Ques : Make a function which calculates the factorial of n using recursion.

```
int factorial(int n){
    return n*factorial(n-1);
}
```
4 × factorial (3);

```
int main(){
    int n;
    printf("Enter a number : ");
    scanf("%d",&n); n = 4
    int fact = factorial(4);
    printf("%d", fact);
    return 0;
}
```

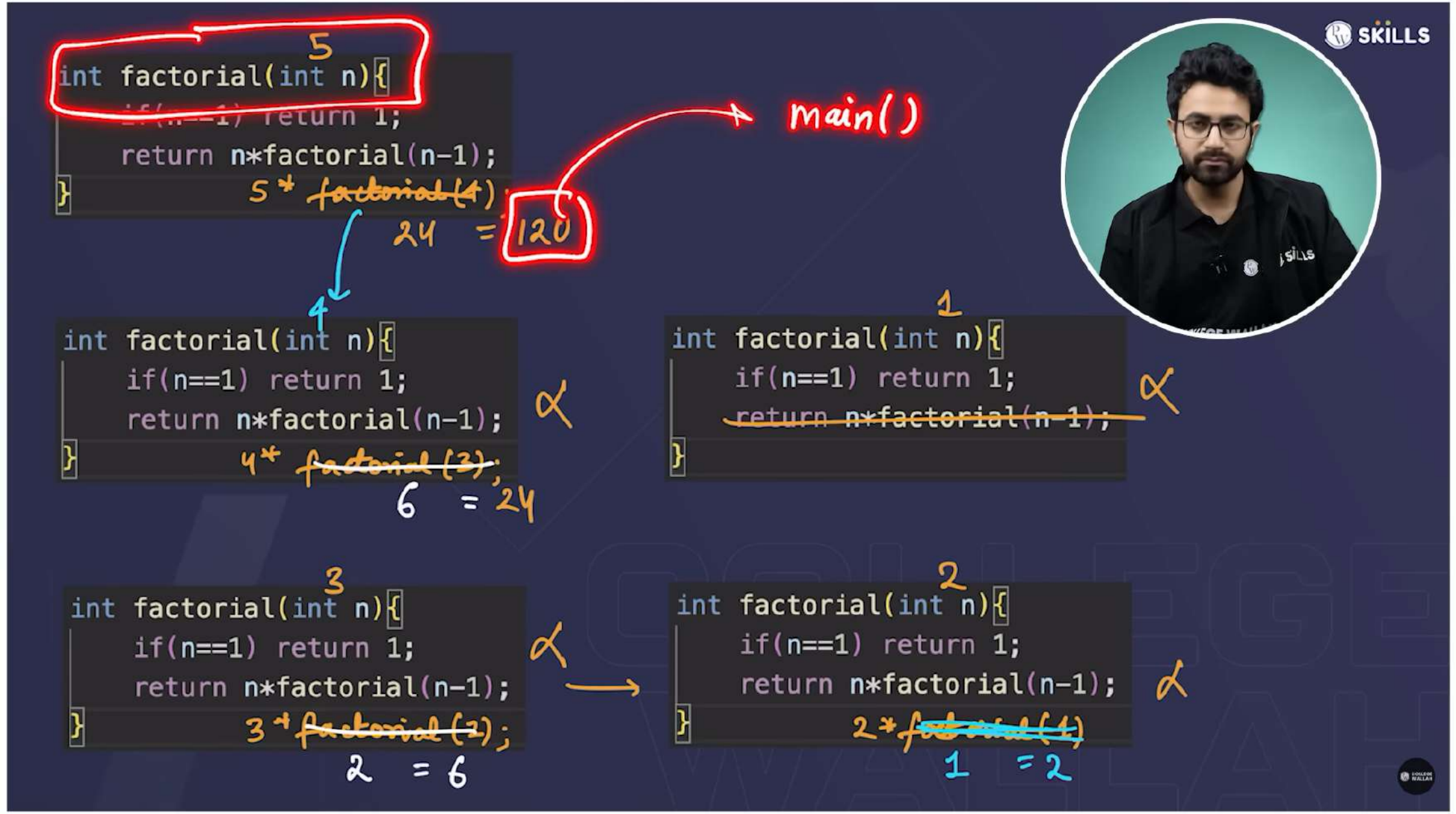
```
int factorial(int n){
    return n*factorial(n-1);
}
```
3 * factorial (2);

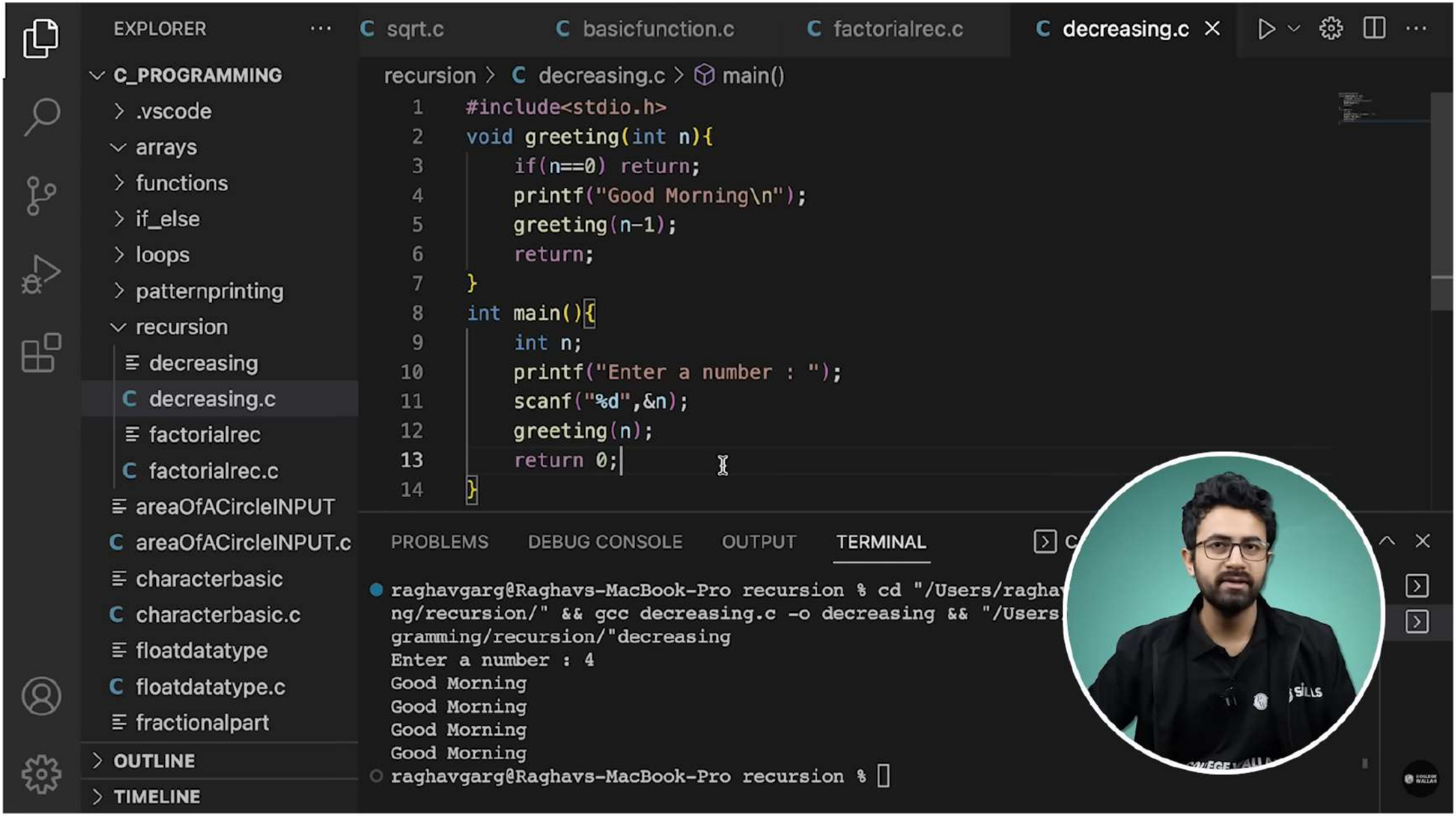
```
int factorial(int n){
    return n*factorial(n-1);
}
```
2 * factorial (1)

```
int factorial(int n){
    return n*factorial(n-1);
}
```
1 * factorial (0)

```
int factorial(int n){
    return n*fact
}
```
0 * factorial (-1)

fact(-1)
= -1 × fact(-2)


5
int factorial(int n){
if(n==1) return 1;
return n*factorial(n-1);
}
5 * factorial(4);
24 = 120
main()
4
int factorial(int n){
if(n==1) return 1;
return n*factorial(n-1);
}
4* factorial(3);
6 = 24
1
int factorial(int n){
if(n==1) return 1;
return n*factorial(n-1);
}
3
int factorial(int n){
if(n==1) return 1;
return n*factorial(n-1);
}
3 * factorial(2);
2 = 6
2
int factorial(int n){
if(n==1) return 1;
return n*factorial(n-1);
}
2 * factorial(1)
1 = 2
SKILLS


EXPLORER
C_PROGRAMMING
.vscode
arrays
functions
if_else
loops
patternprinting
recursion
decreasing
decreasing.c
factorialrec
factorialrec.c
areaOfACircleINPUT
areaOfACircleINPUT.c
characterbasic
characterbasic.c
floatdatatype
floatdatatype.c
fractionalpart
OUTLINE
TIMELINE
sqrt.c
basicfunction.c
factorialrec.c
decreasing.c
recursion > decreasing.c > main()
#include<stdio.h>
void greeting(int n){
if(n==0) return;
printf("Good Morning\n");
greeting(n-1);
return;
}
int main(){
int n;
printf("Enter a number : ");
scanf("%d",&n);
greeting(n);
return 0;
}
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc decreasing.c -o decreasing && "/Users
gramming/recursion/"decreasing
Enter a number : 4
Good Morning
Good Morning
Good Morning
Good Morning
raghavgarg@Raghavs-MacBook-Pro recursion %

```
void greeting(int n){
    if(n==0) return;
    printf("Good Morning\n");
    greeting(n-1);
    return;
}
```

```
void greeting(int n){
    if(n==0) return;
    printf("Good Morning\n");
    greeting(n-1);
    return;
}
```

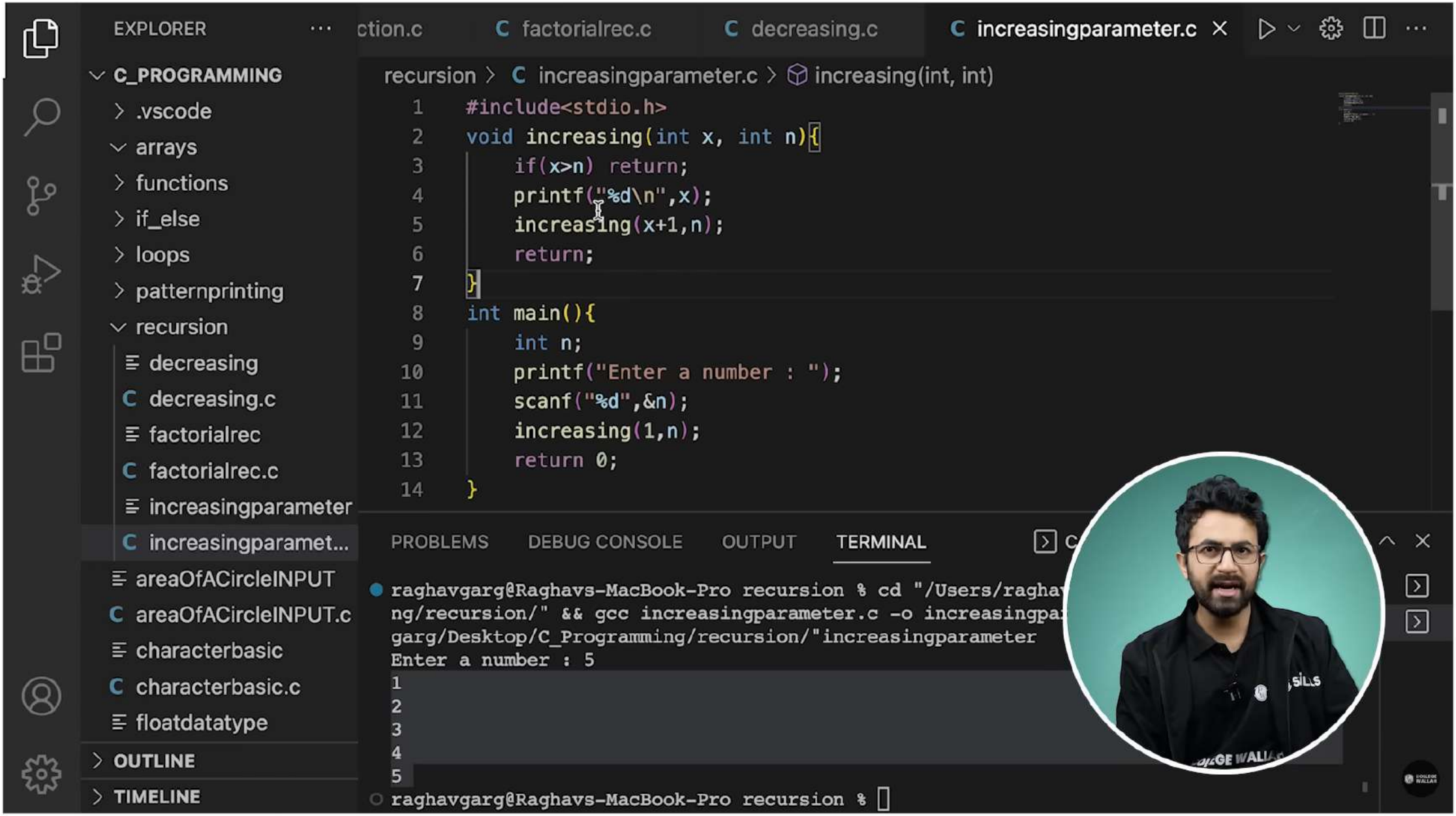
```
void greeting(int n){
    if(n==0) return;
    printf("Good Morning\n");
    greeting(n-1);
    return;
}
```

```
void greeting(int n){
    if(n==0) return;
    printf("Good Morning\n");
    greeting(n-1);
    return;
}
```

```
void greeting(int n){
    if(n==0) return;
    printf("Good Morning\n");
    greeting(n-1);
    return;
}
```

Output

- Good Morning
- Good Morning
- Good Morning
- Good Morning
- Good Morning


EXPLORER
C_PROGRAMMING
.vscode
arrays
functions
if_else
loops
patternprinting
recursion
decreasing
decreasing.c
factorialrec
factorialrec.c
increasingparameter
increasingparamet...
areaOfACircleINPUT
areaOfACircleINPUT.c
characterbasic
characterbasic.c
floatdatatype
OUTLINE
TIMELINE
ction.c
factorialrec.c
decreasing.c
increasingparameter.c
recursion > increasingparameter.c > increasing(int, int)
#include<stdio.h>
void increasing(int x, int n){
if(x>n) return;
printf("%d\n",x);
increasing(x+1,n);
return;
}
int main(){
int n;
printf("Enter a number : ");
scanf("%d",&n);
increasing(1,n);
return 0;
}
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc increasingparameter.c -o increasingpar
garg/Desktop/C_Programming/recursion/"increasingparameter
Enter a number : 5
1
2
3
4
5
raghavgarg@Raghavs-MacBook-Pro recursion %

Ques : Print 1 to n (parameterised)

```c
#include<stdio.h>
void increasing(int x, int n){
    if(x>n) return;
    printf("%d\n",x);
    increasing(x+1,n);
    return;
}
int main(){
    int n;
    printf("Enter a number : ");
    scanf("%d",&n);
    increasing(1,5);
    return 0;
}
```

1 5

n = 5

void increasing (int n)

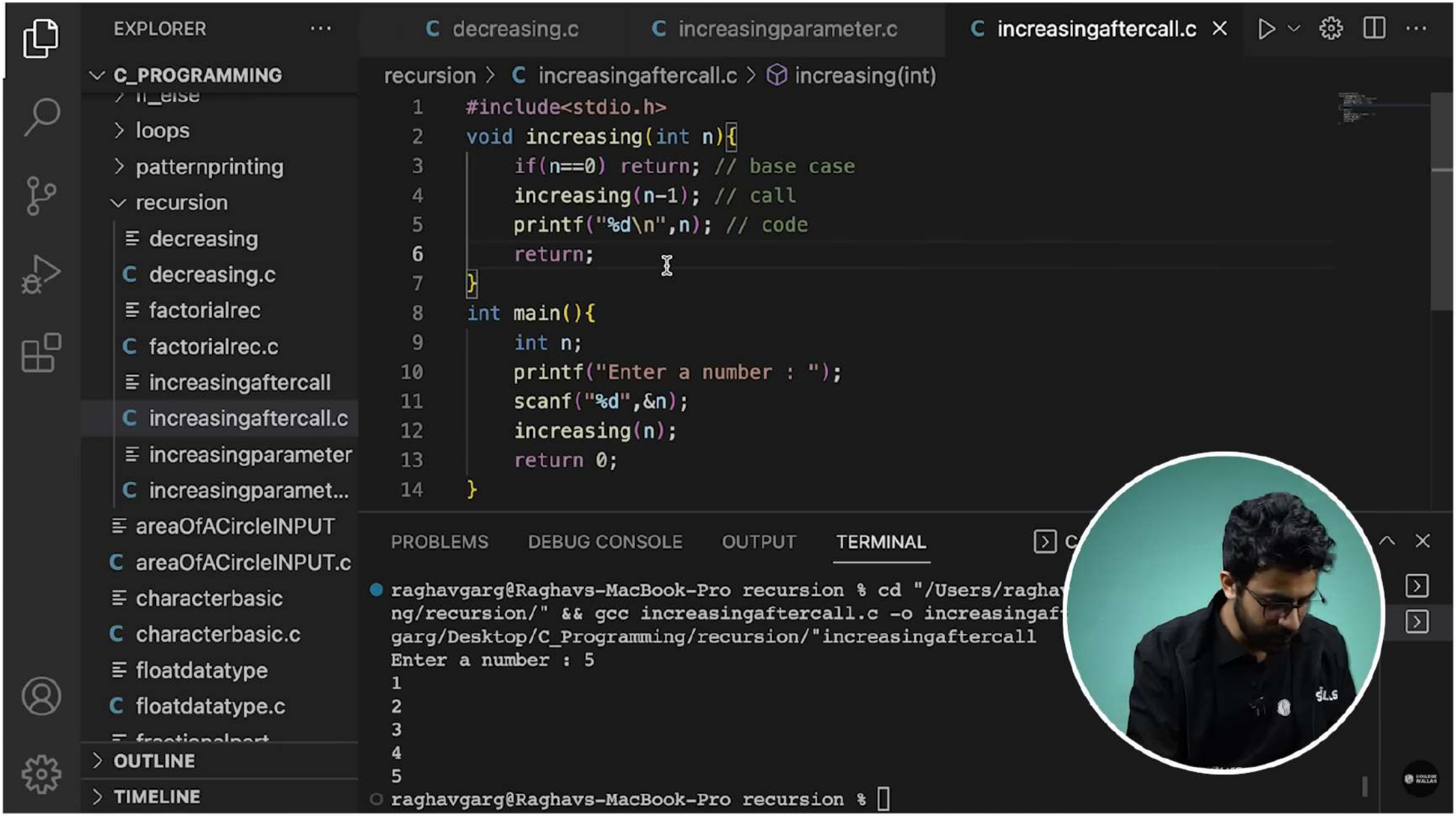
Output

- 1
- 2
- 3
- 4
- 5
-

Ques : Print 1 to n (after recursive call)

```
fun( ){
| base case
|
| code
|
| recursive call
|
| code
|
| return
|
}
```

```
fun( ){
|  base case
|
|  code
|
|  recursive call
|
|  code
|
|  return
|
}
```


EXPLORER
C_PROGRAMMING
loops
patternprinting
recursion
decreasing
decreasing.c
factorialrec
factorialrec.c
increasingaftercall
increasingaftercall.c
increasingparameter
areaOfACircleINPUT
areaOfACircleINPUT.c
characterbasic
characterbasic.c
floatdatatype
floatdatatype.c
OUTLINE
TIMELINE
decreasing.c
increasingparameter.c
increasingaftercall.c
recursion > increasingaftercall.c > increasing(int)
#include<stdio.h>
void increasing(int n){
if(n==0) return; // base case
increasing(n-1); // call
printf("%d\n",n); // code
return;
}
int main(){
int n;
printf("Enter a number : ");
scanf("%d",&n);
increasing(n);
return 0;
}
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc increasingaftercall.c -o increasingaf
garg/Desktop/C_Programming/recursion/"increasingaftercall
Enter a number : 5
1
2
3
4
5
raghavgarg@Raghavs-MacBook-Pro recursion %

Ques : Print 1 to n (after recursive call)

```
void increasing(int n){
    if(n==0) return; // base case
    increasing(n-1); // call
    printf("%d\n",n); // code
    return;
}
```

```
void increasing(int n){
    if(n==0) return; // base case
    increasing(n-1); // call
    printf("%d\n",n); // code
    return;
}
```

```
void increasing(int n){
    if(n==0) return; // base case
    increasing(n-1); // call
    printf("%d\n",n); // code
    return;
}
```

```
void increasing(int n){
    if(n==0) return; // base case
    increasing(n-1); // call
    printf("%d\n",n); // code
    return;
}
```

