

Section 8 Space complexity (Slides 55-59)

```
1 int sum (int k) {
2     if (k <= 0) {
3         return 0;
4     }
5     return k + sum(k-1);
6 }
7
8 int main(void) {
9     int x = sum(3);
10 }
```

Sum:

k: 0

r/a: sum: 5

) return
0

Sum:

k: 1

r/a: sum: 5

) return
1

Sum:

k: 2

r/a: sum: 5

) return
3

Sum:

k: 3

r/a: main: 10

) return
6

main:

x: ???

r/a: 0\$

sum(0)

↑
return
0

sum(1)

↑
return
1

sum(2)

↑
return
3

sum(3)

↑
return
6

main()

- Every stack frame contains a # to be added to the sum.
- We need every frame.

The sum function

- make a recursive call first
- use the return value of the recursive call to calculate the result

Section 8 Space complexity (Slides 55-59)

```

1 int accsum (int k, int acc) {
2     if (k == 0) {
3         return acc;
4     }
5     return accsum (k-1, k+acc);
6 }
7
8 int recursive_sum (int k) {
9     return accsum (k, 0);
10    }
11
12 int main(void) {
13     int x = recursive_sum(3);
14     }

```

The accsum function

- perform calculation first.
- execute the recursive call.
- pass the result of the current step to the next recursive step.

The return value of any recursive step is the same

Consequence:

- don't need the current stack frame for the next recursive step.
- reuse the current stack frame for the next recursive step.

accsum:

k: 0

acc: 6

r/a: accsum: 5

return 6

accsum:

k: 1

acc: 5

r/a: accsum: 5

return 6

accsum:

k: 2

acc: 3

r/a: accsum: 5

return 6

accsum:

k: 3

acc: 0

r/a: recursive_sum: 9

return 6

recursive_sum:

k: 3

r/a: main: 13

return 6

main:

x: ???

r/a: 05

accsum (0, 6)

return 6

accsum (1, 5)

return 6

accsum (2, 3)

return 6

accsum (3, 0)

return 6

recursive_sum (3)

return 6

main()

Section 8 Space complexity (Slides 55-59)

```

1 int accsum(int k, int acc) {
2     if (k == 0) {
3         return acc;
4     }
5     return accsum(k-1, k+acc);
6 }
7
8 int recursive_sum(int k) {
9     return accsum(k, 0);
10 }
11
12 int main(void) {
13     int x = recursive_sum(3);
14 }
```

accsum :

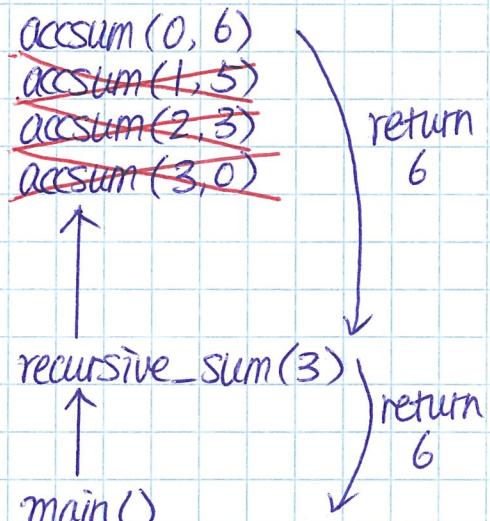
k:	3	2	1	0
acc:	0	3	5	6
r/a:	recursive-sum: 9			

recursive_sum :

k:	3
r/a:	main: 13

main :

x:	???
r/a:	OS



Tail recursion:

- can be transformed into a loop.
- could in principle run forever. (# of stack frames does not grow.).