Recursion

(1) Sum of first N numbers

the first n numbers
now this can be done using
a very

TUF

Re 3. Parameterised and Functional Recursion | Strivers A2Z DSA Course

Recursion

(1) Sum of first N numbers

very strong i'm going to teach you a
couple of ways the first way being

TUF

1:32 / 18:13 • Problem Description

Recursion

(1) Sum of first N numbers

parameter

functional

returns
the answer

TUF

Recursion

(1) Sum of first N numbers

parameter

functional

i hope you have understood the question
sum of first

TUF

Recursion

(1) Sum of first N numbers

parameter

functional

N = 3

6

1+2+3 = 5

definitely 6 because
1 plus 2 plus 3 is

TUF

parameterised

f(i, sum)
{

and i carry a variable sum
okay and that's the recursive function

TUF

parameterised

```
f(i, sum)
{
  if (i < 1)
  {
    print (sum)
    return;
  }
  f(i-1, sum + i)
}
```

sum
plus i is what i'm going to pass

TUF

parameterised

```
f(i, sum)
{
  if (i < 1)
  {
    print (sum)
    return;
  }
  f(i-1, sum + i)
}
```

plus i is what i'm going to pass
so this is what my recursive statement
will

TUF

```
f(i, sum)
{
  if (i < 1)
  {
    print (sum)
    return;
  }
  f(i-1, sum+i)
}

main()
{
  n
  f(
```

the input
and then we can just call f of

TUF

Re 3. Parameterised and Functional Recursion | Strivers A2Z DSA Course

```
f(i, sum)
{
  if (i < 1)
  {
    print (sum)
    return;
  }
  f(i-1, sum + i)
}

main()
{
  n
  f(n, 0)
}
```

3:29 / 18:13 • Parameterised Way

```
f(i, sum)
{
    if (i < 1)
    {
        print (sum)
        return;
    }
    f(i-1, sum + i)
}

main()
{
    n
    f(n, 0)
}
```

TUF

```
{
  if (i < 1)
  {
    print (sum)
    return;
  }
  f(i-1, sum + i)
}

main()
{
  n ← 3
  f(n, 0)
}
```

Output

TUF

parameterised

```
f(i, sum)
{
    if (i < 1)
    {
        print (sum)
        return;
    }
    f(i-1, sum + i)
}

main()
{
    n
    f(n, 0)
```

3 0

3

f(

Output

TUF

) x

t (sum)

hm;

o + 3

um + i)

y(2, 3)

{

y( ) x

y(i-1, 3+2)

}

Output

TUF

memorised

```
f(3, 0)
{
  if (i < 1)
  {
    print (sum)
    return;
  }
  f(i-1, sum + i)
}
```

0 + 3

```
f(2, 3)
{
  if (  )
  f(i-1, 3+2)
}
```

```
f(1, 5)
{
  if (
```

```
main()
{
  n
  f(n, 0)
```