Output

- 1
- 2
- 3
- 4
-

Homework : Print Decreasing - Increasing

n = 4 →

4
3
2
1
1
2
3
4

Hint : Call se pehle , Call ke baad

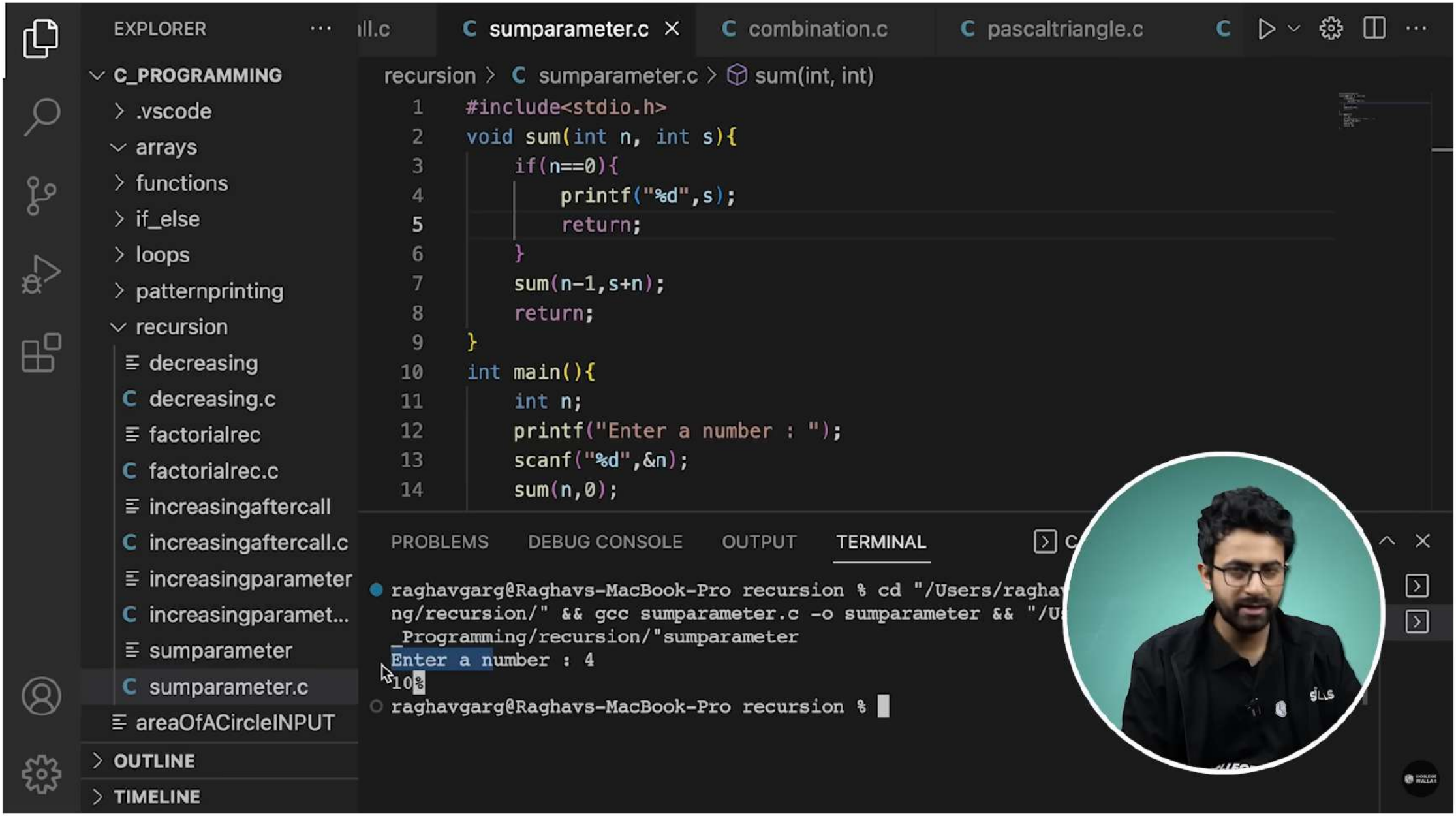
H.W. n = 3 → DRY RUN

Ques : Print sum from 1 to n (Parameterised)

```
                5       0
void sum(int n, int s){
    if(n==0) return;
    sum(n-1,s+n);
    return;
}
```

$S = 1+2+3+4+5$

$S = 5+4+3+2+1$


EXPLORER
C_PROGRAMMING
.vscode
arrays
functions
if_else
loops
patternprinting
recursion
decreasing
decreasing.c
factorialrec
factorialrec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
sumparameter
sumparameter.c
areaOfACircleINPUT
OUTLINE
TIMELINE
ll.c
sumparameter.c
combination.c
pascaltriangle.c
recursion > sumparameter.c > sum(int, int)
#include<stdio.h>
void sum(int n, int s){
if(n==0){
printf("%d",s);
return;
}
sum(n-1,s+n);
return;
}
int main(){
int n;
printf("Enter a number : ");
scanf("%d",&n);
sum(n,0);
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc sumparameter.c -o sumparameter && "/U
_Programming/recursion/"sumparameter
Enter a number : 4
10%
raghavgarg@Raghavs-MacBook-Pro recursion %

n = 4 , s = 0

Ques : Print sum from 1 to n (Parameterised)

```
void sum(int n, int s){
    if(n==0){
        printf("%d",s);
        return;
    }
    sum(n-1,s+n);
    return;
}
```

```
void sum(int n, int s){
    if(n==0){
        printf("%d",s);
        return;
    }
    sum(n-1,s+n);
    return;
}
```

```
void sum(int n, int s){
    if(n==0){
        printf("%d",s);
        return;
    }
    sum(n-1,s+n);
    return;
}
```

```
void sum(int n, int s){
    if(n==0){
        printf("%d",s);
        return;
    }
    sum(n-1,s+n);
    return;
}
```

```
void sum(int n, int s){
    if(n==0){
        printf("%d",s);
        return;
    }
    sum(n-1,s+n);
    return;
}
```

Output

10

Ques : Print sum from 1 to n (Return type)

factorial (n) = n * factorial (n-1) ;

sum(n) = n + sum(n-1) ;

sum(5) = 1 + 2 + 3 + 4 + 5 = 5 + sum(4)

↓

4 + sum(3)

↓

3 + sum(2)

↓

2 + sum(1)

↓

1

Ques : Print sum from 1 to n (Return t

factorial (n) = n * factorial (n-1) ;

sum(n) = n + sum(n-1) ;

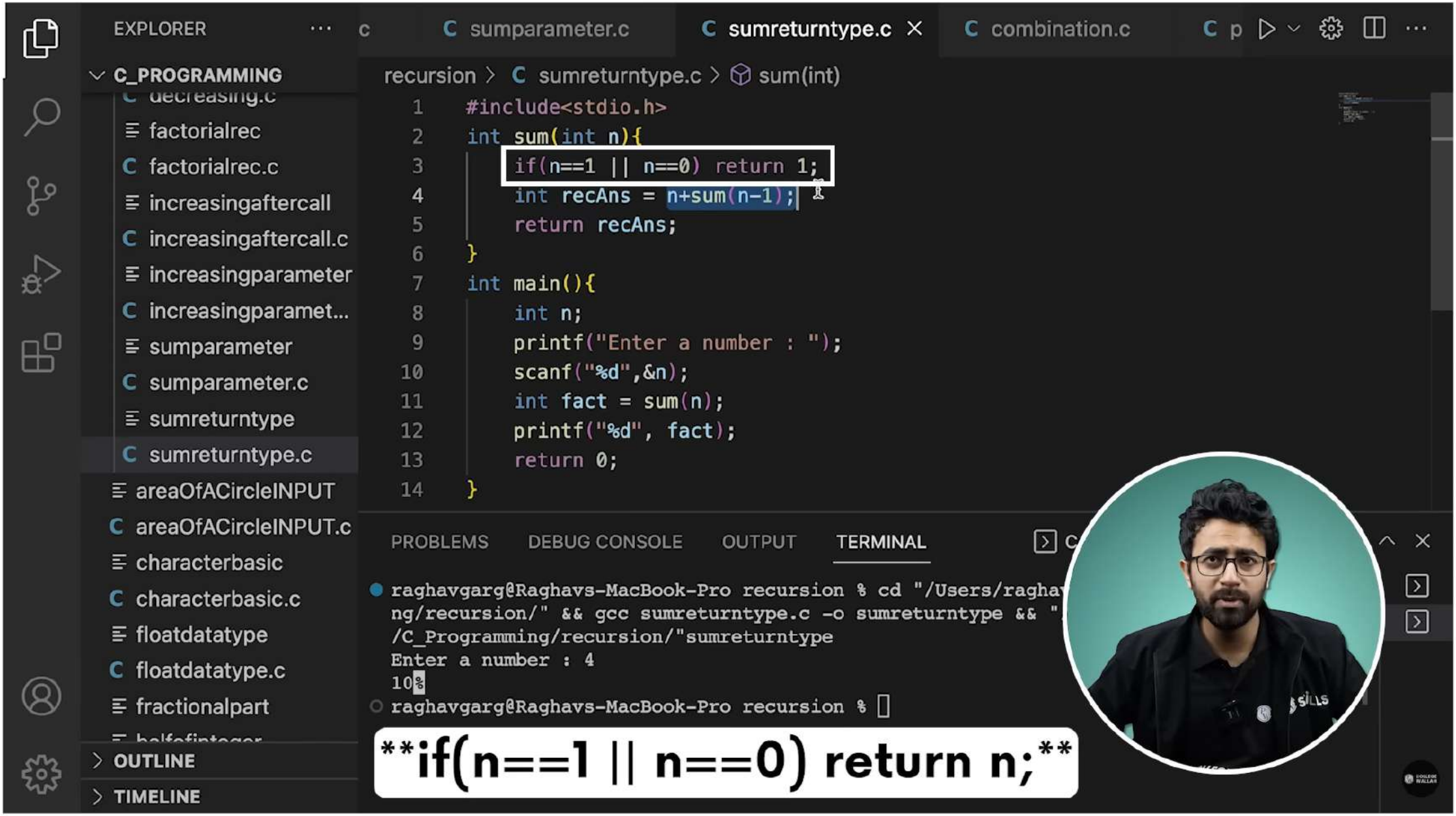
Sum(5) = 1 + 2 + 3 + 4 + 5 = 5 + sum(10) → 15

4 + sum(3) → 6

3 + sum(2) → 6

2 + sum(1) = 3

1


EXPLORER
C_PROGRAMMING
factorialrec
factorialrec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
areaOfACircleINPUT
areaOfACircleINPUT.c
characterbasic
characterbasic.c
floatdatatype
floatdatatype.c
fractionalpart
OUTLINE
TIMELINE
sumparameter.c
sumreturntype.c
combination.c
recursion > sumreturntype.c > sum(int)
#include<stdio.h>
int sum(int n){
if(n==1 || n==0) return 1;
int recAns = n+sum(n-1);
return recAns;
}
int main(){
int n;
printf("Enter a number : ");
scanf("%d",&n);
int fact = sum(n);
printf("%d", fact);
return 0;
}
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc sumreturntype.c -o sumreturntype && "
/C_Programming/recursion/"sumreturntype
Enter a number : 4
10%
raghavgarg@Raghavs-MacBook-Pro recursion %
if(n==1 || n==0) return n;

Ques : Make a function which calculates 'a' raised to the power 'b' using recursion.

```
int power = 1;
int a, b;

for(int i=1; i<=b; i++){
    power = power * a;
}
```

$a^b = \underbrace{a \times a \times a \times a \ldots}_{b \text{ times}}$

Ques : Make a function which calcula
raised to the power 'b' using recursio

```
int power = 1;
int a, b;

for(int i=1; i<=b; i++){
    power = power * a;
}
```

$a^b = \underbrace{a \times a \times a \times a \ldots}_{b \text{ times}}$

```
if (b==0) return 1;
power(a,b) = a * power(a,b-1);
```

3

$$a^b = a * a^{b-1}$$

$2^4 = 2 * 2^3 \longrightarrow 2^3 = 2 * 2^2 \longrightarrow 2^2 = 2 * 2^1 \longrightarrow 2^1 = 2 * 2^0$

$2^0 = 1$

EXPLORER

C_PROGRAMMING

- .vscode
- arrays
- functions
- if_else
- loops
- patternprinting
- recursion
  - decreasing
  - decreasing.c
  - factorialrec
  - factorialrec.c
  - increasingaftercall
  - increasingaftercall.c
  - increasingparameter
  - increasingparamet...
  - powerrec
  - powerrec.c
  - sumparameter

OUTLINE

TIMELINE

c | sumparameter.c | sumreturntype.c | powerrec.c | com

recursion > powerrec.c > power(int, int)

```c
#include<stdio.h>
int power(int a, int b){
    if(b==0) return 1;
    int recAns = a*power(a,b-1);
    return recAns;
}
int main(){
    int a;
    printf("Enter base : ");
    scanf("%d",&a);
    int b;
    printf("Enter power : ");
    scanf("%d",&b);
    int p = power(a,b);
```

PROBLEMS | DEBUG CONSOLE | OUTPUT | TERMINAL

```
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerrec.c -o powerrec && "/Users/rag
ming/recursion/"powerrec
Enter base : 3
Enter power : 4
3 raised to the power 4 is 81%
raghavgarg@Raghavs-MacBook-Pro recursion %
```

*Multiple Calls

Ques : Write a function to calculate the nth fibonacci number using recursion.

1 1 2 3 5 8 13 21 34 55 89 ...

1 2 3 4 5 6 7 8 9 10 11

```
fibo(n) = fibo(n-1) + fibo(n-2);
```

*Multiple Calls

Ques : Write a function to calculate the nth fibonacci number using recursion.

1 1 2 3 5 8 13 21 34 55 89 . . .

a, b, sum

```
sum = a + b;
a = b;
b = sum;
```

*Multiple Calls

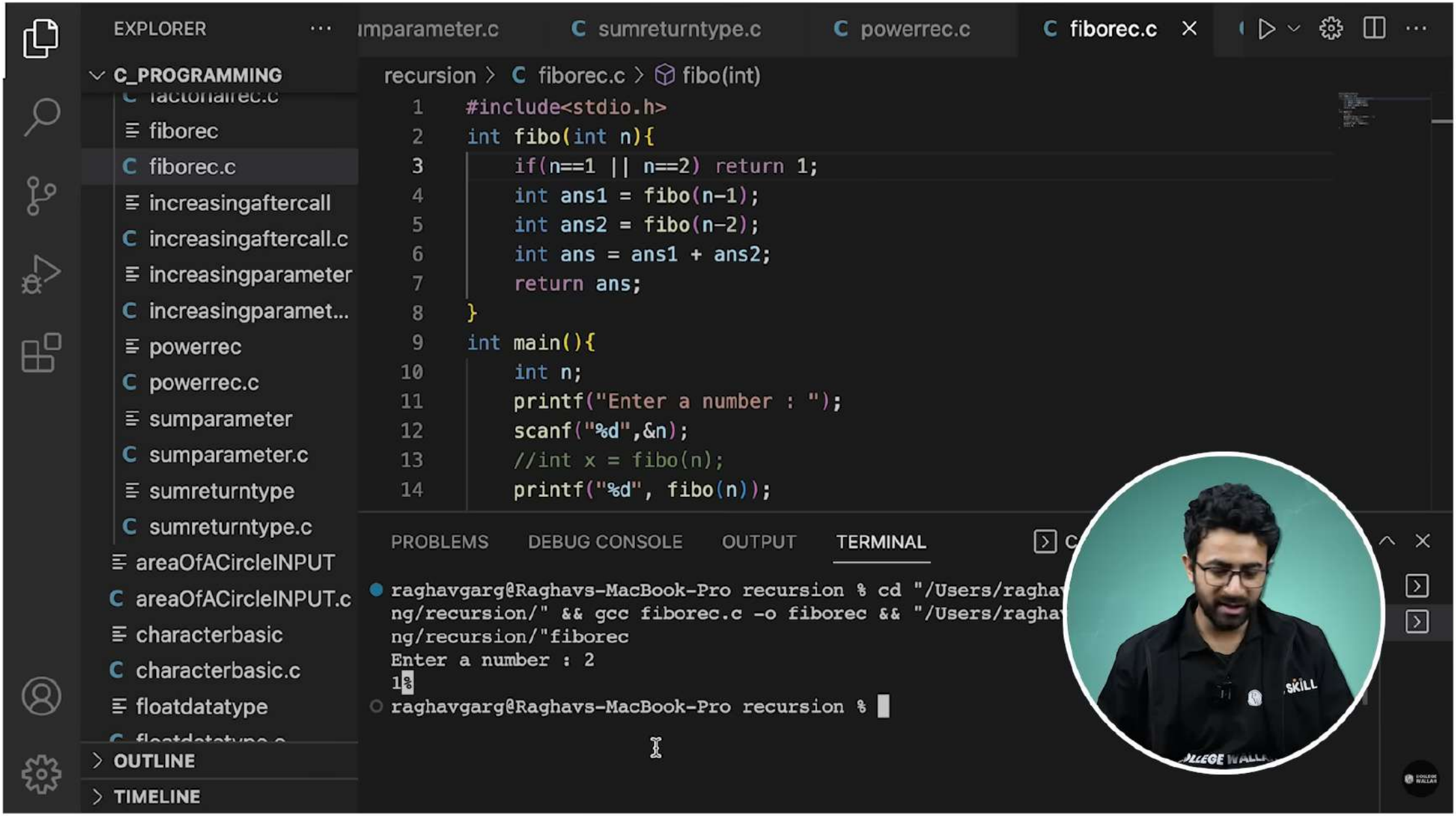
Ques : Write a function to calculate the nth fibonacci number using recursion.

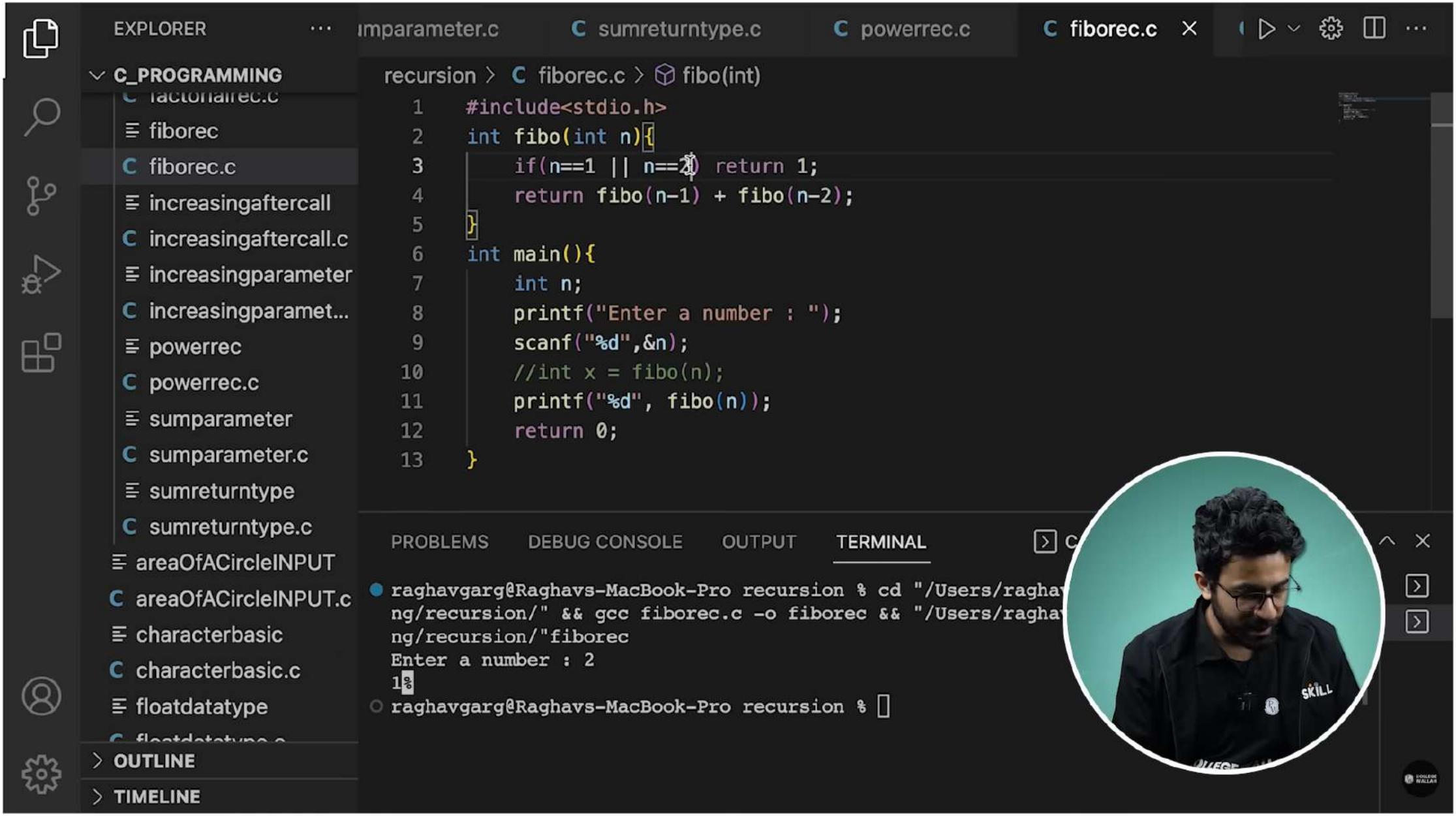
1 1 2 3 5 8 13 21 34 55 89 ...

