: integer;
begin
        x: y + 2;
        Q(x);
        writeln(x)
        end;
begin
        x:=5;
        P(x);
        Q(x);
        writeln(x)
end.
```

Output of the program if the parameter passing mechanism is call-by-value and the scope rule is static scooping?

(a) 12 7 10 5

(b) 14 14 10 10

(c) 12 7 12 5

(d) 10 10 14 14

Question

Consider the following ANSI C function

```
int SimpleFunction(int Y[], int n, int x) {
int total = Y[], loopIndex;
For (loopIndex = 1; loopIndex<=n-1; loopIndex++)
      total = x * total Y[loopIndex];
return total;
}
```

Let Z be an array of 10 elements with Z[i] = 1 for all I such that . The value returned by simpleFunction (Z, 10, 2) is _____.

Question

Consider the following C function.

```
int fun1(int n) {
    static int i = 0;
    if (n > 0) {
        + + i;
        fun1(n-1);
    }
    return (i);
}
```

```
int fun2(int n) {
    static int i = 0
    if (n > 0) {
        i = i + fun1 (n);
        fun 2(n -1);
    }
    return (i);
}
```

The return value of `fun2 (5)` is _____.

Slide

2 mins Summary

- Topic
- Topic
- Topic
- Topic
- Topic

practice problem

Slide

THANK - YOU