3

Output

$y(2, 3)$

$\{$

$y(\ )\ \times$

$y(i-1, 3+2)$

$\}$

$y(1, 5)$

$\{$

$y(\ )\ \times$

$y(i-1, 5$

Output

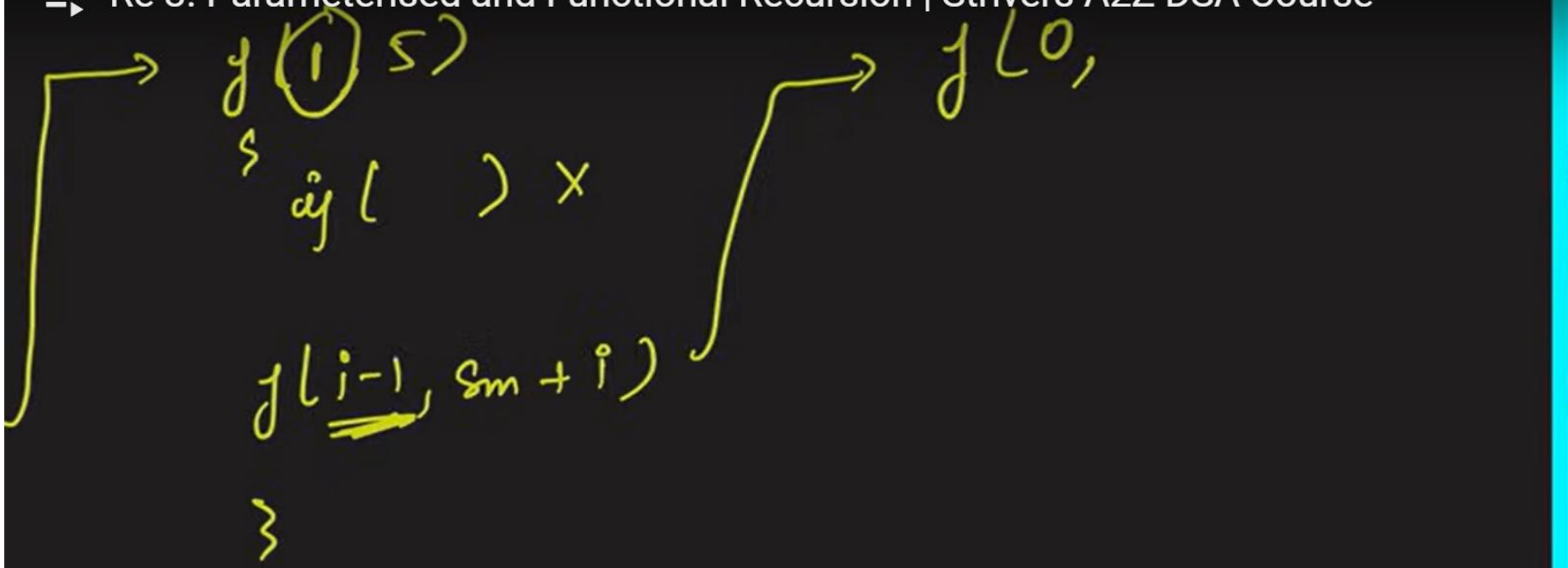
TUF

```
f(2, 3)
{
  if( ) x

  f(i-1, 3+2)
}

f(1, 5)
{
  if( ) x

  f(i-1, sm + i)
}
```

Output

TUF

Re 3. Parameterised and Functional Recursion | Strivers A2Z DSA Course



5:26 / 18:13 • Parameterised Way

$f(1, 5)$

$\{ \; ij(\quad) \times$

$f(i-1, \overset{5+1}{\cancel{5m}} + i)$

$\}$

$f(0, 6)$

$\{ \; ij$

TUF

```
f(1, 5)
{
  if ( )  ×

  f(i-1, sum + i)
}
```

```
f(0, 6)
{
  if (i < 1) ✓
    print(sum)
```

TUF

3)

) x

1, 3+2)

f(1, 5)

{

if ( ) x

f(i-1, 5+1 + i)

}

f(0, 6)

{

if (i < 1)

print(

subn

f( ) x

}

TUF

memoised

f(3, 0)

{

if (i < 1) ×

{

print (sum)

return;

}

f(i-1, sum + i) 0 + 3

}

main()

{

n ← 3

f(n, 0)

f(2, 3)

{

if ( ) ×

f(i-1, 3+2)

}

f(1, 5)

{

if (

f(i-1,

}

Output

6

memorised

f(3, 0)

{

if (i < 1) ×

{

print (sum)

return;

}

f(i-1, sum + i) 0 + 3

}

f(2, 3)

{

if ( ) ×

f(i-1, 3 + 2)

}

f(1, 5)

{

if (

f(i-1,

}

main()

{

n ← 3

f(n, 0)