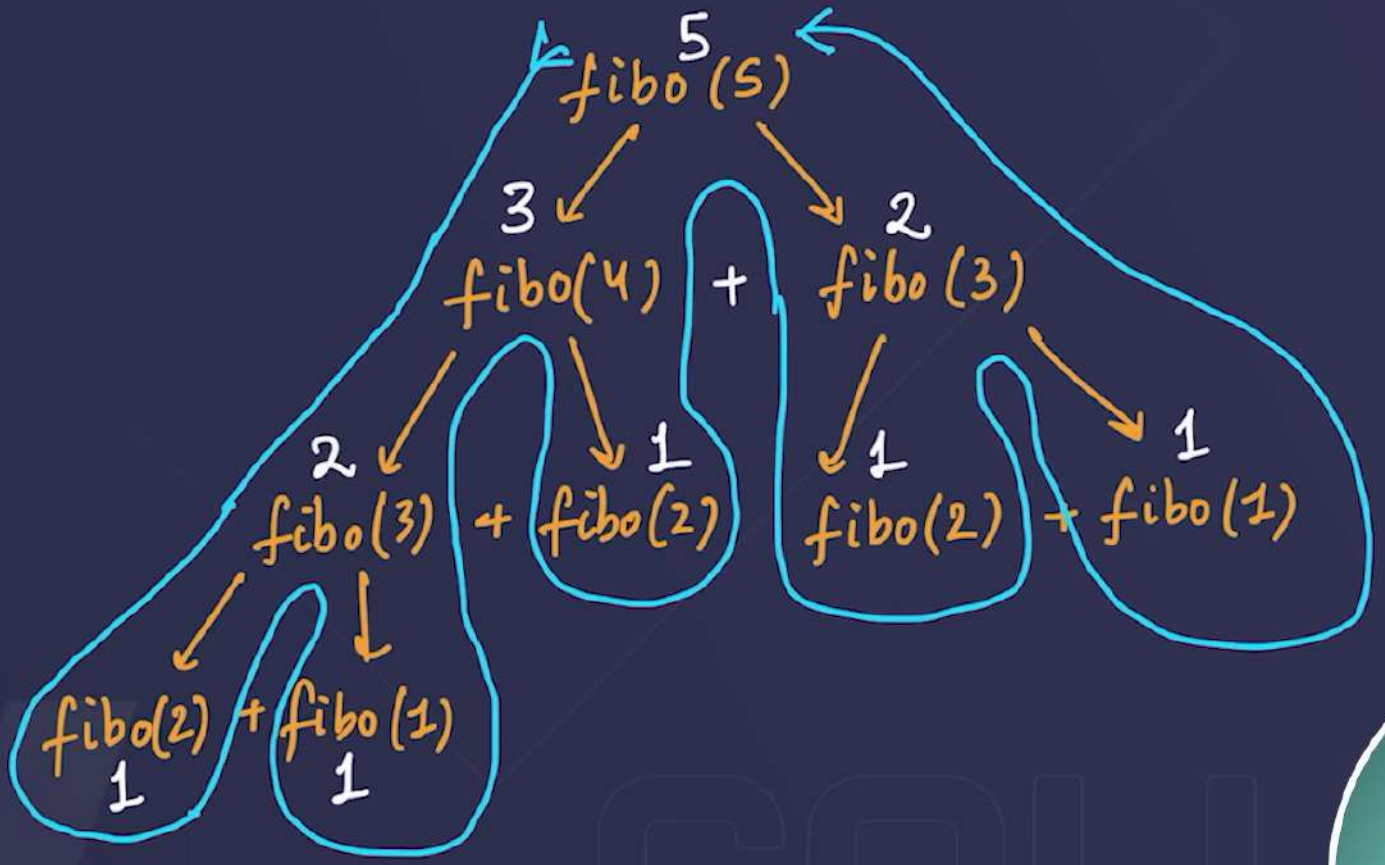
1 2 3 4 5 6 7 8 9 10 11

$$fibo(n) = fibo(n-1) + fibo(n-2);$$

$$fibo(2) = fibo(1) + fibo(0)$$


EXPLORER
C_PROGRAMMING
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerrec
powerrec.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
areaOfACircleINPUT
areaOfACircleINPUT.c
characterbasic
characterbasic.c
floatdatatype
OUTLINE
TIMELINE
imparameter.c
sumreturntype.c
powerrec.c
fiborec.c
recursion > fiborec.c > fibo(int)
#include<stdio.h>
int fibo(int n){
if(n==1 || n==2) return 1;
int ans1 = fibo(n-1);
int ans2 = fibo(n-2);
int ans = ans1 + ans2;
return ans;
}
int main(){
int n;
printf("Enter a number : ");
scanf("%d",&n);
//int x = fibo(n);
printf("%d", fibo(n));
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/ragha
ng/recursion/" && gcc fiborec.c -o fiborec && "/Users/ragha
ng/recursion/"fiborec
Enter a number : 2
1%
raghavgarg@Raghavs-MacBook-Pro recursion %


EXPLORER
C_PROGRAMMING
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerrec
powerrec.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
areaOfACircleINPUT
areaOfACircleINPUT.c
characterbasic
characterbasic.c
floatdatatype
OUTLINE
TIMELINE
imparameter.c
sumreturntype.c
powerrec.c
fiborec.c
recursion > fiborec.c > fibo(int)
#include<stdio.h>
int fibo(int n){
if(n==1 || n==2) return 1;
return fibo(n-1) + fibo(n-2);
}
int main(){
int n;
printf("Enter a number : ");
scanf("%d",&n);
//int x = fibo(n);
printf("%d", fibo(n));
return 0;
}
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc fiborec.c -o fiborec && "/Users/raghav
ng/recursion/"fiborec
Enter a number : 2
1%
raghavgarg@Raghavs-MacBook-Pro recursion %



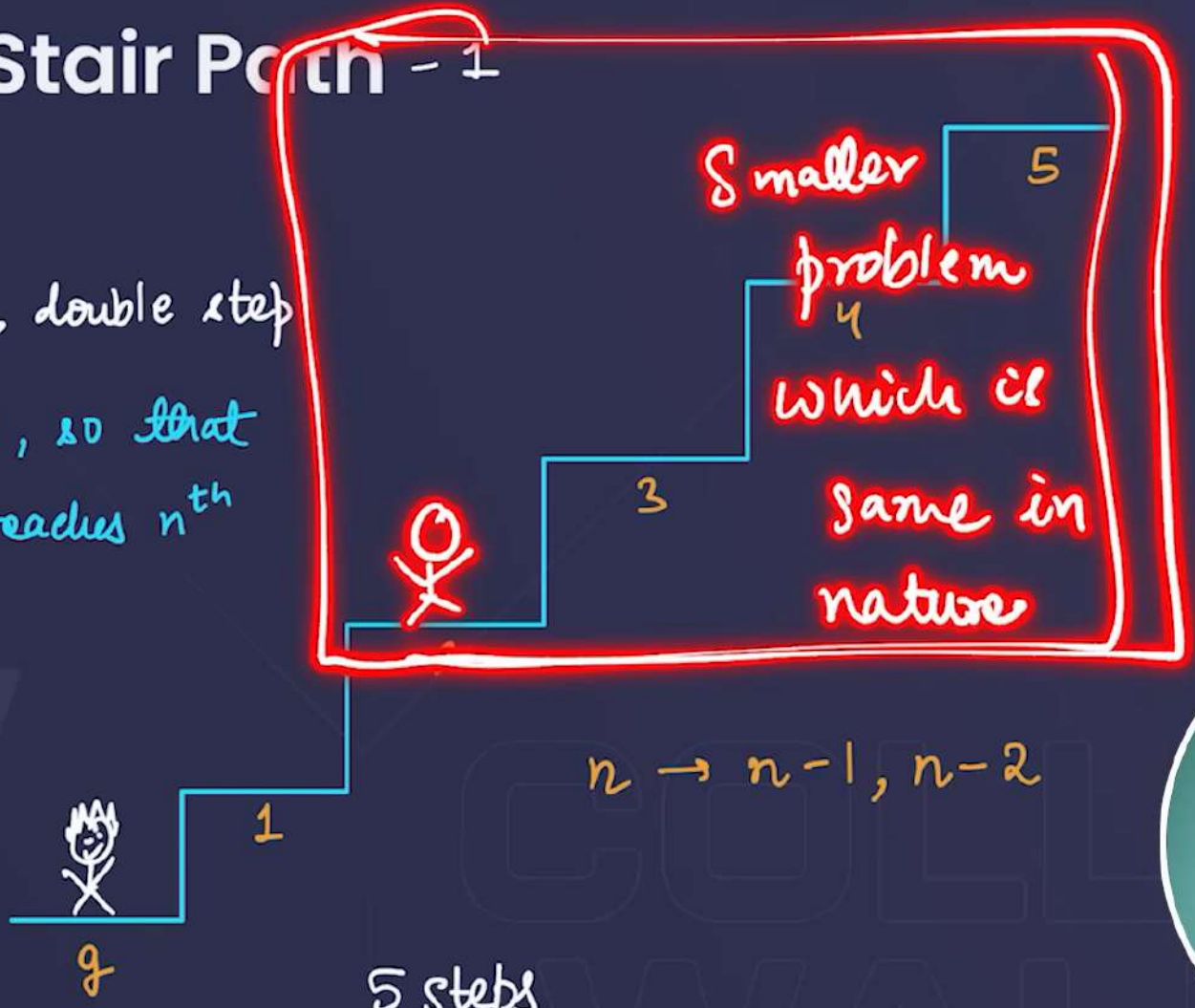
(n==1 || n==2)
return 1

Ques : Stair Path - 1

n^{th} stair

single step, double step

no. of ways, so that the person reaches n^{th} stair.



$n \rightarrow n-1, n-2$

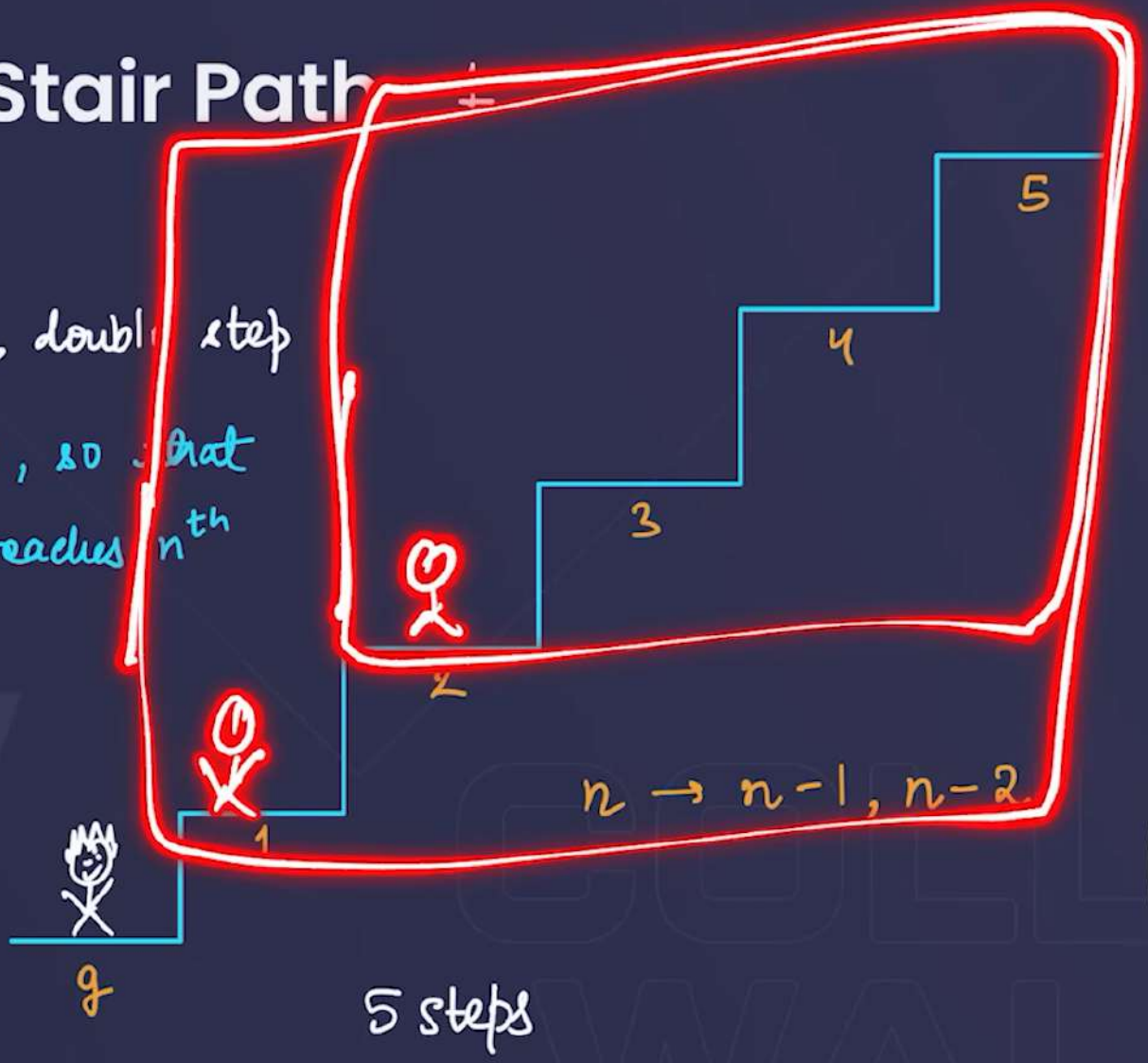
1 1 1 1 1
1 1 1 2
1 2 1 1
1 1 2 1
2 1 1 1
1 2 2

Ques : Stair Path

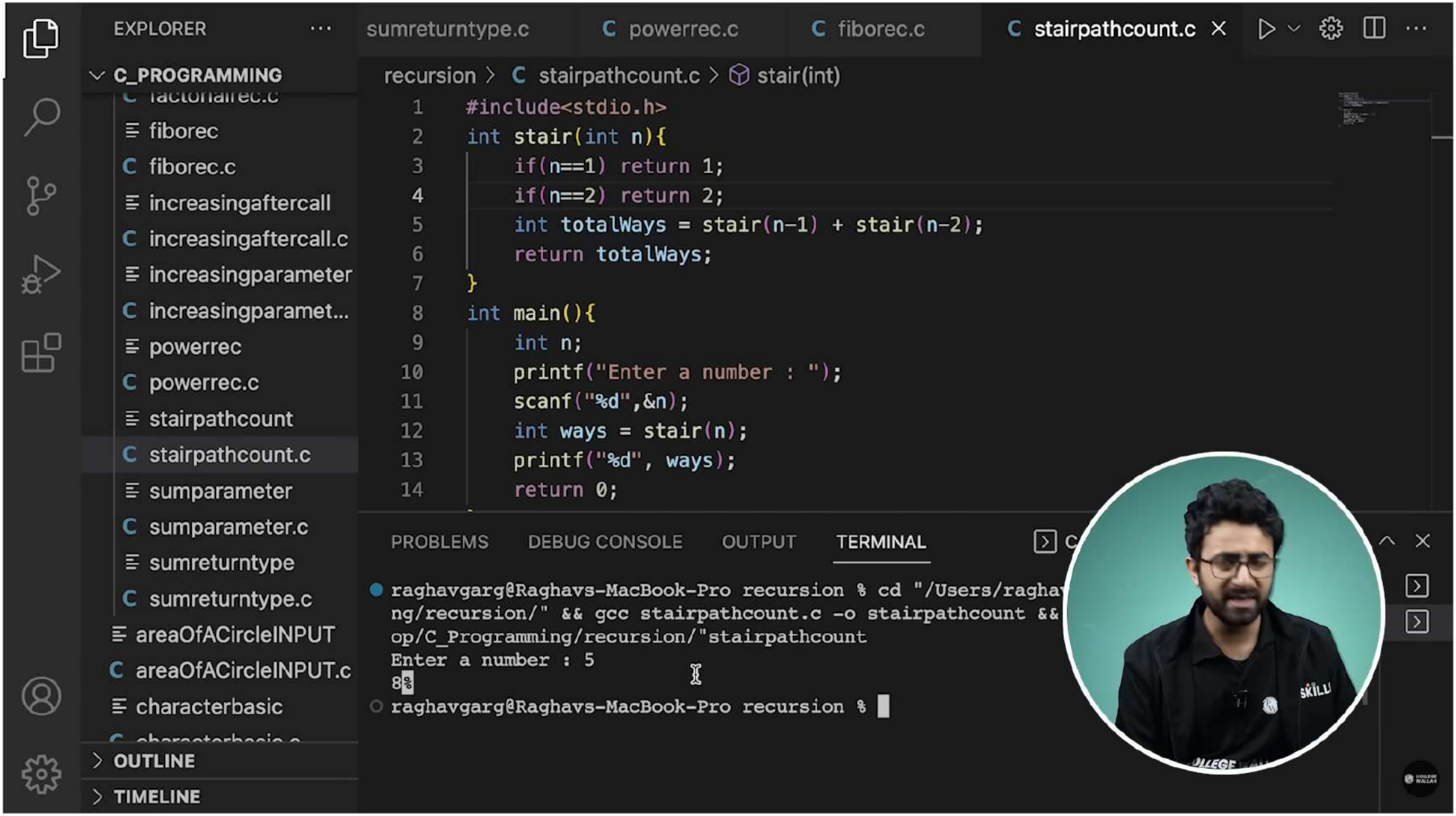
n^{th} stair

single step, double step

no. of ways, so that the person reaches n^{th} stair.



1 1 1 1 1
1 1 1 2
1 2 1 1
+ 1 2 1
2 1 1 1
1 2 2

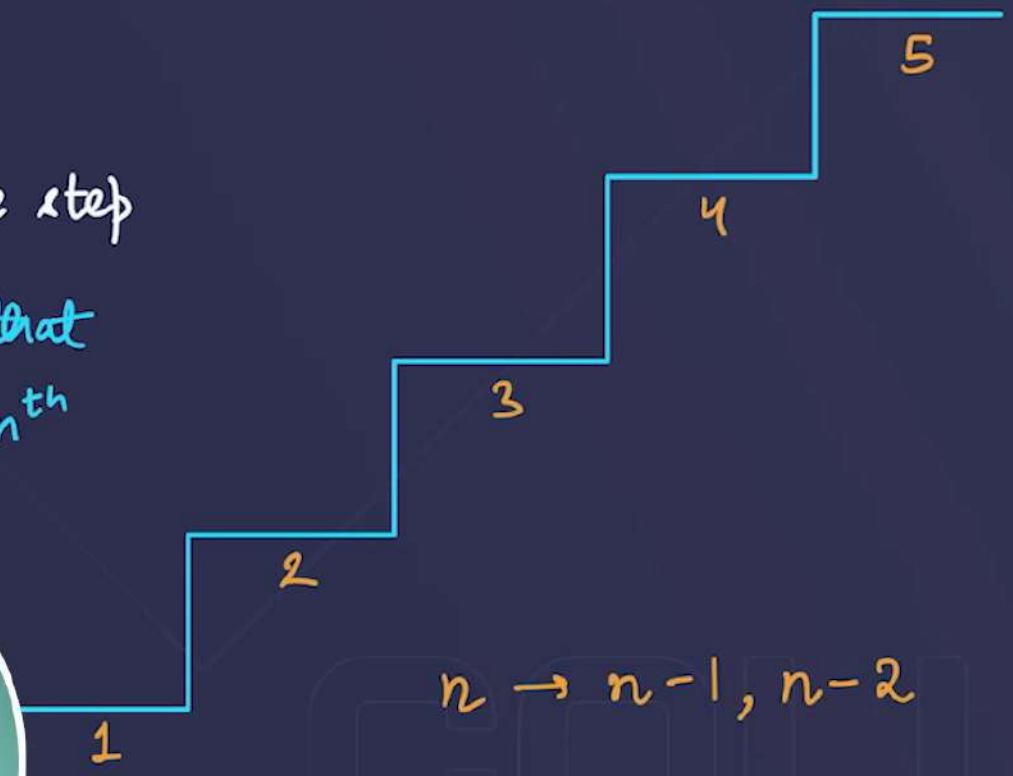

EXPLORER
C_PROGRAMMING
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
areaOfACircleINPUT
areaOfACircleINPUT.c
characterbasic
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
fiborec.c
stairpathcount.c
recursion > stairpathcount.c > stair(int)
1 #include<stdio.h>
2 int stair(int n){
3 if(n==1) return 1;
4 if(n==2) return 2;
5 int totalWays = stair(n-1) + stair(n-2);
6 return totalWays;
7 }
8 int main(){
9 int n;
10 printf("Enter a number : ");
11 scanf("%d",&n);
12 int ways = stair(n);
13 printf("%d", ways);
14 return 0;
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc stairpathcount.c -o stairpathcount &&
op/C_Programming/recursion/"stairpathcount
Enter a number : 5
8%
raghavgarg@Raghavs-MacBook-Pro recursion %

Ques : Stair Path - 1

n^{th} stair

single step, double step

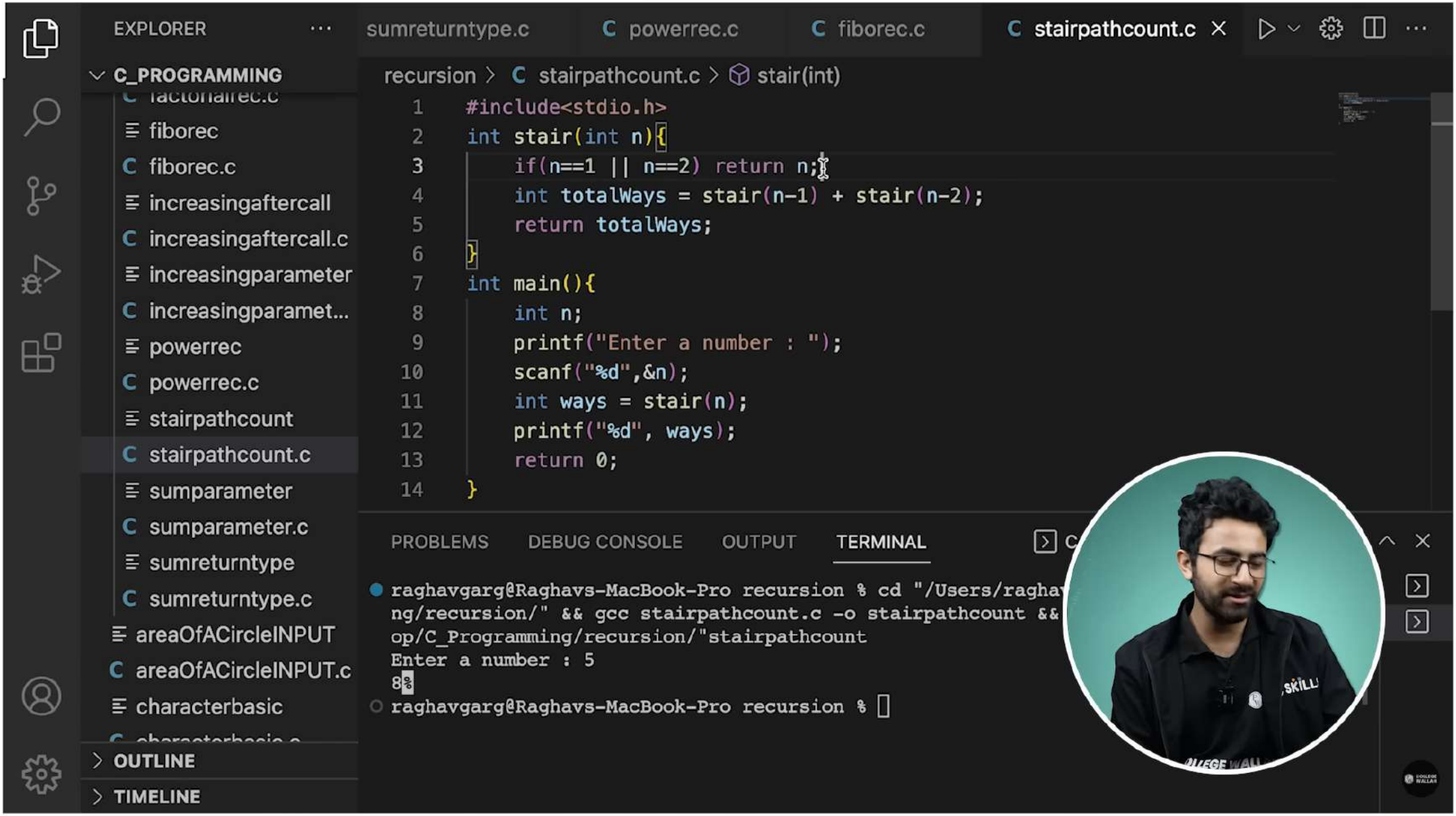
no. of ways, so that the person reaches n^{th} stair.



$n \rightarrow n-1, n-2$

5 steps

1	1	1	1	✓
1	1	1	2	✓
1	2	1		✓
1	1	2	1	✓
2	1			✓
1	2			✓
2	1			✓
2	2	1		✓


EXPLORER
C_PROGRAMMING
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
areaOfACircleINPUT
areaOfACircleINPUT.c
characterbasic
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
fiborec.c
stairpathcount.c
recursion > C stairpathcount.c > stair(int)
1 #include<stdio.h>
2 int stair(int n){
3 if(n==1 || n==2) return n;
4 int totalWays = stair(n-1) + stair(n-2);
5 return totalWays;
6 }
7 int main(){
8 int n;
9 printf("Enter a number : ");
10 scanf("%d",&n);
11 int ways = stair(n);
12 printf("%d", ways);
13 return 0;
14 }
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc stairpathcount.c -o stairpathcount &&
op/C_Programming/recursion/"stairpathcount
Enter a number : 5
8
raghavgarg@Raghavs-MacBook-Pro recursion %

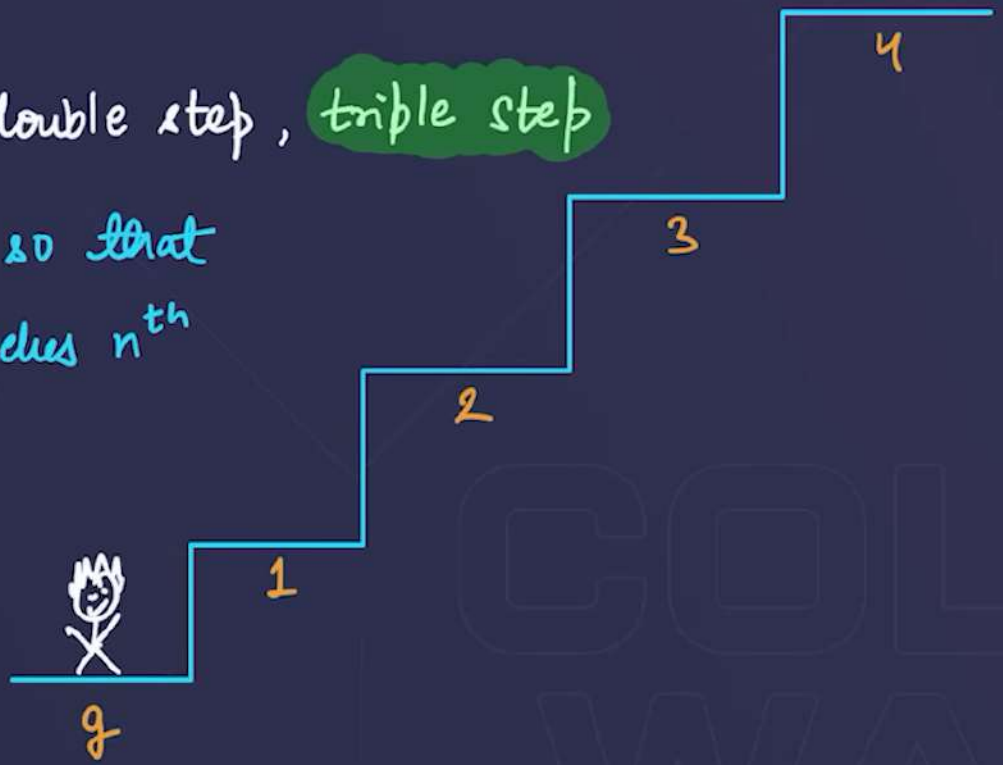
~~Ques~~ : Stair Path - 2

H.W.

n^{th} stair

single step, double step, triple step

no. of ways, so that the person reaches n^{th} stair.



1 1 1 1

1 1 2

1 2 1

2 1 1

2 2

1 3

3 1

Ques : Power function (logarithmic)

$a^b = a * a^{b-1}$; `if (b == 0) return 1;`

$2^{100} = 2 * 2^{99}$

$2^{99} = 2 * 2^{98}$

$2^{98} = 2 * 2^{97}$

.....

$2^1 = 2 * 2^0$

100 calls

Ques : Power function (logarithmic)

$a^b = a * a^{b-1}$;

```
if (b == 0) return 1;
```

$2^{64} = 2 * 2^{63}$

$2^{63} = 2 * 2^{62}$

$\vdots$

$2^1 = 2 * 2^0$

64 calls

$a^b = a^{b/2} * a^{b/2}$;

$2^{64} = 2^{32} * 2^{32}$

$2^{32} = 2^{16} * 2^{16}$

$2^{16} = 2^8 * 2^8$

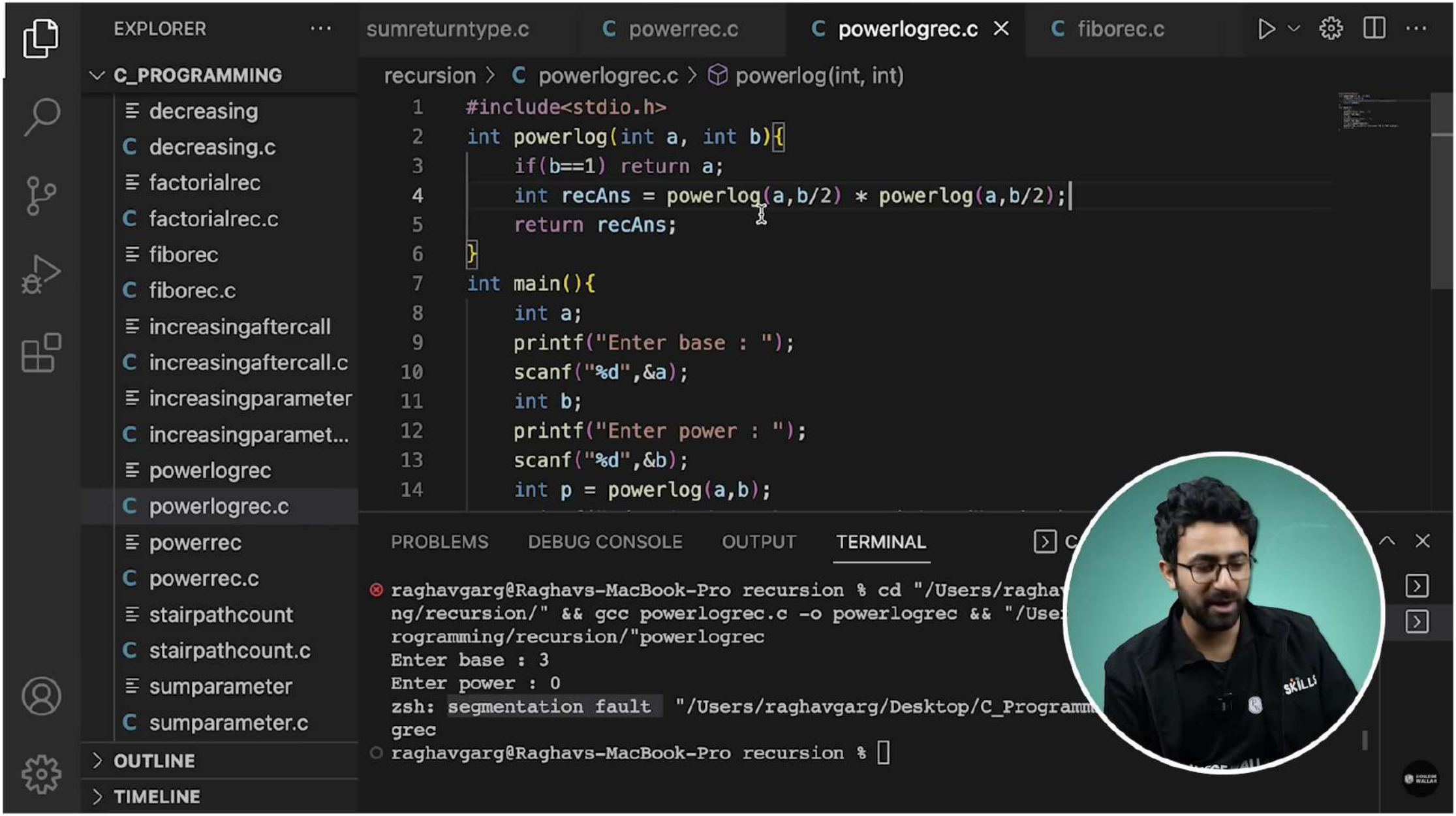
$2^8 = 2^4 * 2^4$

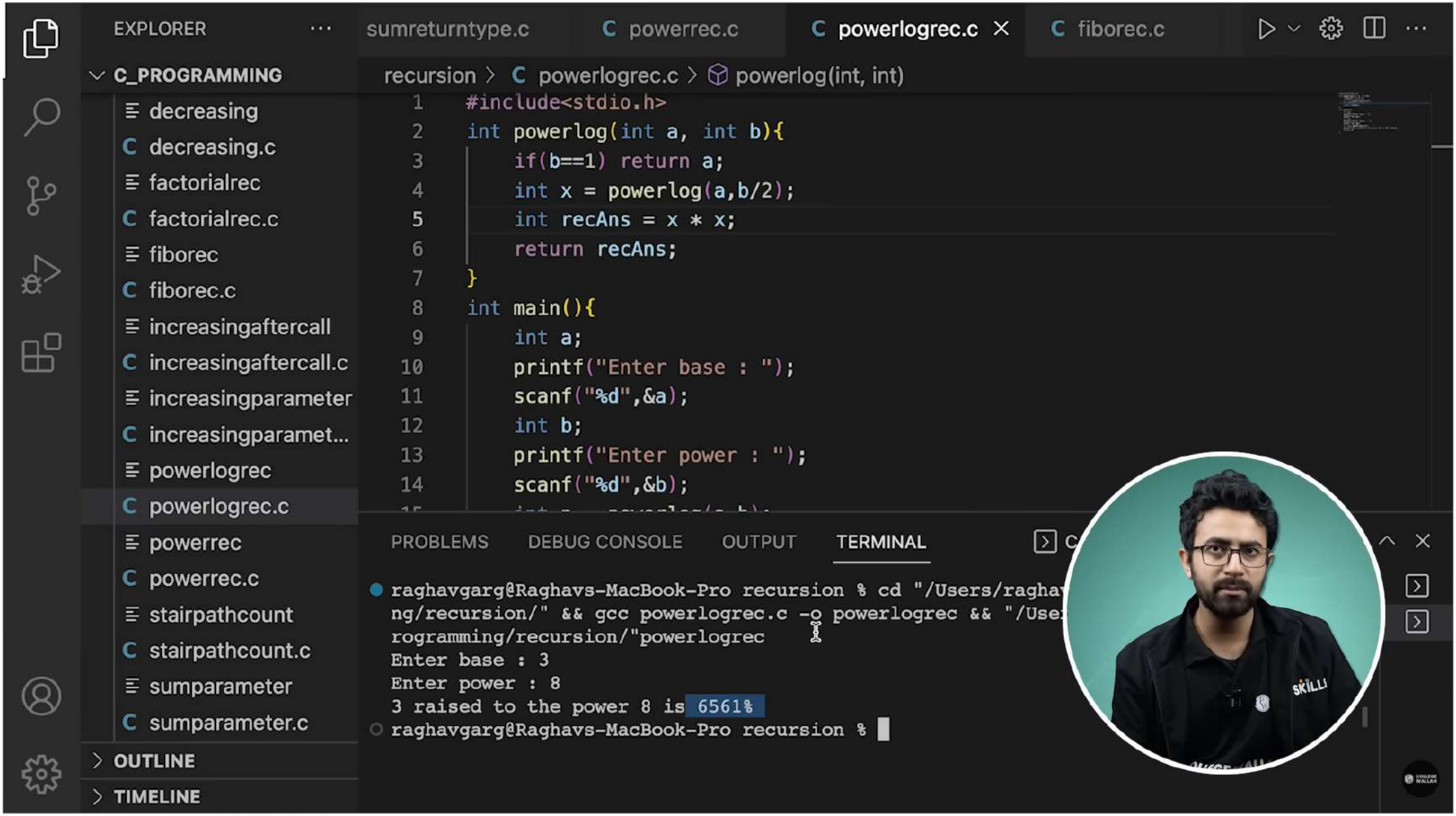
$2^4 = 2^2 * 2^2$

$2^2 = 2^1 * 2^1$

6 calls

```
if (b == 1) return a;
```


EXPLORER
C_PROGRAMMING
decreasing
decreasing.c
factorialrec
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
powerlogrec.c
fiborec.c
recursion > powerlogrec.c > powerlog(int, int)
#include<stdio.h>
int powerlog(int a, int b){
if(b==1) return a;
int recAns = powerlog(a,b/2) * powerlog(a,b/2);
return recAns;
}
int main(){
int a;
printf("Enter base : ");
scanf("%d",&a);
int b;
printf("Enter power : ");
scanf("%d",&b);
int p = powerlog(a,b);
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerlogrec.c -o powerlogrec && "/User
rogramming/recursion/"powerlogrec
Enter base : 3
Enter power : 0
zsh: segmentation fault "/Users/raghavgarg/Desktop/C_Programm
grec
raghavgarg@Raghavs-MacBook-Pro recursion %


EXPLORER
C_PROGRAMMING
decreasing
decreasing.c
factorialrec
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
powerlogrec.c
fiborec.c
recursion > powerlogrec.c > powerlog(int, int)
#include<stdio.h>
int powerlog(int a, int b){
if(b==1) return a;
int x = powerlog(a,b/2);
int recAns = x * x;
return recAns;
}
int main(){
int a;
printf("Enter base : ");
scanf("%d",&a);
int b;
printf("Enter power : ");
scanf("%d",&b);
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerlogrec.c -o powerlogrec && "/User
rogramming/recursion/"powerlogrec
Enter base : 3
Enter power : 8
3 raised to the power 8 is 6561%
raghavgarg@Raghavs-MacBook-Pro recursion %

Ques : Power function (logarithmic)

$$a^b = a^{b/2} * a^{b/2}$$

Problem :

$$2^7 = 2^{7/2} * 2^{7/2}$$

$$2^7 = 2^3 * 2^3 = 16$$

$$2^3 = 2^{3/2} * 2^{3/2}$$

$$2^3 = 2^1 * 2^1$$

$$2^3 = 2 \times 2 = 4$$

Solution :