Output

6

TUF

$f(2, 3)$

$f(1, 5)$

$f(0, 6)$

TUF

$f(n, 0)$

}

; sm

$f(3, 0)$

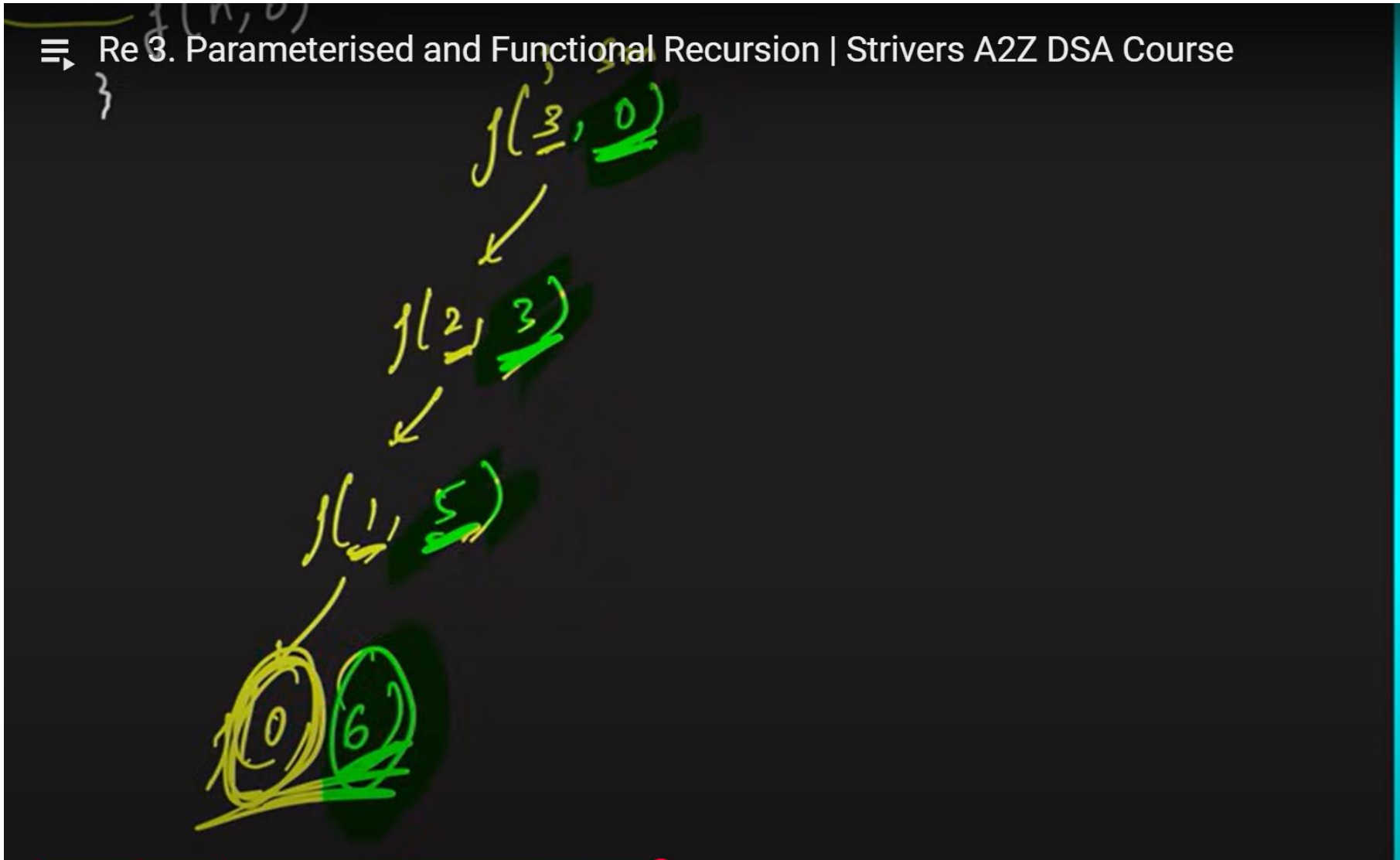
$f(2, 3)$

$f(1, 5)$

0 6

TUF

Re 3. Parameterised and Functional Recursion | Strivers A2Z DSA Course



7:03 / 18:13 • Parameterised Way

functional

Announcing the FREE ka TREE SERIES #shorts

TUF

functional

TUF

functional

TUF

functional

TUF

functional

$n = 3$

$3 + f(2)$

$f(n) \rightarrow$ Sm 0

TUF

functional

$n = 3$

$3 + f(2)$

$2 + f(1)$

$1 + f(0)$

0

$f(n) \rightarrow$ sum of first N nos

TUF

Functional

$n = 3$

$3 + f(2)$

$2 + f(1)$

$1 + f(0)$

0

$f(n) \rightarrow$ Sum of first N nos

$f(3) \rightarrow 3 + f(n$

TUF

functional

$n = 3$

$3 + f(2)$

$2 + f(1)$

$1 + f(0)$ 0

$f(n) \rightarrow$ sum of first N nos

$f(3) \rightarrow 3 + f(2)$

TUF

```
f(n)
{
    if (n==0)
        return 0;

    return
```

TUF

```
f(n)
{
    if (n==0)
        return 0;

    return n + f(n-1);
}
main()
{
    r
```