if b is even

$$a^b = a^{b/2} * a^{b/2}$$

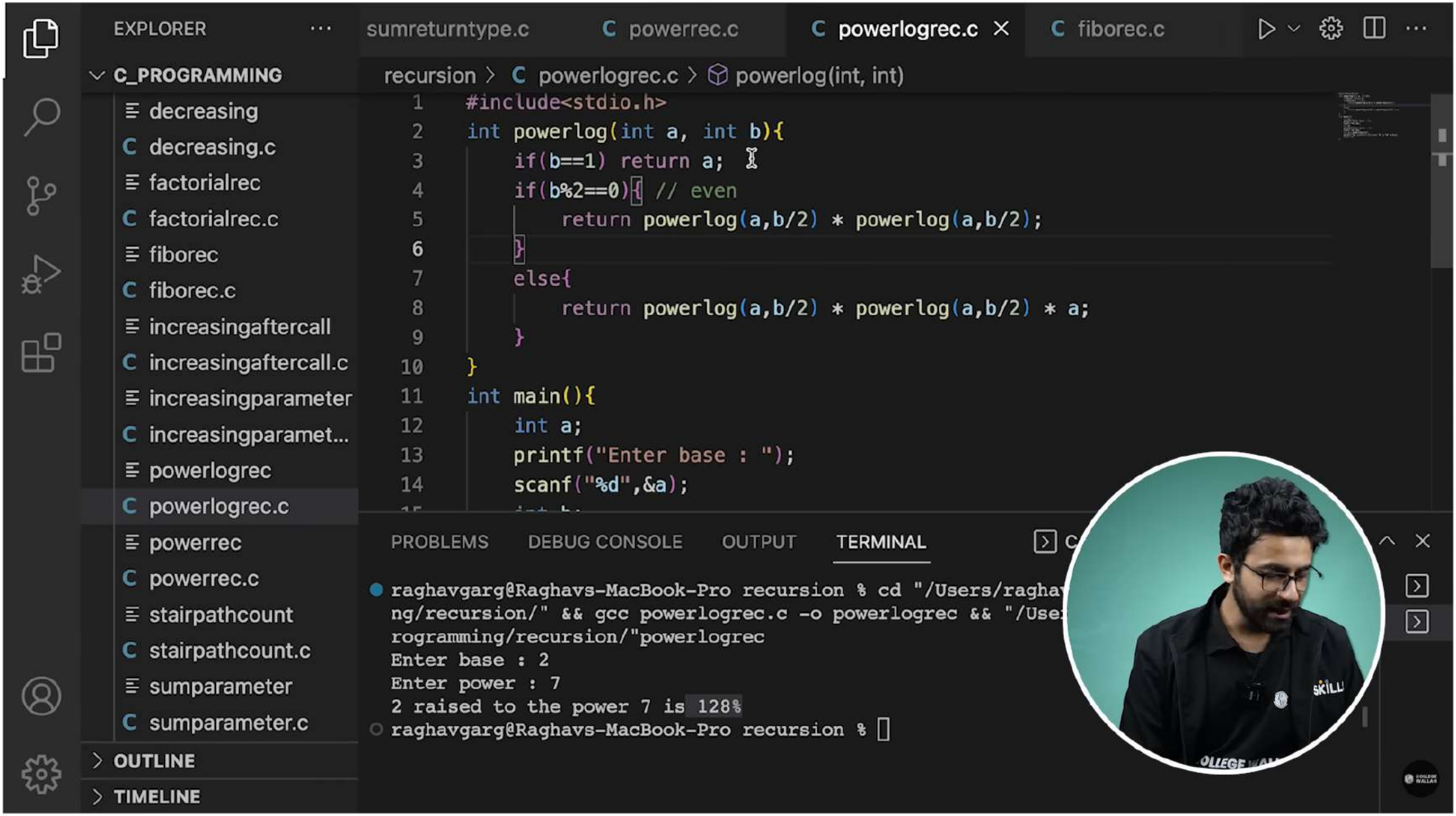
if b is odd

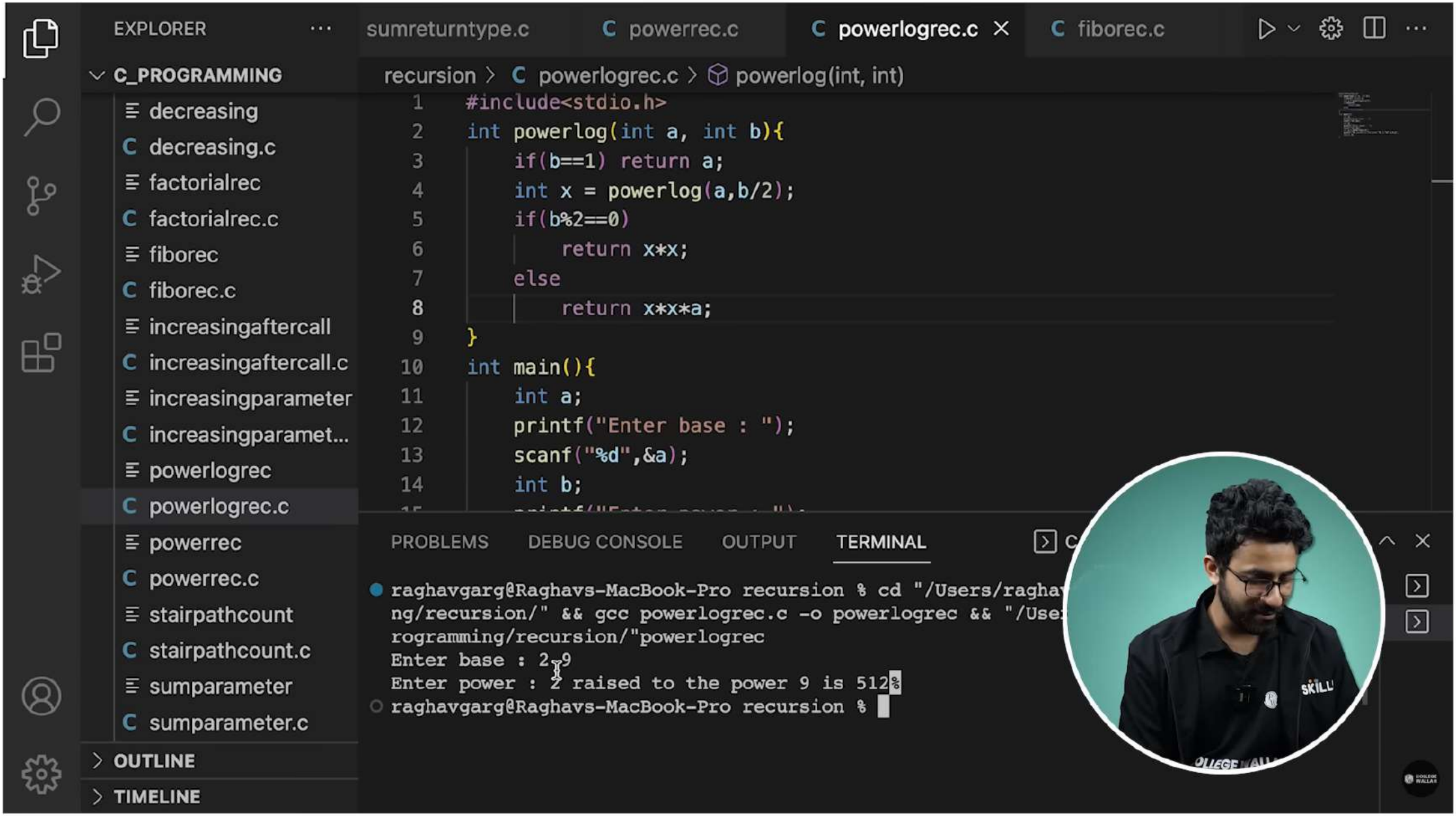
$$a^b = a^{b/2} * a^{b/2} * a$$

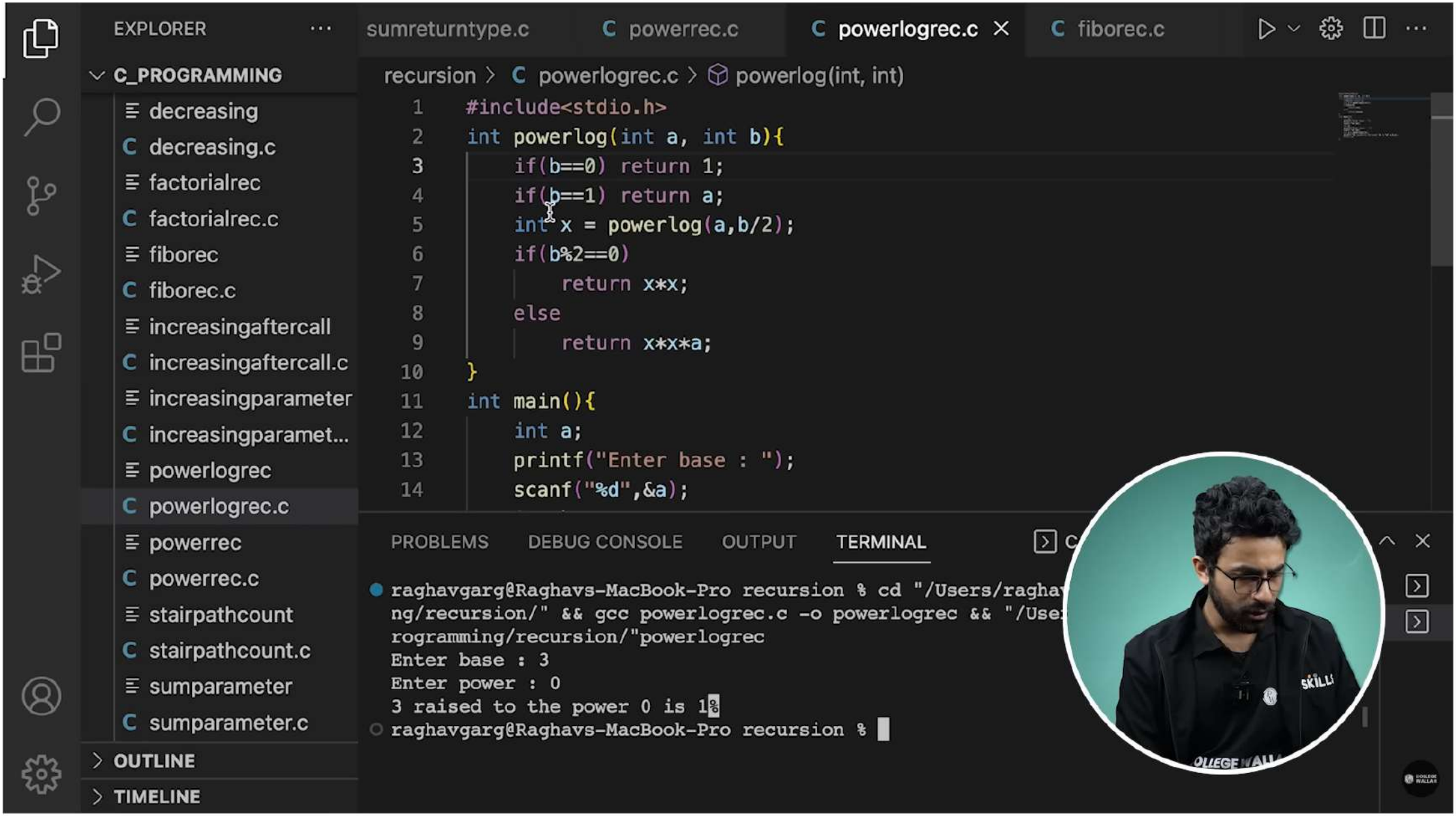
b = 5

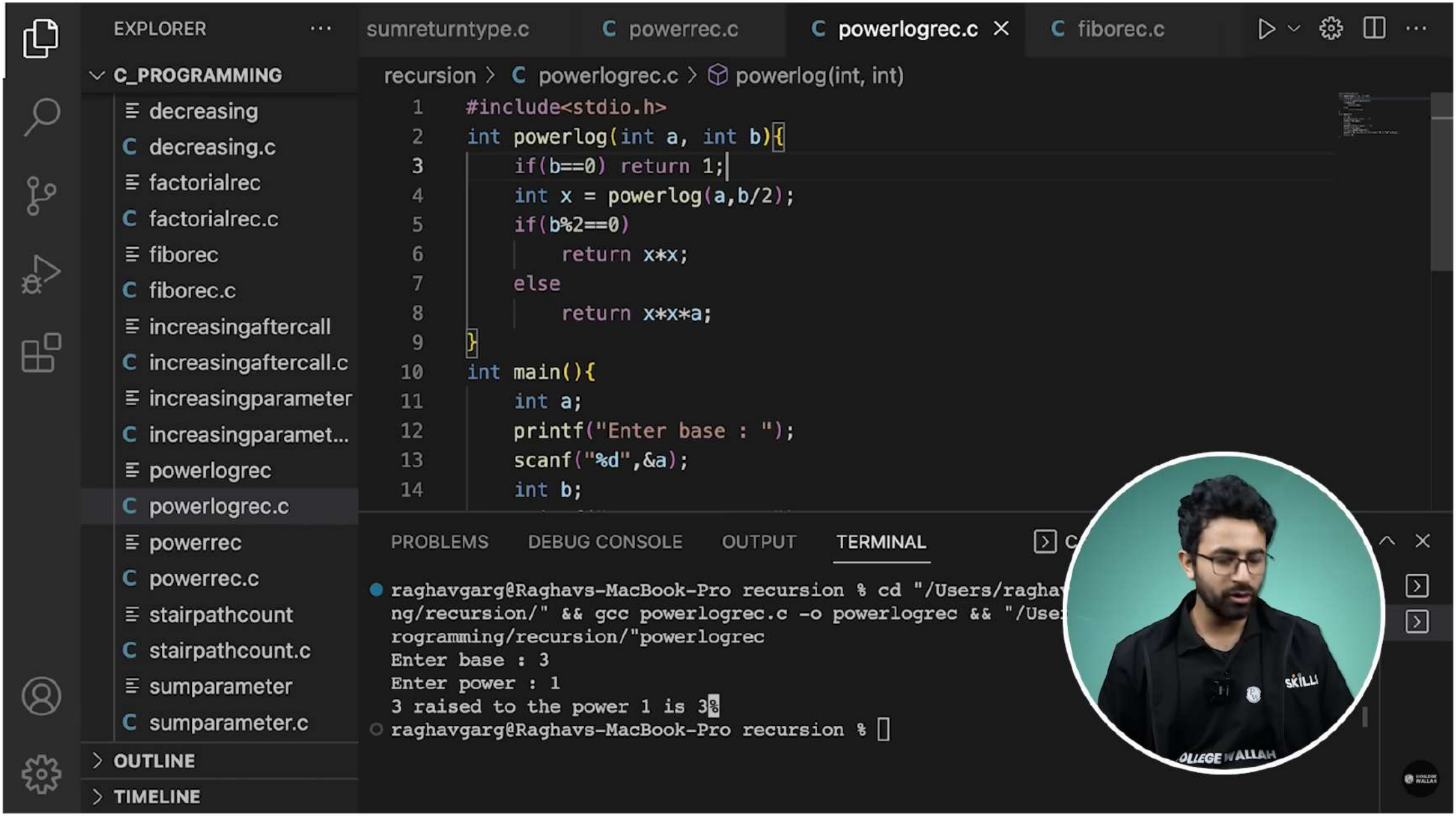
$$a^5 = a^{5/2} \times a^{5/2} \times a$$

$$= a^2 \times a^2 \times a$$


EXPLORER
C_PROGRAMMING
decreasing
decreasing.c
factorialrec
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
powerlogrec.c
fiborec.c
recursion > powerlogrec.c > powerlog(int, int)
#include<stdio.h>
int powerlog(int a, int b){
if(b==1) return a;
if(b%2==0){ // even
return powerlog(a,b/2) * powerlog(a,b/2);
}
else{
return powerlog(a,b/2) * powerlog(a,b/2) * a;
}
}
int main(){
int a;
printf("Enter base : ");
scanf("%d",&a);
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerlogrec.c -o powerlogrec && "/User
rogramming/recursion/"powerlogrec
Enter base : 2
Enter power : 7
2 raised to the power 7 is 128%
raghavgarg@Raghavs-MacBook-Pro recursion %


EXPLORER
C_PROGRAMMING
decreasing
decreasing.c
factorialrec
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
powerlogrec.c
fiborec.c
recursion > powerlogrec.c > powerlog(int, int)
#include<stdio.h>
int powerlog(int a, int b){
if(b==1) return a;
int x = powerlog(a,b/2);
if(b%2==0)
return x*x;
else
return x*x*a;
}
int main(){
int a;
printf("Enter base : ");
scanf("%d",&a);
int b;
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerlogrec.c -o powerlogrec && "/User
rogramming/recursion/"powerlogrec
Enter base : 2 9
Enter power : 2 raised to the power 9 is 512%
raghavgarg@Raghavs-MacBook-Pro recursion %


EXPLORER
C_PROGRAMMING
decreasing
decreasing.c
factorialrec
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
powerlogrec.c
fiborec.c
recursion > powerlogrec.c > powerlog(int, int)
#include<stdio.h>
int powerlog(int a, int b){
if(b==0) return 1;
if(b==1) return a;
int x = powerlog(a,b/2);
if(b%2==0)
return x*x;
else
return x*x*a;
}
int main(){
int a;
printf("Enter base : ");
scanf("%d",&a);
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerlogrec.c -o powerlogrec && "/User
rogramming/recursion/"powerlogrec
Enter base : 3
Enter power : 0
3 raised to the power 0 is 1
raghavgarg@Raghavs-MacBook-Pro recursion %


EXPLORER
C_PROGRAMMING
decreasing
decreasing.c
factorialrec
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
powerlogrec.c
fiborec.c
recursion > powerlogrec.c > powerlog(int, int)
#include<stdio.h>
int powerlog(int a, int b){
if(b==0) return 1;
int x = powerlog(a,b/2);
if(b%2==0)
return x*x;
else
return x*x*a;
}
int main(){
int a;
printf("Enter base : ");
scanf("%d",&a);
int b;
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerlogrec.c -o powerlogrec && "/User
rogramming/recursion/"powerlogrec
Enter base : 3
Enter power : 1
3 raised to the power 1 is 3%
raghavgarg@Raghavs-MacBook-Pro recursion %
SKILLS
OLLEGE WALLAH

```
int powerlog(int a, int b){
    if(b==0) return 1;
    int x = powerlog(a,b/2);
    if(b%2==0)
        return x*x;
    else
        return x*x*a;
}
```

$2^9 = 2^4 * 2^4 * 2;$

$2^4 = 2^2 * 2^2;$

$2^2 = 2^1 * 2^1;$

$2^1 = 2^0 * 2^0 * 2;$

2^0

1

a = 2

b = 9

```
int powerlog(int a, int b){
    if(b==0) return 1;
    int x = powerlog(a,b/2);
    if(b%2==0)
        return x*x;
    else
        return x*x*a;
}
```

$2^9 = 16^4 \times 16^1 \times 2;$ 512

$16^4 = 4^4 \times 4^4;$

$4^4 = 2^4 \times 2^4;$

$2^1 = 2^0 * 2^0 * 2;$

1

a = 2

b = 9

```
int powerlog(int a, int b){
    if(b==0) return 1;
    int x = powerlog(a,b/2);
    if(b%2==0)
        return x*x;
    else
        return x*x*a;
}
```

$2^9 = 2^4 * 2^4 * 2;$

$2^4 = 2^2 * 2^2;$

$2^2 = 2^1 * 2^1;$

$2^1 = 2^0 * 2^0 * 2;$

$2^0 \rightarrow 1$

$a = 2$

$b = 9$

$b \rightarrow b/2 \rightarrow b/1 \rightarrow b/0 \cdots \rightarrow 1$

$a^b \Rightarrow$ b calls

$2^{100} =$ 100 calls

$a^b \rightarrow$ b calls

M-2 : $b + \frac{b}{2} + \frac{b}{4} + \frac{b}{8} + \cdots 2 \quad 1$ G.P.

b ↓ first term

common ratio : $1/2$

$a^b \rightarrow$ b calls

M-2 : $b + \frac{b}{2} + \frac{b}{4} + \frac{b}{8} + \cdots 2 \quad 1$ G.P.

first term

common ratio : $1/2$

no. of terms

$a^b \rightarrow$ b calls

M-2 : $b + \frac{b}{2} + \frac{b}{4} + \frac{b}{8} + \cdots 2\ 1$ G.P.

$1 \xrightarrow{\times 2} 2\ 4 \ldots \frac{b}{4}\ \frac{b}{2}\ b$

n terms

$b = a_n$

$a_1 = 1$

$r = 2$

$\rightarrow$

$a_n = a_1 \times r^{n-1}$

$b = 1 \times 2^{n-1}$

$\rightarrow b = \frac{2^n}{2} \Rightarrow 2^n = 2b$

$\Rightarrow \boxed{n = \log_2 2b}$

$a^b \rightarrow$ b calls $\log_2(b)$ calls

M-2 : $b + \frac{b}{2} + \frac{b}{4} + \frac{b}{8} + \dots 2\ 1$ G.P.

$\times 2$

$1\ \ 2\ \ 4\ \dots\ \frac{b}{4}\ \ \frac{b}{2}\ \ b$

n terms

$a_n = a_1 \times r^{n-1}$

$b = 1 \times 2^{n-1}$

$\rightarrow b = \frac{2^n}{2} \Rightarrow 2^n = 2b$

$\Rightarrow n = \log_2 2b$

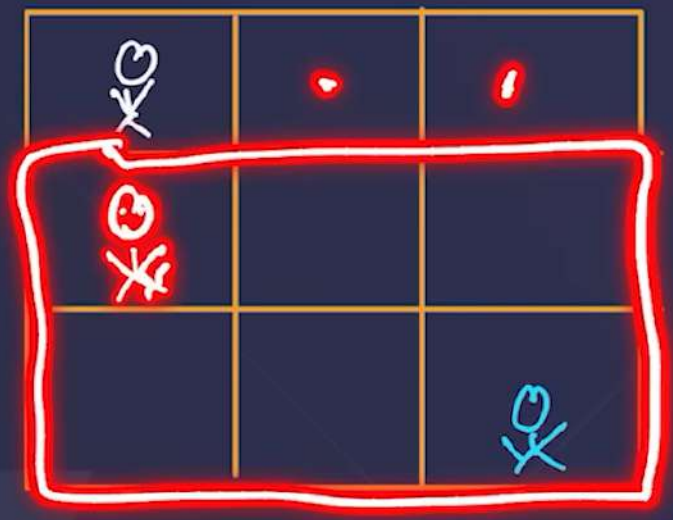
$n = \log(b) - \log(2)$

Ques : Maze path

n, m

no. of ways ⟶ 'Down, Right'

'1 step at a time'



DDRR

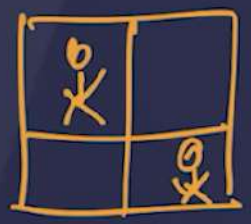
DRDR

DRRD

RRDD

RDRD

RDDR

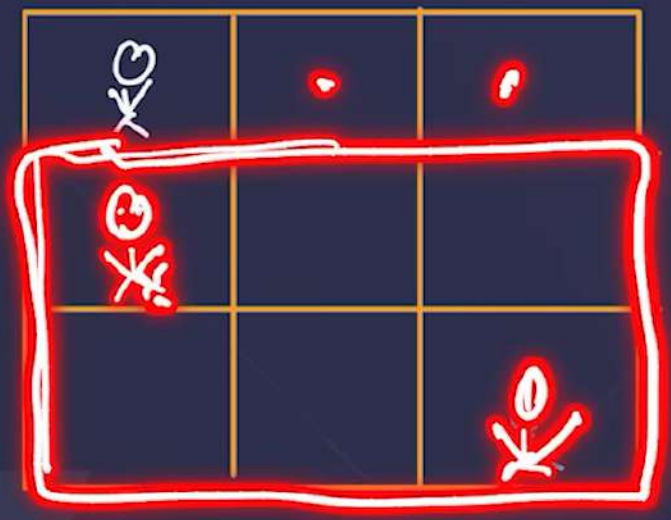


Ques : Maze path

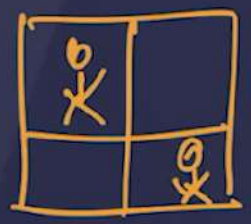
n, m

no. of ways → 'Down, Right'

'1 step at a time'



DDRR
DRDR
DRRD
RRDD
RDRD
RDDR

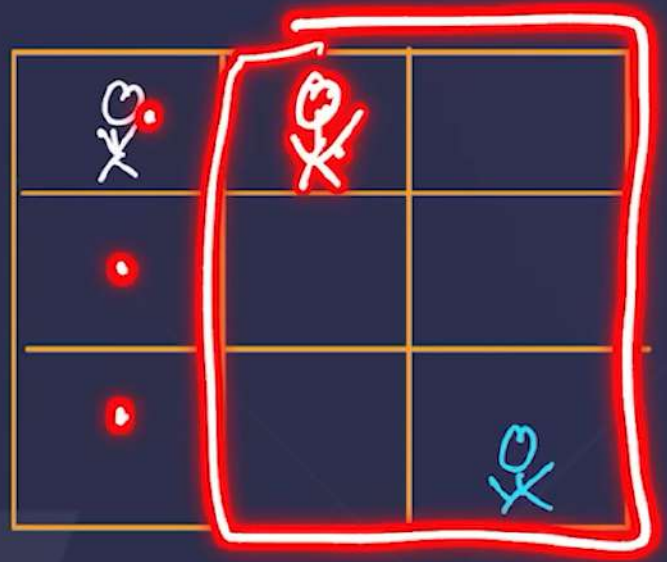


Ques : Maze path

n, m

no. of ways → 'Down, Right'

'1 step at a time'



DDRR

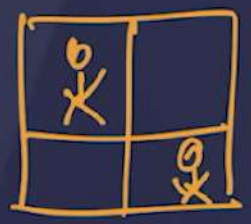
DRDR

DRRD

RRDD

RDRD

RDDR

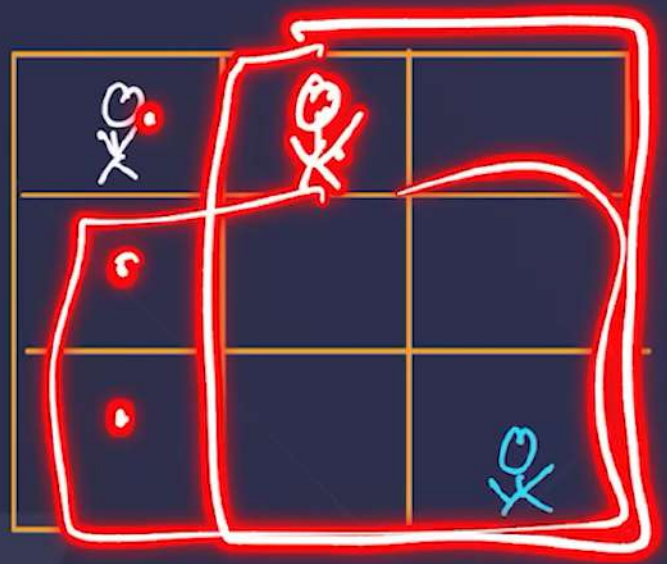


Ques : Maze path

n, m

no. of ways ⟶ 'Down, Right'

'1 step at a time'



DDRR

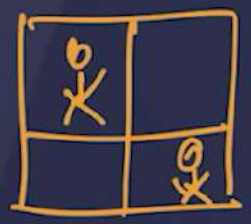
DRDR

DRRD

RRDD

RDRD

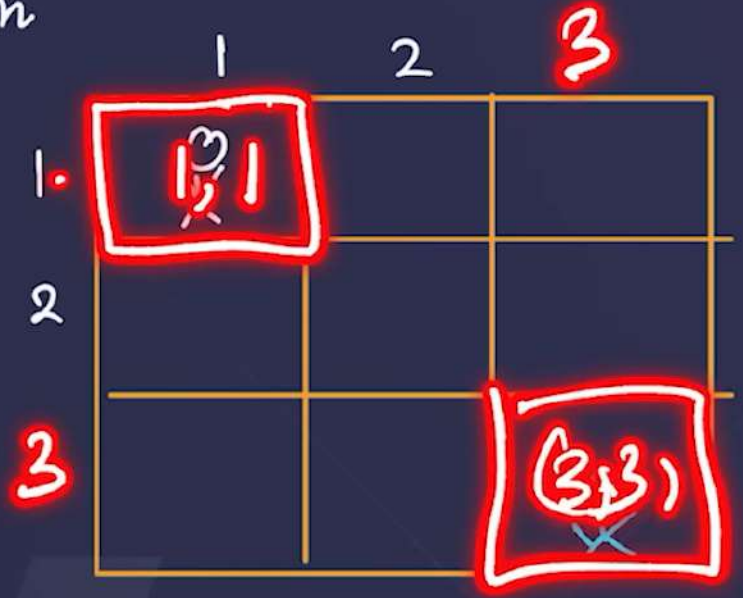
RDDR



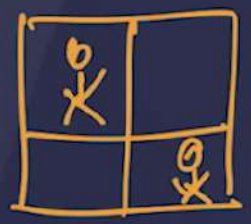
Ques : Maze path

no. of ways ⟶ 'Down, Right' '1 step at a time'

n, m



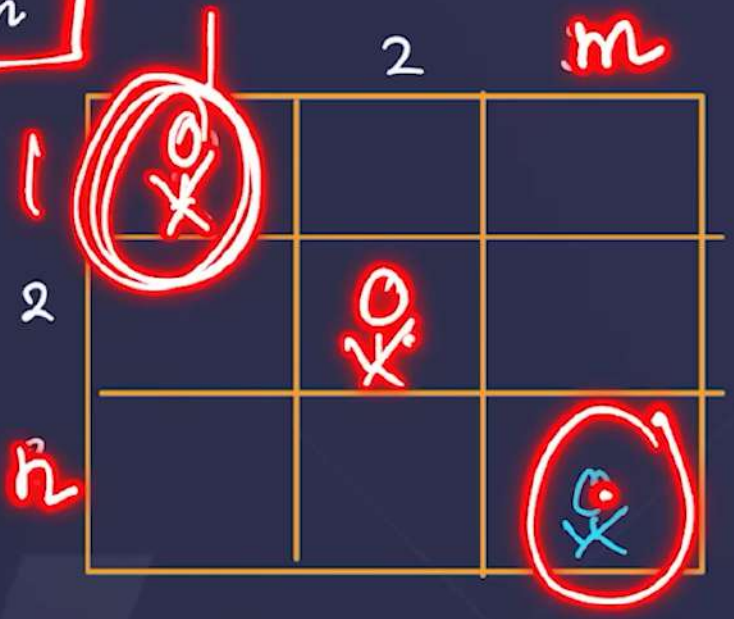
DDRR
DRDR
DRRD
RRDD
RDRD
RDDR



Ques : Maze path

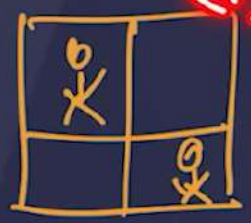
no. of ways ⟶ 'Down, Right' '1 step at a time'

n, m



DDRR
DRDR
DRRD
RRDD
RDRD
RDDR

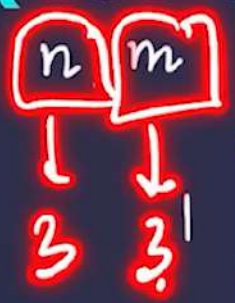
(1, 1) → (n, m)

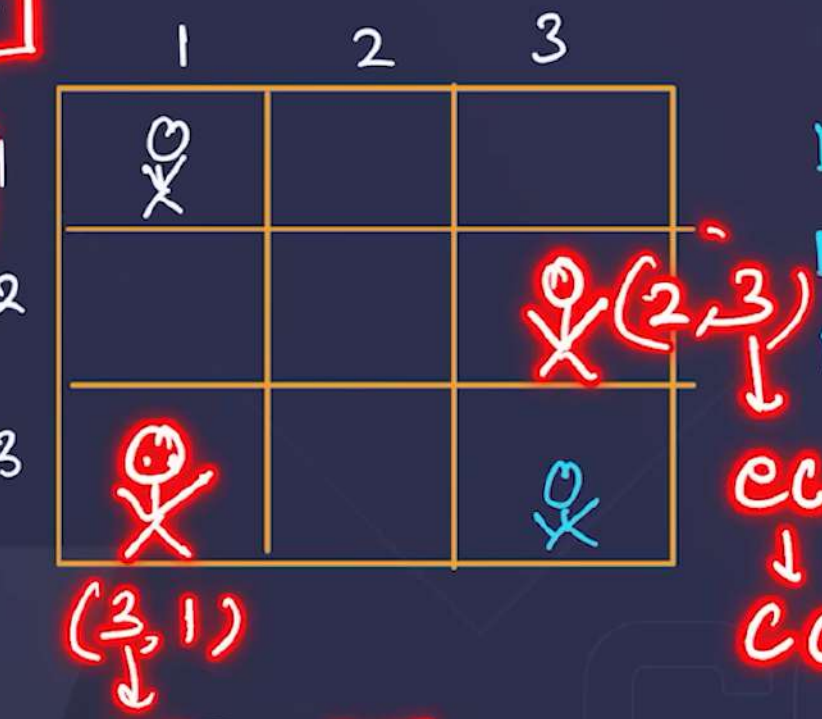


Ques : Maze path

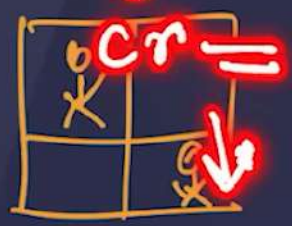
no. of ways → 'Down, Right'

'1 step at a time'





DDRR
DRDR
DRRD
cc. RRDD
RDRD
cc. RDDR

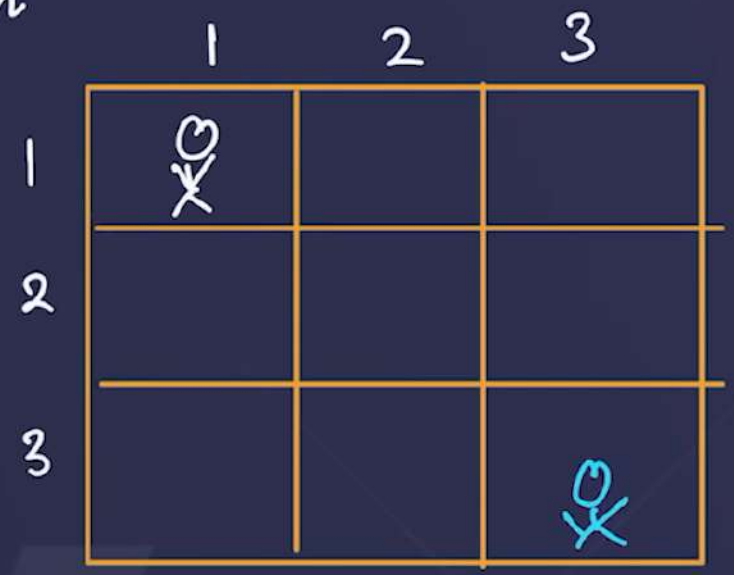


cc = ec →

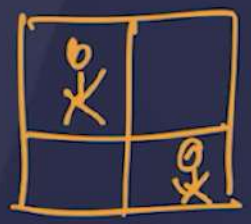
Ques : Maze path

no. of ways ⟶ 'Down, Right' '1 step at a time'

n, m



DDRR
DRDR
DRRD
RRDD
RDRD
RDDR



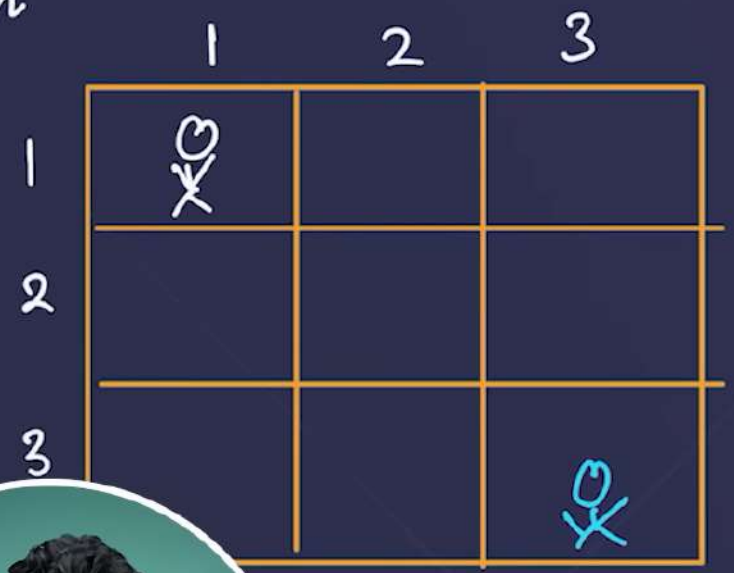
DR
RD

Ques : Maze path

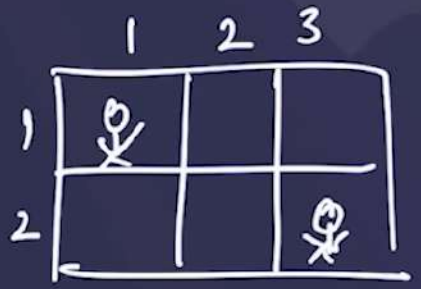
no. of ways ⟶ 'Down , Right'

'1 step at a time'

n , m

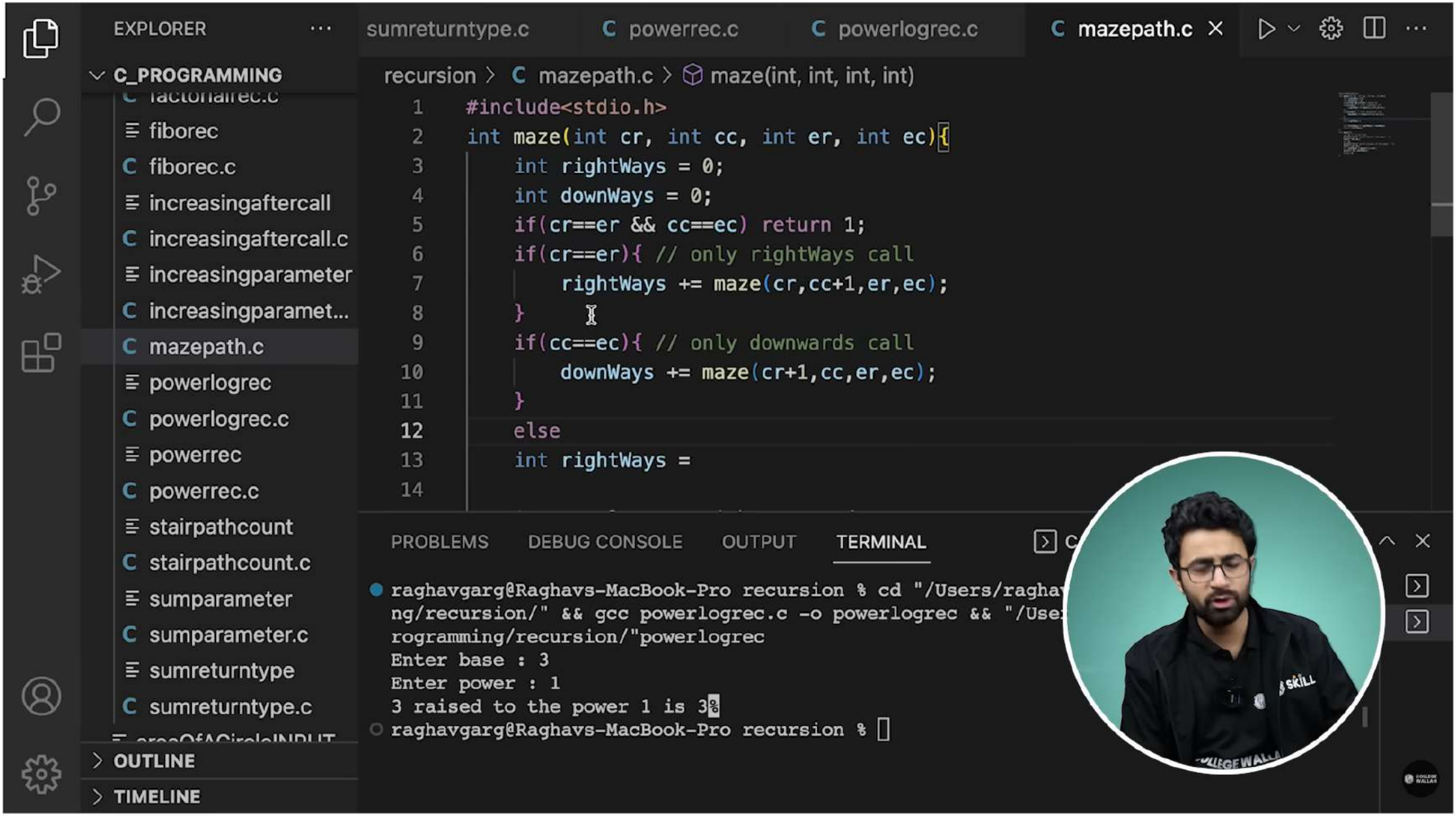


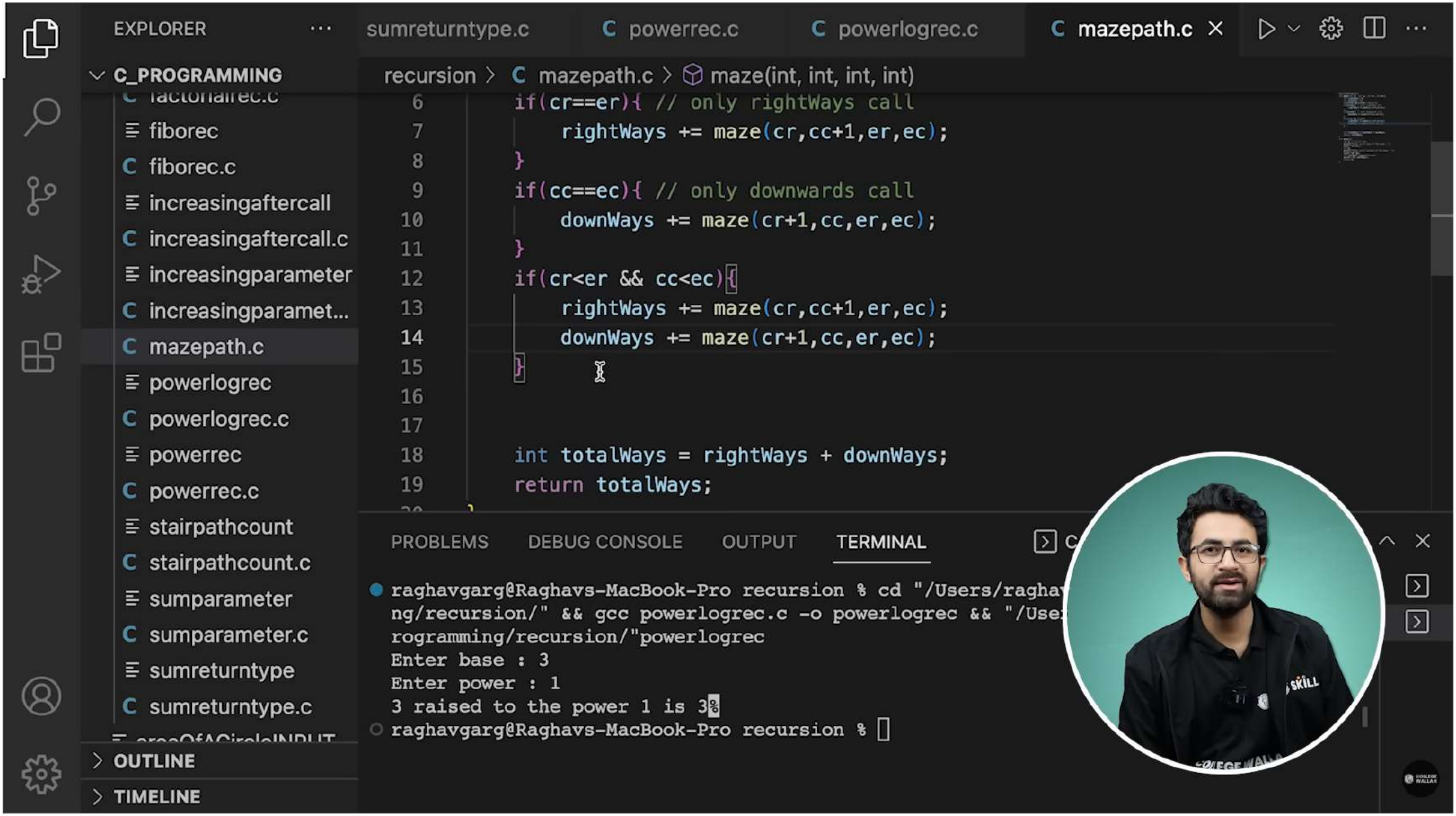
DDRR
DRDR
DRRD
RRDD
RDRD
RDDR

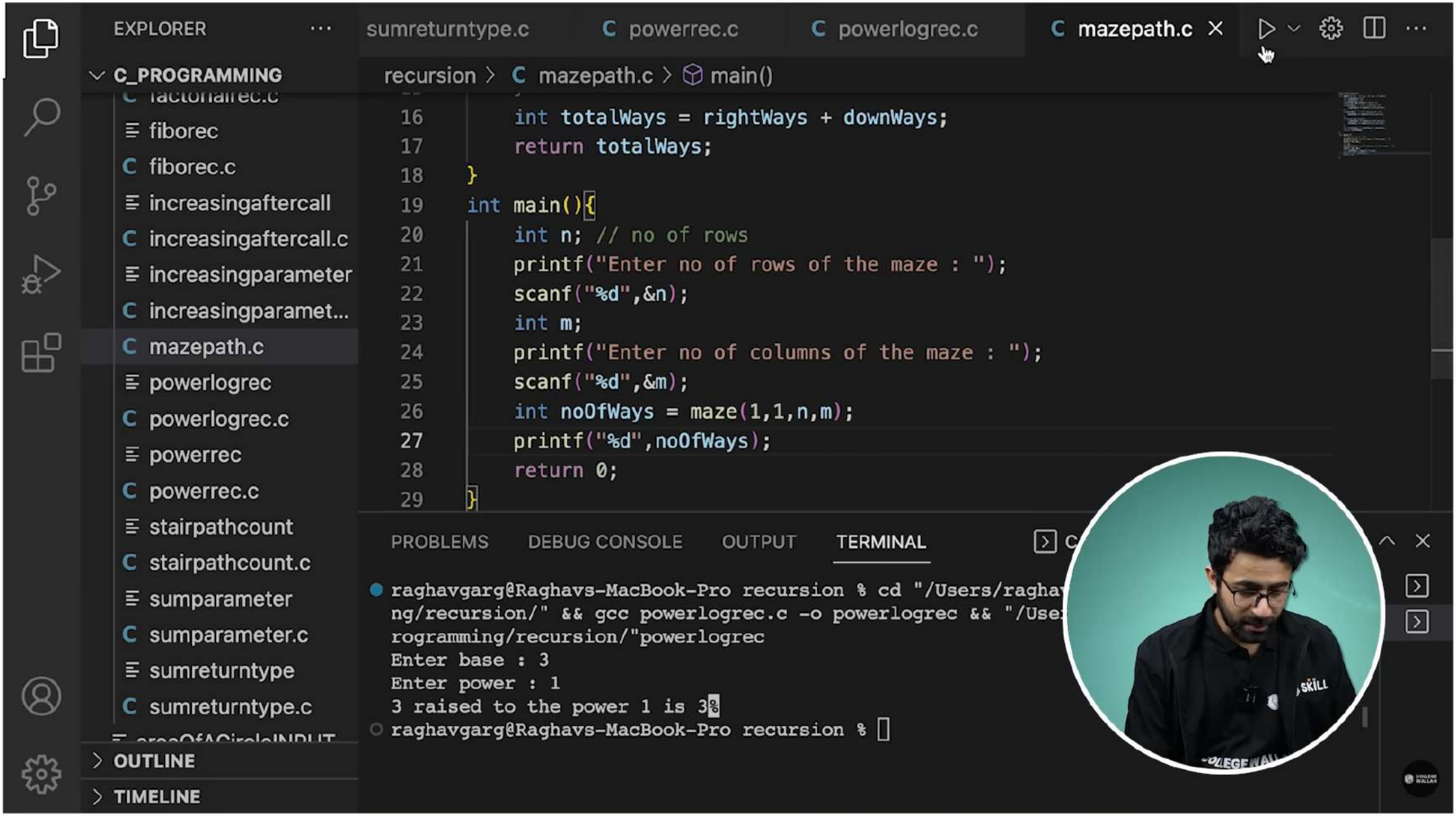


DRR
RRD
RDR

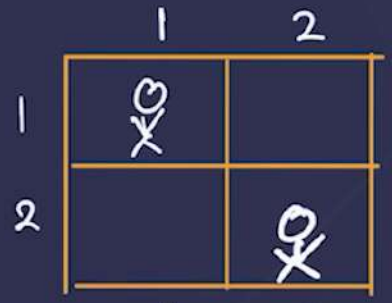
DR
RD


EXPLORER
C_PROGRAMMING
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
mazepath.c
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
powerlogrec.c
mazepath.c
recursion > mazepath.c > maze(int, int, int, int)
1 #include<stdio.h>
2 int maze(int cr, int cc, int er, int ec){
3 int rightWays = 0;
4 int downWays = 0;
5 if(cr==er && cc==ec) return 1;
6 if(cr==er){ // only rightWays call
7 rightWays += maze(cr,cc+1,er,ec);
8 }
9 if(cc==ec){ // only downwards call
10 downWays += maze(cr+1,cc,er,ec);
11 }
12 else
13 int rightWays =
14
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerlogrec.c -o powerlogrec && "/Use
rogramming/recursion/"powerlogrec
Enter base : 3
Enter power : 1
3 raised to the power 1 is 3%
raghavgarg@Raghavs-MacBook-Pro recursion %


EXPLORER
C_PROGRAMMING
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
mazepath.c
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
powerlogrec.c
mazepath.c
recursion > mazepath.c > maze(int, int, int, int)
6 if(cr==er){ // only rightWays call
7 rightWays += maze(cr,cc+1,er,ec);
8 }
9 if(cc==ec){ // only downwards call
10 downWays += maze(cr+1,cc,er,ec);
11 }
12 if(cr<er && cc<ec){
13 rightWays += maze(cr,cc+1,er,ec);
14 downWays += maze(cr+1,cc,er,ec);
15 }
16
17
18 int totalWays = rightWays + downWays;
19 return totalWays;
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerlogrec.c -o powerlogrec && "/User
rogramming/recursion/"powerlogrec
Enter base : 3
Enter power : 1
3 raised to the power 1 is 3%
raghavgarg@Raghavs-MacBook-Pro recursion %