TUF

```
f(n)
{
    if (n==0)
        return 0;

    return n + f(n-1);
}
main()
{
    n ← 3
print(f(n));
}
```

TUF

```
f(3)
{
    if (n==0)
        return 0;

    return 3 + f(n-1);
}

main()
{
    n ← 3
    print(f(n));
}
```

TUF

```
f(2)
{
  if(n==0)
  return 2+f(1)
}
```

f(1)

TUF

```
f(2)
{
  if (n==0) ✗
  return 2+f(1)
}
```

```
f(1)
{
  if (n==0) ✗
  return 0
  return 1+f(0)
}
```

TUF

```
f(1)
{ if (n==0) X
    return 0

return 1 + f(0)

}
```

TUF

Re 3. Parameterised and Functional Recursion | Strivers A2Z DSA Course

```
f(1)
{
if (n==0)
return 0

return 1 + f(0)
}
```

```
f(0)
{
if (n==0)
return 0
```

11:59 / 18:13 • Functional Recursion

```
f(1)
{
if (n==0) ×
return 0

return 1 + f(0)
}
```

```
f(0)
{
if (n==0)
return 0
```

TUF

f(1)

{

if (n==0) ×

return 0

return 1 + 0

}

0

f(0)

{

if (n==0)

return 0

TUF

Let's give back to the community :)

f(2)
{
if (n==0) ×
return 2+1
}

1

f(1)
{
if (n==0) ×
return 0
return 1+0
}

0

f(0)
{
if (

TUF

Let's give back to the community :)

```
f(3)
{
  if (n==0)
    return 0;

  return 3 +
}

main()
{
  n ← 3
  f(n);
```

```
f(2)
{
  if(n==0)
  return 2+1
}
```

```
f(1)
{
  if
  return
}
```

TUF

```
f(3)
{
    if (n==0)
        return 0;
    return 3 + 3
}

main()
{
    n = 3
    cm(f(n));
}
```

```
f(2)
{
    if(n==0)
        return 2+1
}
```

3

TUF

```
f(3)
{
  if (n==0)
    return 0;
  return 3 + 3
}

main()
{
  n = 3
  cout(f(n));
}
```

6

```
f(2)
{
  if (n==0)
  return 2+1
}
```

3

Output

6

TUF

```
f(3)
{
    if (n==0)
        return 0;
    return 3 + 3
}

main()
{
    n = 3
    cout(f(n));
}
```

6

```
f(2)
{
    if(n==0)
    return 2+1
}
```

3

1

Output

6

```
f(3)
{
    if (n==0)
        return 0;
    return 3 + 3
}

main()
{
    n = 3
    cm(f(n));
}
```

```
f(2)
{
    if(n==0)
    return 2+1
}
```

3

1

6

Output

6

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;

int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif



    return 0;
}
```

input.txt

```
1
```

output.txt

```
0
1
2
```

```
1[Finished in 1.6s]
```

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int sum(int n)
{}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif



    return 0;
}
```

input.txt

```
1
```

output.txt

```
0
1
2
```

```
1[Finished in 1.6s]
```

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int sum(int n)
{
    if( n == 0) return 0;
    return n + sum(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << sum(n);


    return 0;
}
```

input.txt

```
1
```

output.txt

```
0
1
2
```

TUF

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int sum(int n)
{
    if( n == 0) return 0;
    return n + sum(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << sum(n);


    return 0;
}
```

input.txt

```
1
```

output.txt

```
6
```

```
[Finished in 1.3s]
```

Re 3. Parameterised and Functional Recursion | Strivers A2Z DSA Course

Task

factorial of N

N=3 → 6

1×2×3

N=4 → 24

1×2×3×4

14:14 / 18:13 • Code ›

```
f(3)
{
  if (n==0)
    return 0;
  return 3 + 3
}

main()
{
  n = 3
  cout(f(n));
}
```

6

```
f(2)
{
  if (n==0)
  return 2+1
}
```

3

1

Output

6

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int fact(int n)
{
    if( n == 0) return 0;
    return n + sum(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << sum(n);


    return 0;
}
```

input.txt

```
1
```

output.txt

```
6
```

```
[Finished in 1.3s]
```

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int fact(int n)
{
    if( n == 0) return 0;
    return n * sum(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << sum(n);


    return 0;
}
```

input.txt

```
1
```

output.txt

```
6
```

```
[Finished in 1.3s]
```

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int fact(int n)
{
    if( n == 0) return 0;
    return n * sum(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << fact(n);


    return 0;
}
```

input.txt

```
1
```

output.txt

```
6
```

```
[Finished in 1.3s]
```

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int fact(int n)
{
    if( n == 0) return 0;
    return n * sum(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << fact(n);
    return 0;
}
```

input.txt

```
1
```

output.txt

```
6
```

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int fact(int n)
{
    if( n == 0) return 0;
    return n * fact(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << fact(n);
    return 0;
}
```

error: use of undeclared identifier 'sum'

input.txt

```
1
```

output.txt

```
6
```

```
1 error generated.
[Finished in 0.7s with exit code 1]
[cmd: ['bash', '-c', "g++ -std=c++14 '/Users/striver/Desktop/SublimeFolder/code.cpp' -o '/Users/striver/Desktop/Subl
code' && '/Users/striver/Desktop/SublimeFolder/code'"]]
[dir: /Users/striver/Desktop/SublimeFolder]
```

TUF

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int fact(int n)
{
    if( n == 0) return 0;
    return n * fact(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << fact(n);
    return 0;
}
```

input.txt

```
1
```

output.txt

```
0
```

[Finished in 1.2s]

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int fact(int n)
{
    if( n == 0) return 1;
    return n * fact(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << fact(n);
    return 0;
}
```

input.txt

```
1
```

output.txt

```
0
```

[Finished in 0.8s]

TUF

```
f(n)
{
    if (n==0)
        return 0;
    return n x f(n-1)
}
```

TUF

```
f(n)
{
  if (n==0)
      return 0;
  return n x f(n-1)
}

main ()
{
  n ← 3
  print(f(n))
}
```

```
f(3)
{
    if (n==0)
        return 0;
    return n x f(n-1)
}

main ()
{
    n = 3
    print(f(n))
}
```

```
f(3)
{
    if (n==0)
        return 0;
    return 3 x f(n-1)
}

main()
{
    n = 3
    print(f(n))
}
```

```
f(2)
{
    if()
    return 2 x f(2-1)
```

TUF

n==0) x

return 0;

3 x f(n-1)

f(2)

{

if() x

return 2 x f(2-1);

}

f(

)

3

(f(n))

TUF

```
f(2)
{
    if() x

    ret 2 x f(2-1);