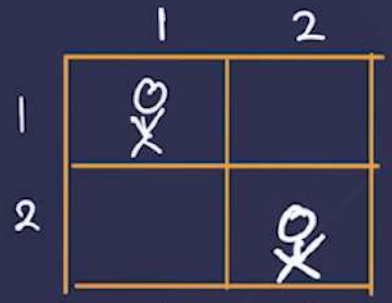

EXPLORER
C_PROGRAMMING
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
mazepath.c
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
OUTLINE
TIMELINE
sumreturntype.c
powerrec.c
powerlogrec.c
mazepath.c
recursion > mazepath.c > main()
16 int totalWays = rightWays + downWays;
17 return totalWays;
18 }
19 int main(){
20 int n; // no of rows
21 printf("Enter no of rows of the maze : ");
22 scanf("%d",&n);
23 int m;
24 printf("Enter no of columns of the maze : ");
25 scanf("%d",&m);
26 int noOfWays = maze(1,1,n,m);
27 printf("%d",noOfWays);
28 return 0;
29 }
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc powerlogrec.c -o powerlogrec && "/User
rogramming/recursion/"powerlogrec
Enter base : 3
Enter power : 1
3 raised to the power 1 is 3%
raghavgarg@Raghavs-MacBook-Pro recursion %

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```



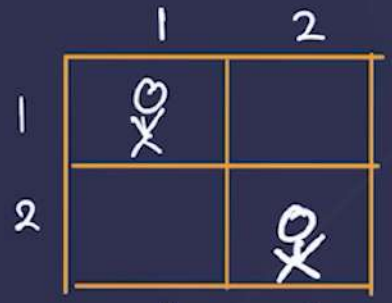
```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec)
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```



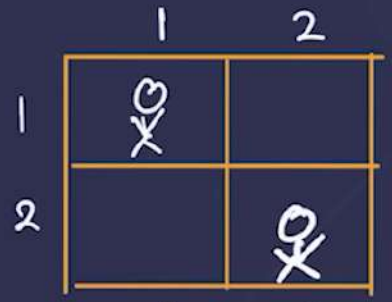

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```



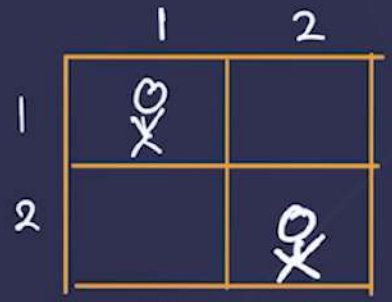

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```



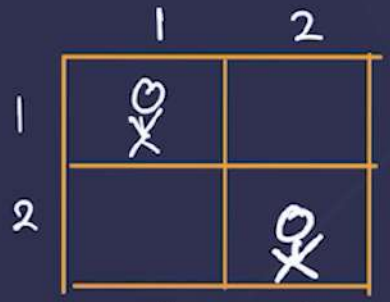

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);

        int totalWays = rightWays + downWays;
        return totalWays;
    }
}
```

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```

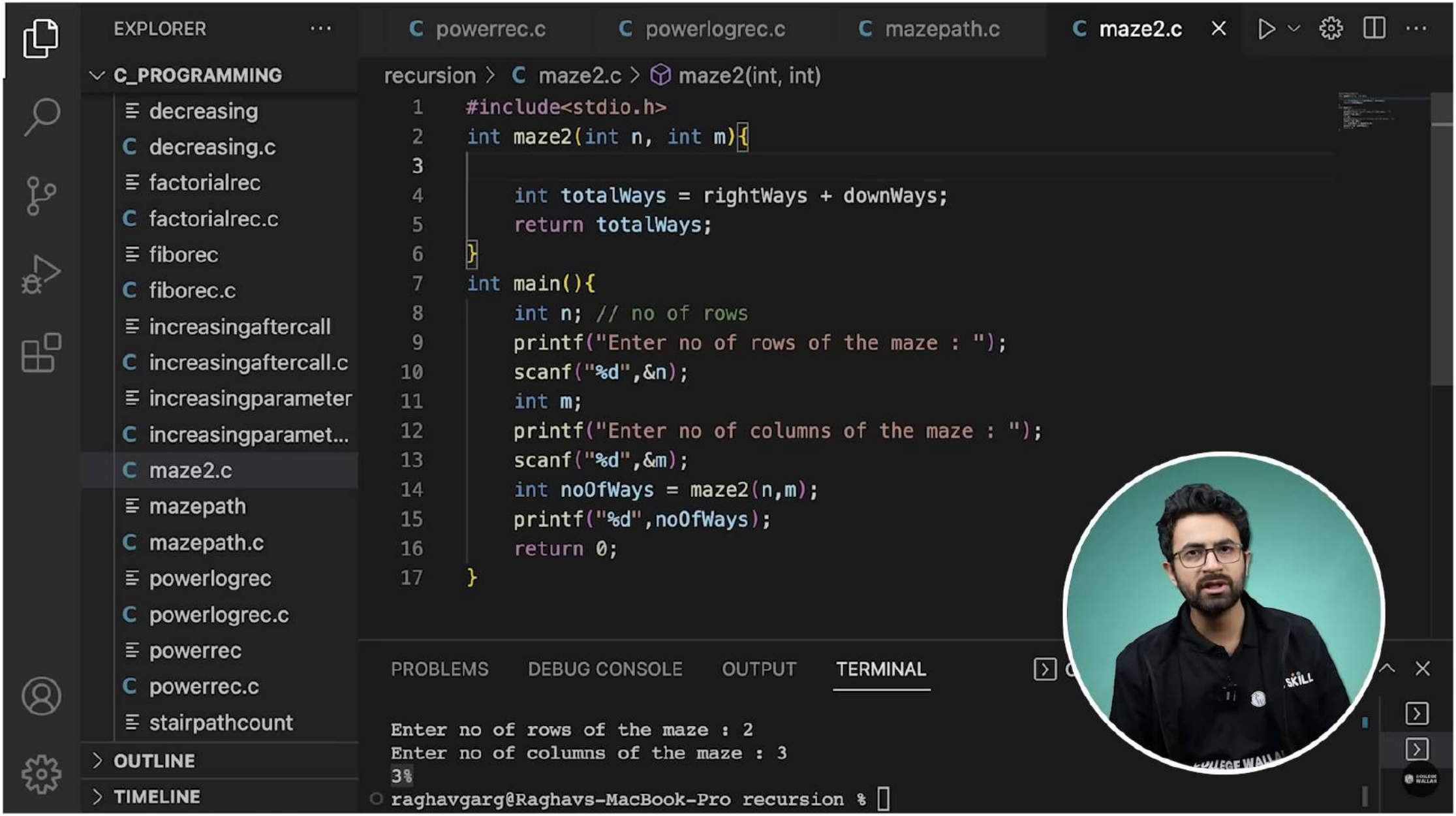


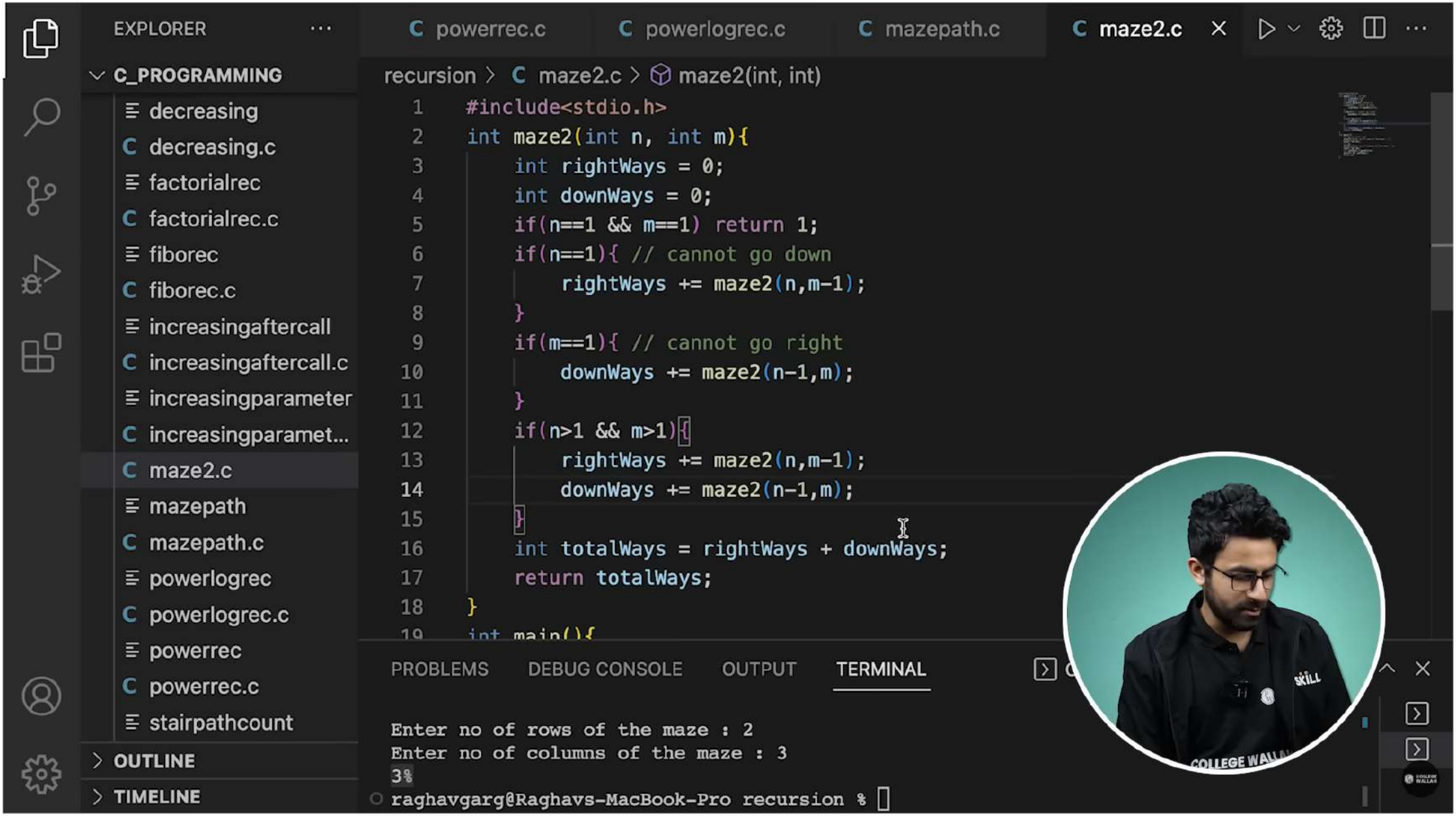

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```

```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```

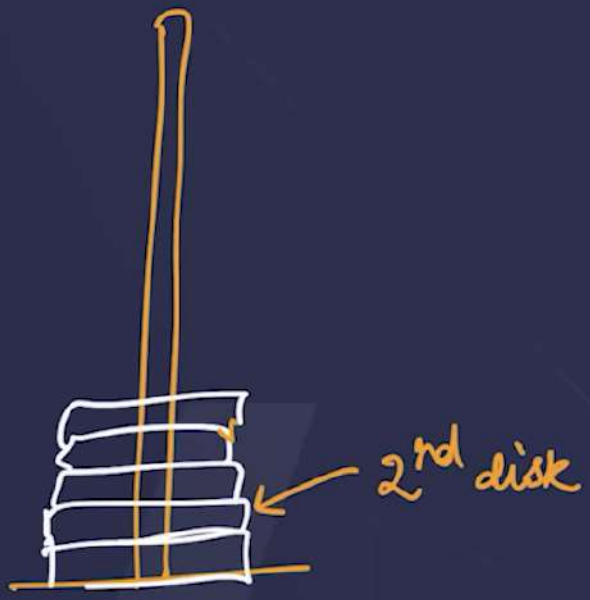


```
int maze(int cr, int cc, int er, int ec){
    int rightWays = 0;
    int downWays = 0;
    if(cr==er && cc==ec) return 1;
    if(cr==er){ // only rightWays call
        rightWays += maze(cr,cc+1,er,ec);
    }
    if(cc==ec){ // only downwards call
        downWays += maze(cr+1,cc,er,ec);
    }
    if(cr<er && cc<ec){
        rightWays += maze(cr,cc+1,er,ec);
        downWays += maze(cr+1,cc,er,ec);
    }
    int totalWays = rightWays + downWays;
    return totalWays;
}
```


EXPLORER
C_PROGRAMMING
decreasing
decreasing.c
factorialrec
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
maze2.c
mazepath
mazepath.c
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
OUTLINE
TIMELINE
powerrec.c
powerlogrec.c
mazepath.c
maze2.c
recursion > maze2.c > maze2(int, int)
#include<stdio.h>
int maze2(int n, int m){
int totalWays = rightWays + downWays;
return totalWays;
}
int main(){
int n; // no of rows
printf("Enter no of rows of the maze : ");
scanf("%d",&n);
int m;
printf("Enter no of columns of the maze : ");
scanf("%d",&m);
int noOfWays = maze2(n,m);
printf("%d",noOfWays);
return 0;
}
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
Enter no of rows of the maze : 2
Enter no of columns of the maze : 3
3%
raghavgarg@Raghavs-MacBook-Pro recursion %


EXPLORER
C_PROGRAMMING
decreasing
decreasing.c
factorialrec
factorialrec.c
fiborec
fiborec.c
increasingaftercall
increasingaftercall.c
increasingparameter
increasingparamet...
maze2.c
mazepath
mazepath.c
powerlogrec
powerlogrec.c
powerrec
powerrec.c
stairpathcount
OUTLINE
TIMELINE
powerrec.c
powerlogrec.c
mazepath.c
maze2.c
recursion > maze2.c > maze2(int, int)
#include<stdio.h>
int maze2(int n, int m){
int rightWays = 0;
int downWays = 0;
if(n==1 && m==1) return 1;
if(n==1){ // cannot go down
rightWays += maze2(n,m-1);
}
if(m==1){ // cannot go right
downWays += maze2(n-1,m);
}
if(n>1 && m>1){
rightWays += maze2(n,m-1);
downWays += maze2(n-1,m);
}
int totalWays = rightWays + downWays;
return totalWays;
}
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
Enter no of rows of the maze : 2
Enter no of columns of the maze : 3
3%
raghavgarg@Raghavs-MacBook-Pro recursion %

Call Stack


CD Rack
2nd disk


fact(1)
fact(2)
fact(3)
fact(4)
fact(5)
main()


SKILLS
COLLEGE WALLA

Pre In Post

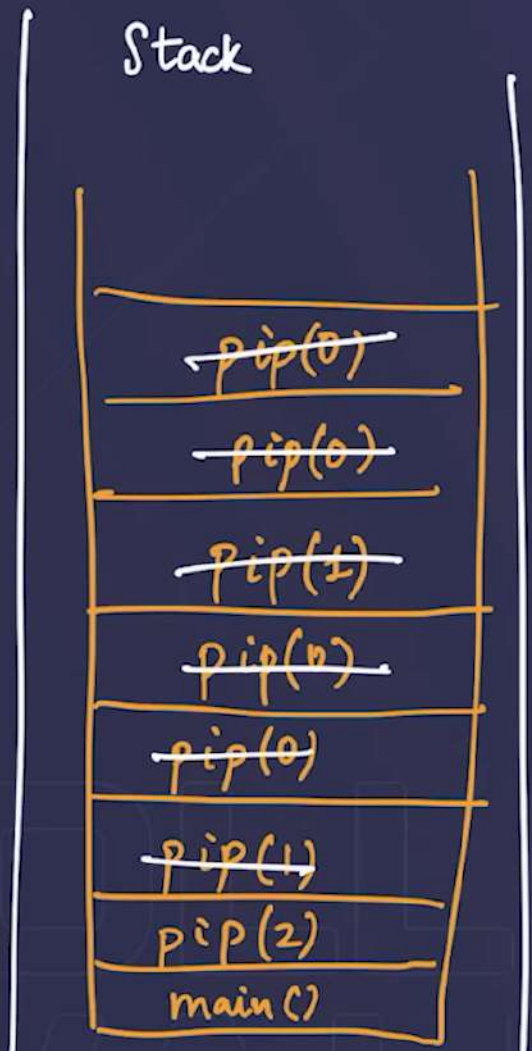
Binary tree

Stack / Queue

Pre In Post

n = 2

```
void preInPost(int n){
    if(n==0) return;
    printf("Pre %d\n",n);
    preInPost(n-1);
    printf("In %d\n",n);
    preInPost(n-1);
    printf("Post %d\n",n);
    return;
}
```



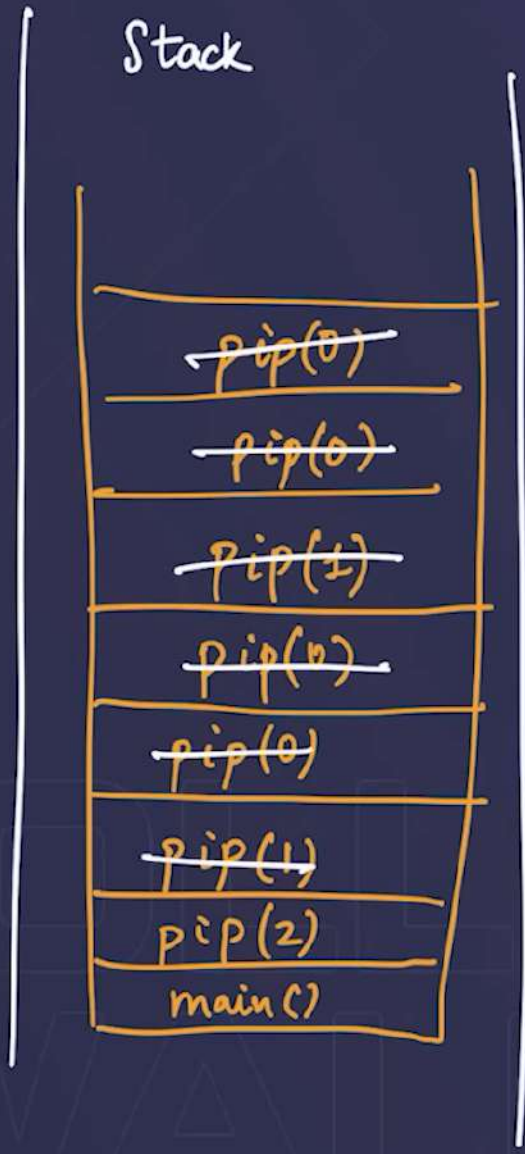
Output

- Pre 2
- Pre 1
- In 1
- Post 1
- In 2
- Pre 1
- In 1
- Post 1
- Post 2

Pre In Post

n = 2