}
```

```
f(1)
{
    if (n==0) x

    ret 1 x
```

2)

ej() x

2 x f(2-1);

f(1)

{

if (n==0) x

ret 1 x f(0)

}

TUF

2)

ej() ×

2 × f(2-1);

f(1)

{

if (n==0) ×

return 1 × f(0)

}

f(0)

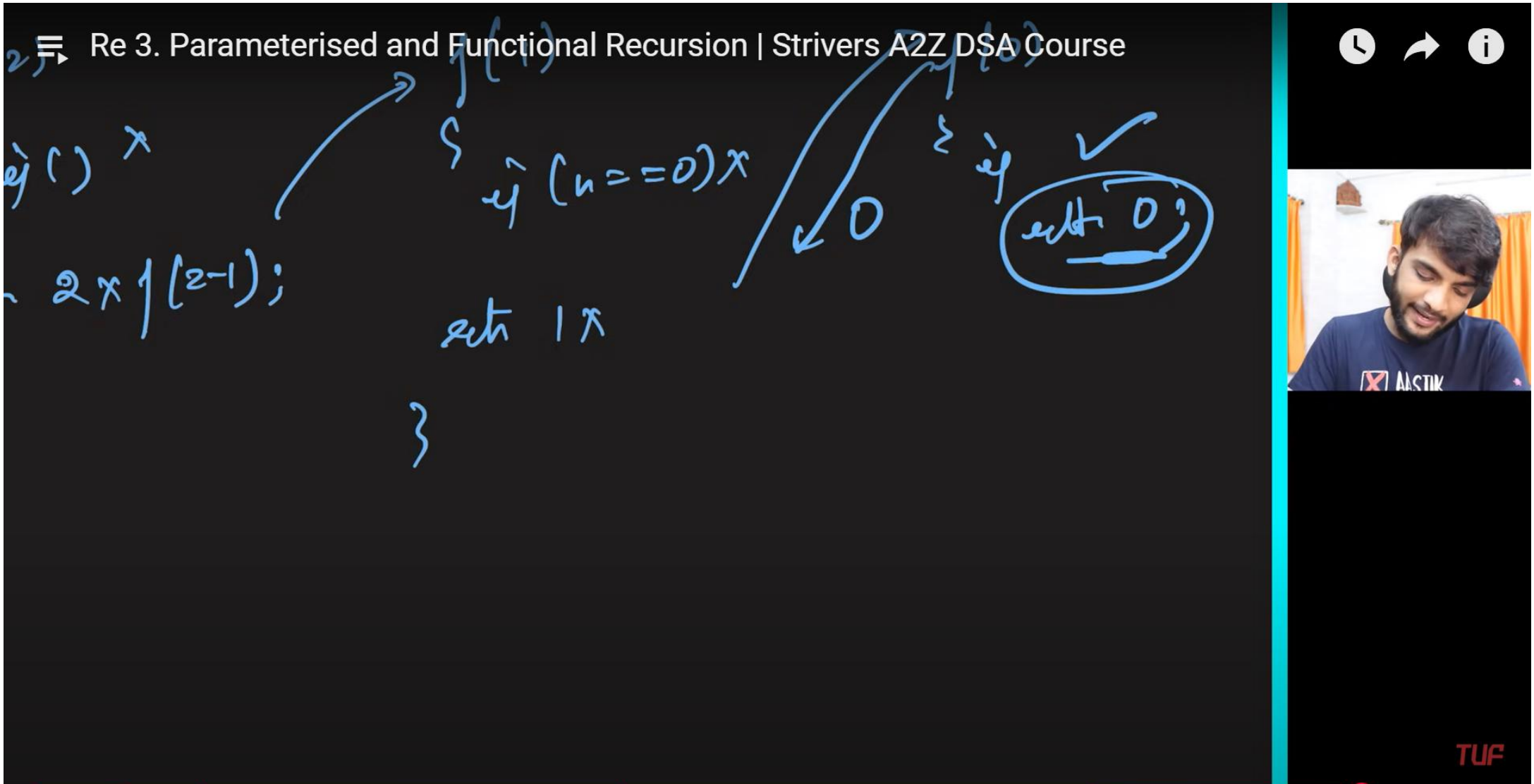
{

if ✓

return 0;

TUF

Re 3. Parameterised and Functional Recursion | Strivers A2Z DSA Course



16:27 / 18:13 • Code

```
) x
0;
f(n-1)

f(2)
{
if() x
ret 2 x f(2-1);
}

f(1)
{
if (n==0) x
ret 1 x 0
}
```

0

0

TUF

```
f(3)
{
    if (n==0)
        return 0;
    return 3 * f(n-1)
}

main()
{
    n = 3
    print(f(n))
}
```

```
f(2)
{
    if()
    return 2 * f(2-1);
}
```

0

0

```
f(3)
{
  if (n==0)
    return 0;
  return 3 x f(n-1)
}

main()
{
  n = 3
  print(f(n))
}
```

```
f(2)
{
  if()
  return 2 x f(2-1);
}
```

TUF

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int fact(int n)
{
    if( n == 0) return 1;
    return n * fact(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << fact(n);
    return 0;
}
```

input.txt

```
1
```

output.txt

```
0
```

[Finished in 0.8s]

code.cpp | stdc++.h

```cpp
#include<bits/stdc++.h>
using namespace std;
int fact(int n)
{
    if( n == 0) return 1;
    return n * fact(n-1);
}
int main() {
    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n = 3;
    cout << fact(n);
    return 0;
}
```

input.txt

```
1
```

output.txt

```
0
```

[Finished in 0.8s]

TUF

f(n)

if (n==0)

return 0;

return n × f(n-1);

}

main ()

{

n

print (f(n))

}

f(2)

if () ×

return 2 × f(2-1);

}

TC → O(N)

SC → O(N)

TUF

f(n)

if (n==0)

return 0;

return n × f(n-1);

}

main()

{

n ← 3

print(f(n))

}

f(2)

if()

return 2 × f(2-1);

}

TC → O(N)

SC → O(N)

TUF