```
void preInPost(int n){
    if(n==0) return;
    printf("Pre %d\n",n);
    preInPost(n-1);
    printf("In %d\n",n);
    preInPost(n-1);
    printf("Post %d\n",n);
    return;
}
```

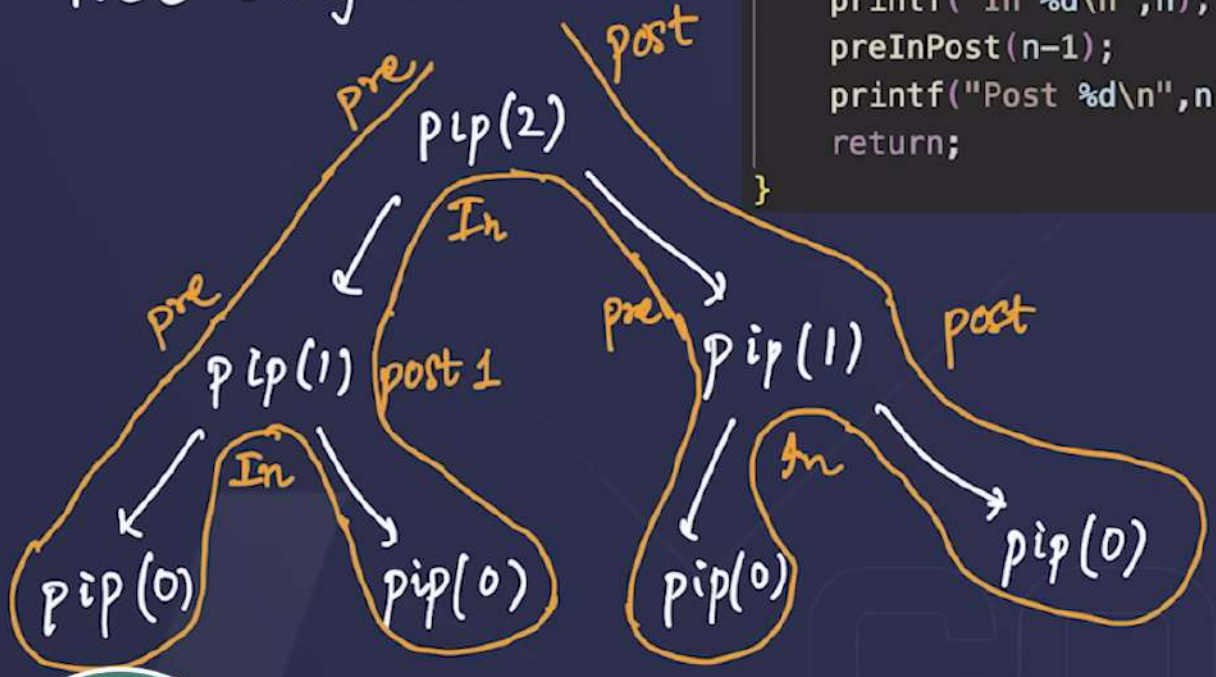


Output

- Pre 2
- Pre 1
- In 1
- Post 1
- In 2
- Pre 1
- In 1
- Post 1
- Post 2

Pre In Post

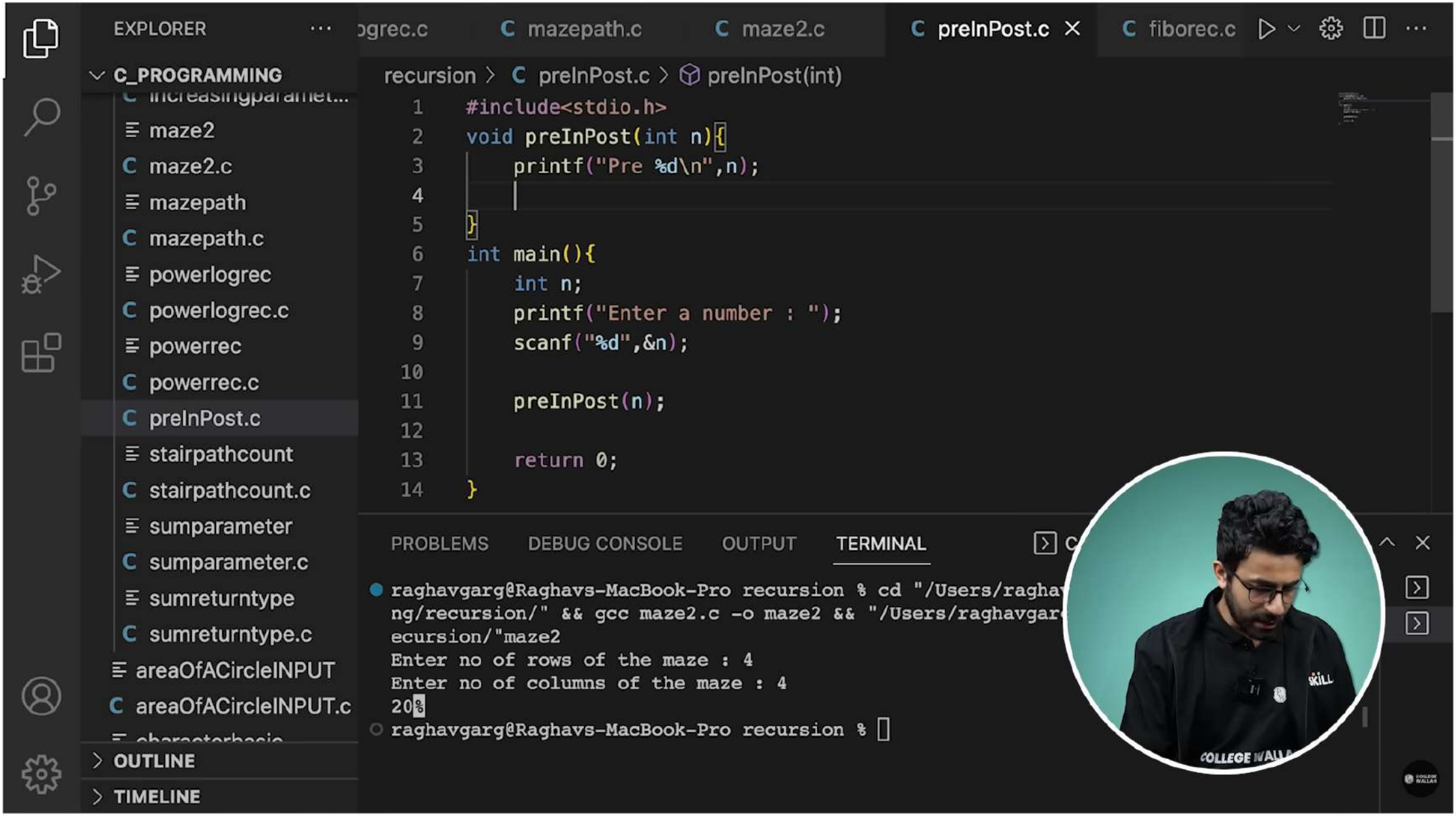
Tree Diagram

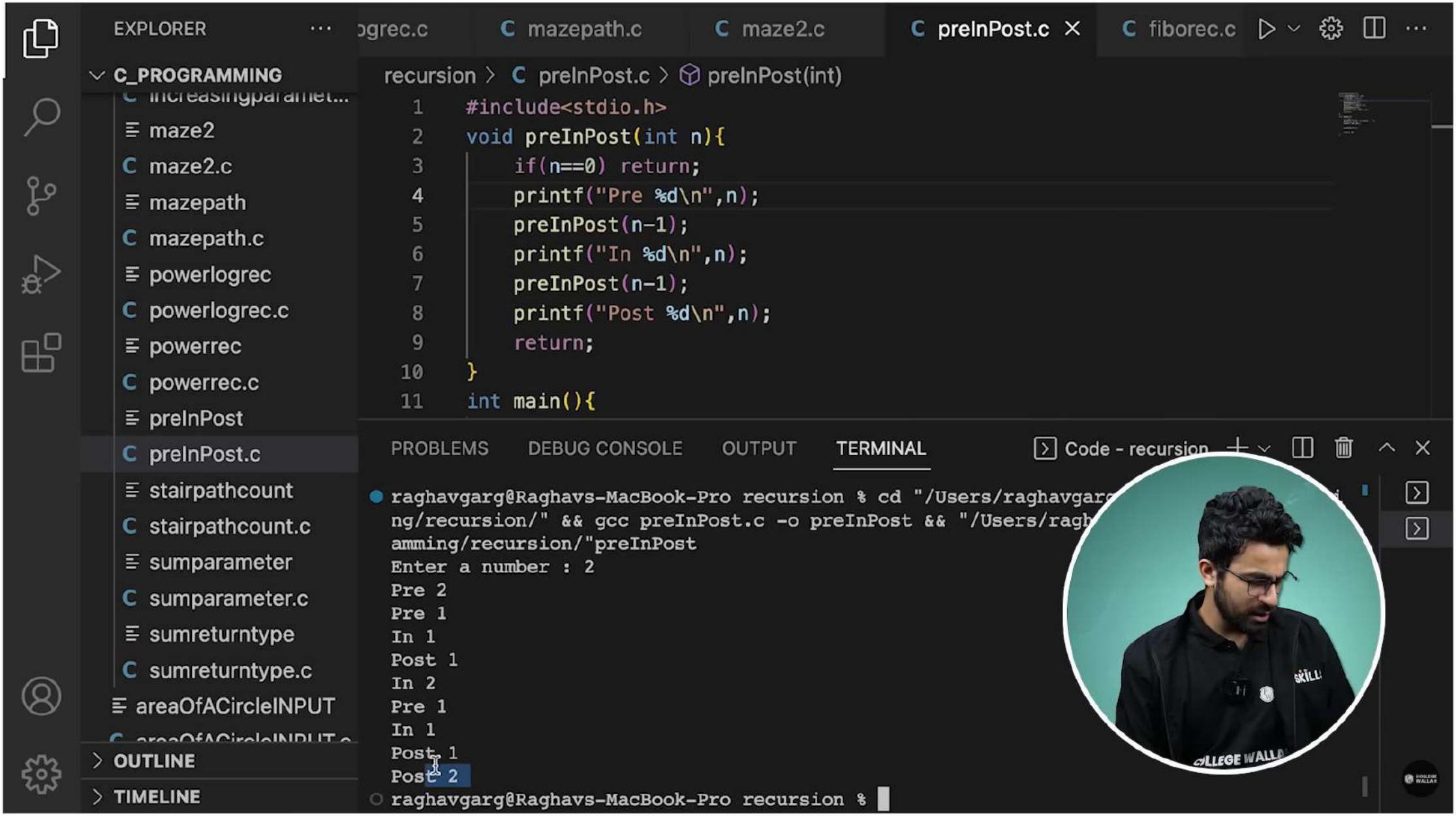



```
void preInPost(int n){
    if(n==0) return;
    printf("Pre %d\n",n);
    preInPost(n-1);
    printf("In %d\n",n);
    preInPost(n-1);
    printf("Post %d\n",n);
    return;
}
```

Output

- Pre 2
- Pre 1
- In 1
- Post 1
- In 2
- Pre 1
- In 1
- Post 1
- Post 2


EXPLORER
C_PROGRAMMING
maze2
maze2.c
mazepath
mazepath.c
powerlogrec
powerlogrec.c
powerrec
powerrec.c
preInPost.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
areaOfACircleINPUT
areaOfACircleINPUT.c
OUTLINE
TIMELINE
mazepath.c
maze2.c
preInPost.c
fiborec.c
recursion > preInPost.c > preInPost(int)
#include<stdio.h>
void preInPost(int n){
printf("Pre %d\n",n);
}
int main(){
int n;
printf("Enter a number : ");
scanf("%d",&n);
preInPost(n);
return 0;
}
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc maze2.c -o maze2 && "/Users/raghavgar
ecursion/"maze2
Enter no of rows of the maze : 4
Enter no of columns of the maze : 4
20%
raghavgarg@Raghavs-MacBook-Pro recursion %

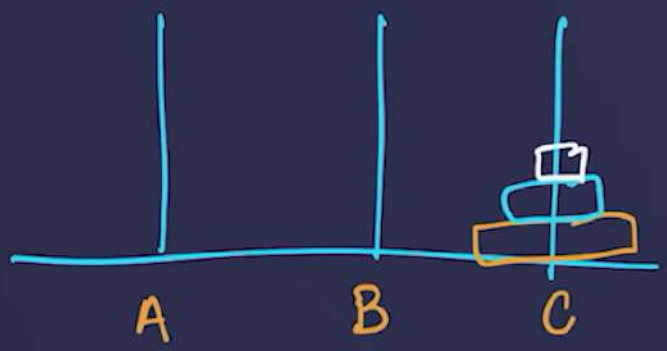

EXPLORER
C_PROGRAMMING
maze2
maze2.c
mazepath
mazepath.c
powerlogrec
powerlogrec.c
powerrec
powerrec.c
preInPost
preInPost.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
areaOfACircleINPUT
OUTLINE
TIMELINE
ogrec.c
mazepath.c
maze2.c
preInPost.c
fiborec.c
recursion > preInPost.c > preInPost(int)
#include<stdio.h>
void preInPost(int n){
if(n==0) return;
printf("Pre %d\n",n);
preInPost(n-1);
printf("In %d\n",n);
preInPost(n-1);
printf("Post %d\n",n);
return;
}
int main(){
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
Code - recursion
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghavgarg
ng/recursion/" && gcc preInPost.c -o preInPost && "/Users/ragh
amming/recursion/"preInPost
Enter a number : 2
Pre 2
Pre 1
In 1
Post 1
In 2
Pre 1
In 1
Post 1
Post 2
raghavgarg@Raghavs-MacBook-Pro recursion %

Ques : Print zig-zag

Input	Output
1	111
2	211121112
3	32111211123211121123
4	432111211123211121112343211121112321112111234

Ques : Tower of HANOI

"Last"



Biggest disk to the 3rd rod

Steps

A → C

A → B

C → B

A → C

B → A

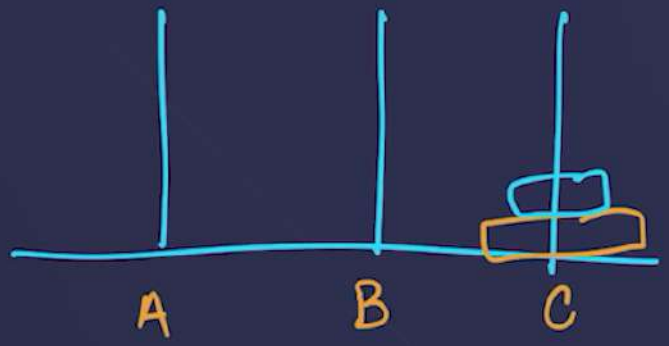
B → C

A → C

Input → n → n. of disks

disks	min moves
3 →	$2^3 - 1 = 7$
4 →	$2^4 - 1 = 15$
5 →	$2^5 - 1 = 31$

Ques : Tower of HANOI



A → B

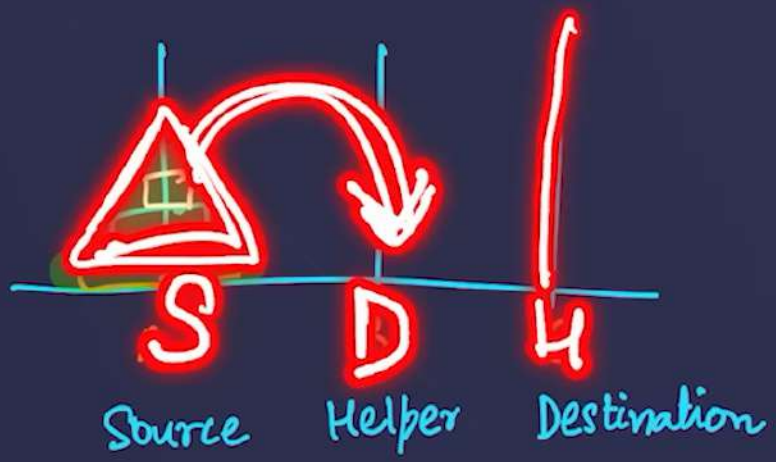
A → C

B → C

n = 2

m. moves → $2^2 - 1 = 3$

Ques : Tower of HANOI



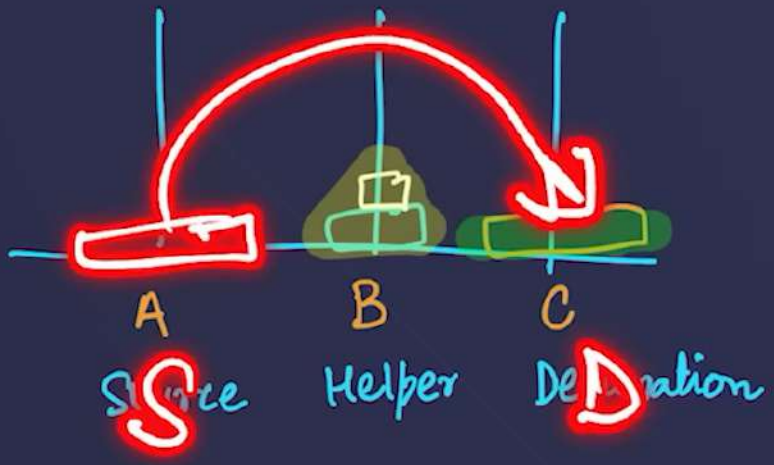
n-1 disks

S → H Small pyramid // call

S → D largest disk

H → D Small pyramid // call

Ques : Tower of HANOI



S → H

S → D

H → D

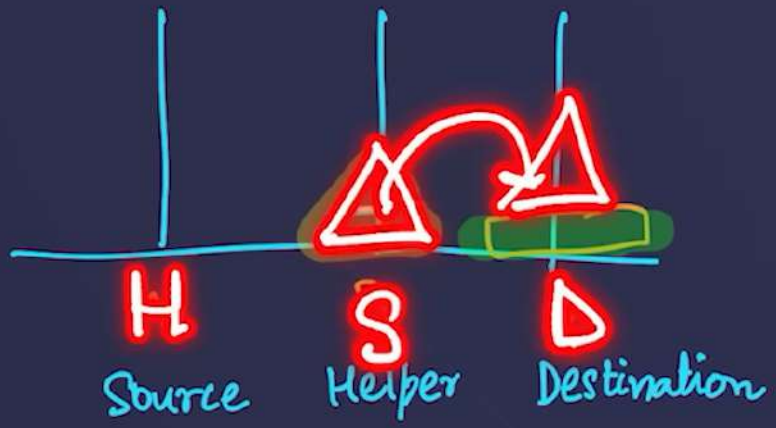
n-1 disks

Small pyramid //call

largest disk

Small pyramid //call

Ques : Tower of HANOI



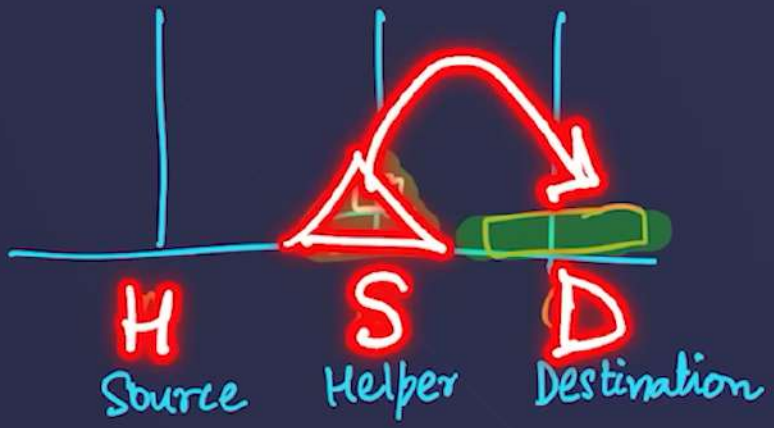
n-1 disks

S → H Small pyramid // call

S → D largest disk

H → D Small pyramid // call

Ques : Tower of HANOI

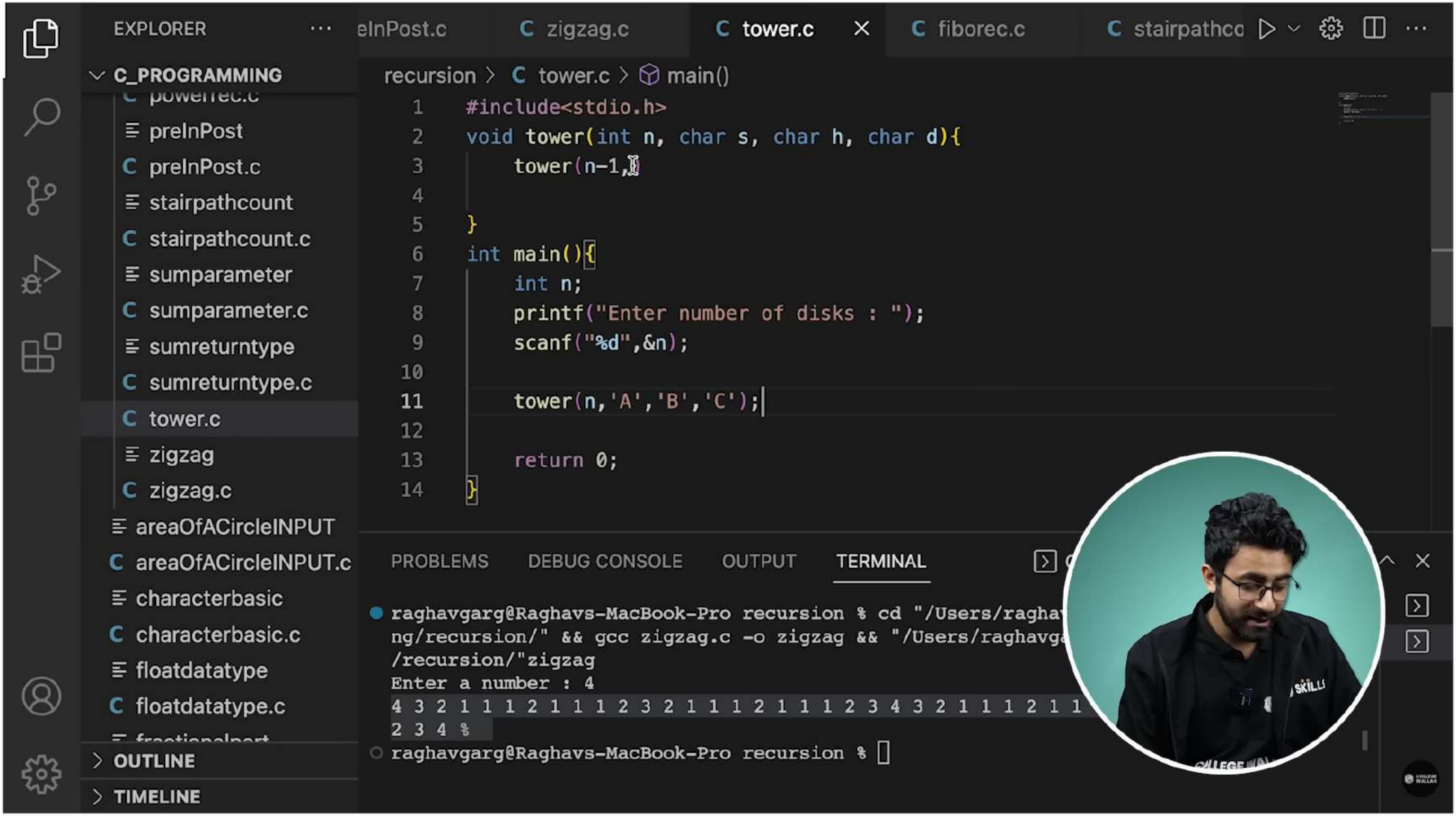


n-1 disks

$S \rightarrow H$ Small pyramid // call

$S \rightarrow D$ largest disk

$H \rightarrow D$ Small pyramid // call


EXPLORER
C_PROGRAMMING
preInPost
preInPost.c
stairpathcount
stairpathcount.c
sumparameter
sumparameter.c
sumreturntype
sumreturntype.c
tower.c
zigzag
zigzag.c
areaOfACircleINPUT
areaOfACircleINPUT.c
characterbasic
characterbasic.c
floatdatatype
floatdatatype.c
OUTLINE
TIMELINE
zigzag.c
tower.c
fiborec.c
recursion > tower.c > main()
#include<stdio.h>
void tower(int n, char s, char h, char d){
tower(n-1,
}
int main(){
int n;
printf("Enter number of disks : ");
scanf("%d",&n);
tower(n,'A','B','C');
return 0;
}
PROBLEMS
DEBUG CONSOLE
OUTPUT
TERMINAL
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghav
ng/recursion/" && gcc zigzag.c -o zigzag && "/Users/raghavga
/recursion/"zigzag
Enter a number : 4
4 3 2 1 1 1 2 1 1 1 2 3 2 1 1 1 2 1 1 1 2 3 4 3 2 1 1 1 2 1 1
2 3 4 %
raghavgarg@Raghavs-MacBook-Pro recursion %

EXPLORER

C_PROGRAMMING

- preInPost
- preInPost.c
- stairpathcount
- stairpathcount.c
- sumparameter
- sumparameter.c
- sumreturntype
- sumreturntype.c
- tower
- tower.c
- zigzag
- zigzag.c
- areaOfACircleINPUT
- areaOfACircleINPUT.c
- characterbasic
- characterbasic.c
- floatdatatype

OUTLINE

TIMELINE

eInPost.c | zigzag.c | tower.c | fiborec.c | stairpathco

recursion > tower.c > tower(int, char, char, char)

```c
#include<stdio.h>
void tower(int n, char s, char h, char d){
    if(n==0) return;
    tower(n-1,s,d,h);
    printf("%c -> %c\n",s,d);
    tower(n-1,h,s,d);
    return;
}
```

PROBLEMS | DEBUG CONSOLE | OUTPUT | TERMINAL

Code - recursion

```
raghavgarg@Raghavs-MacBook-Pro recursion % cd "/Users/raghavgarg/Desktop/C_Programming/recursion/" && gcc tower.c -o tower && "/Users/raghavgarg/Desktop/C_Programming/recursion/"tower
Enter number of disks : 3
A -> C
A -> B
C -> B
A -> C
B -> A
B -> C
A -> C
raghavgarg@Raghavs-MacBook-Pro recursion %
```