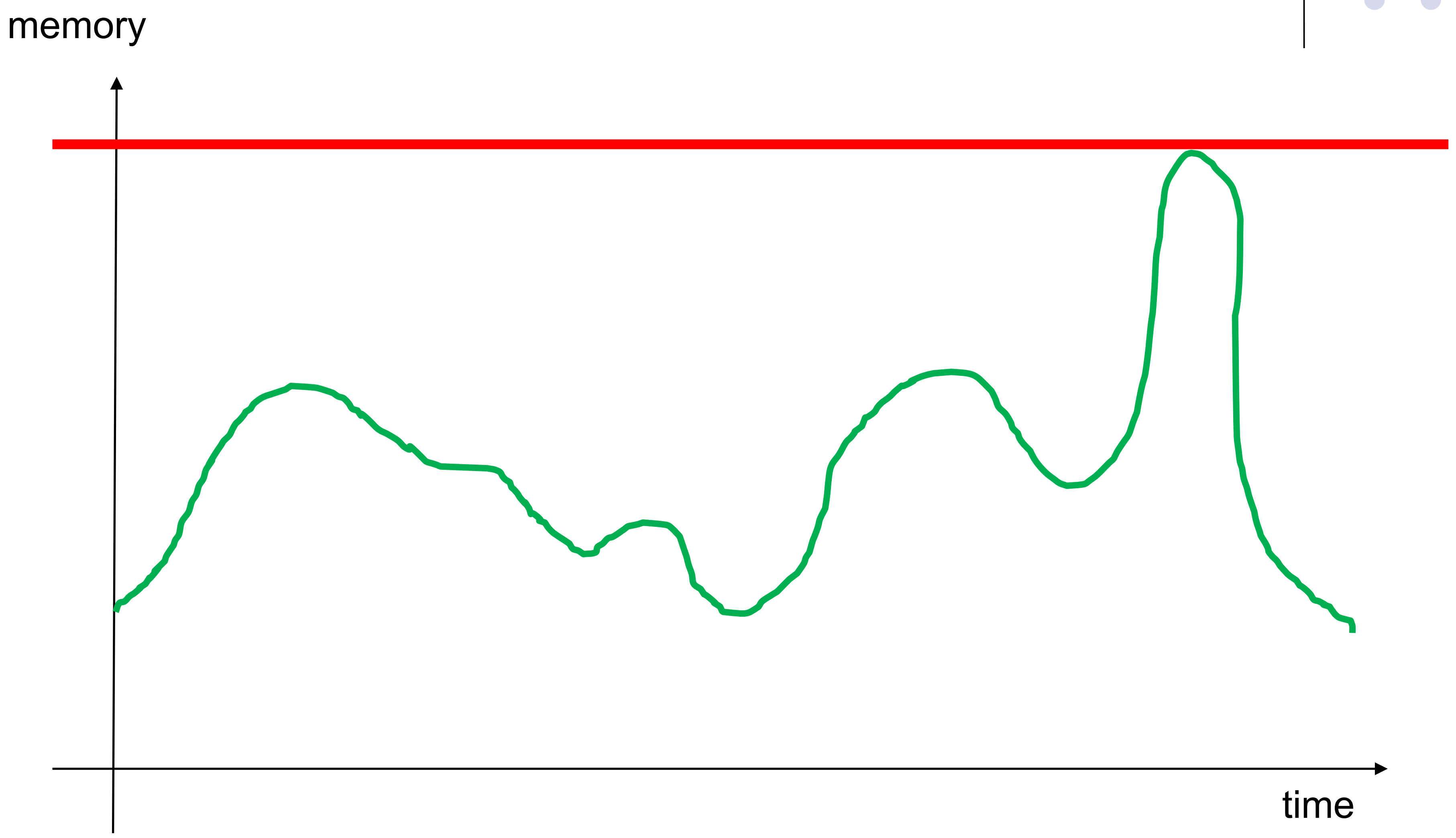
How much memory does my program need?



How much memory does my program need?

- Three main memories:
- Memory for static duration variables (data memory)
- Working memory for every function
- Dynamically allocated memory

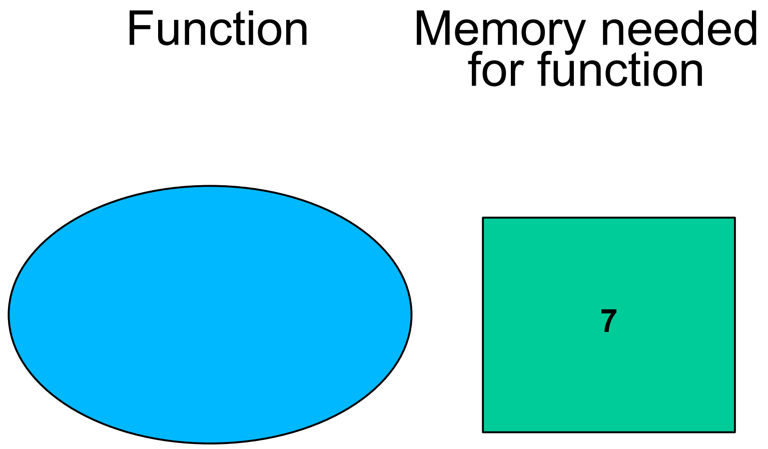
Data memory

- Comprises memory for all static storage duration variables (all global variables, plus local ones with `static`).
- Can be calculated (or at least you can have a pretty good estimate) if you know basic platform characteristics.
- Memory for thread duration variables only needs to be multiplied with number of threads.

```
char a;
int b;
char c;
short d;
```

```
struct S {
  char a;
  int b;
  char c;
  short d;
};
```

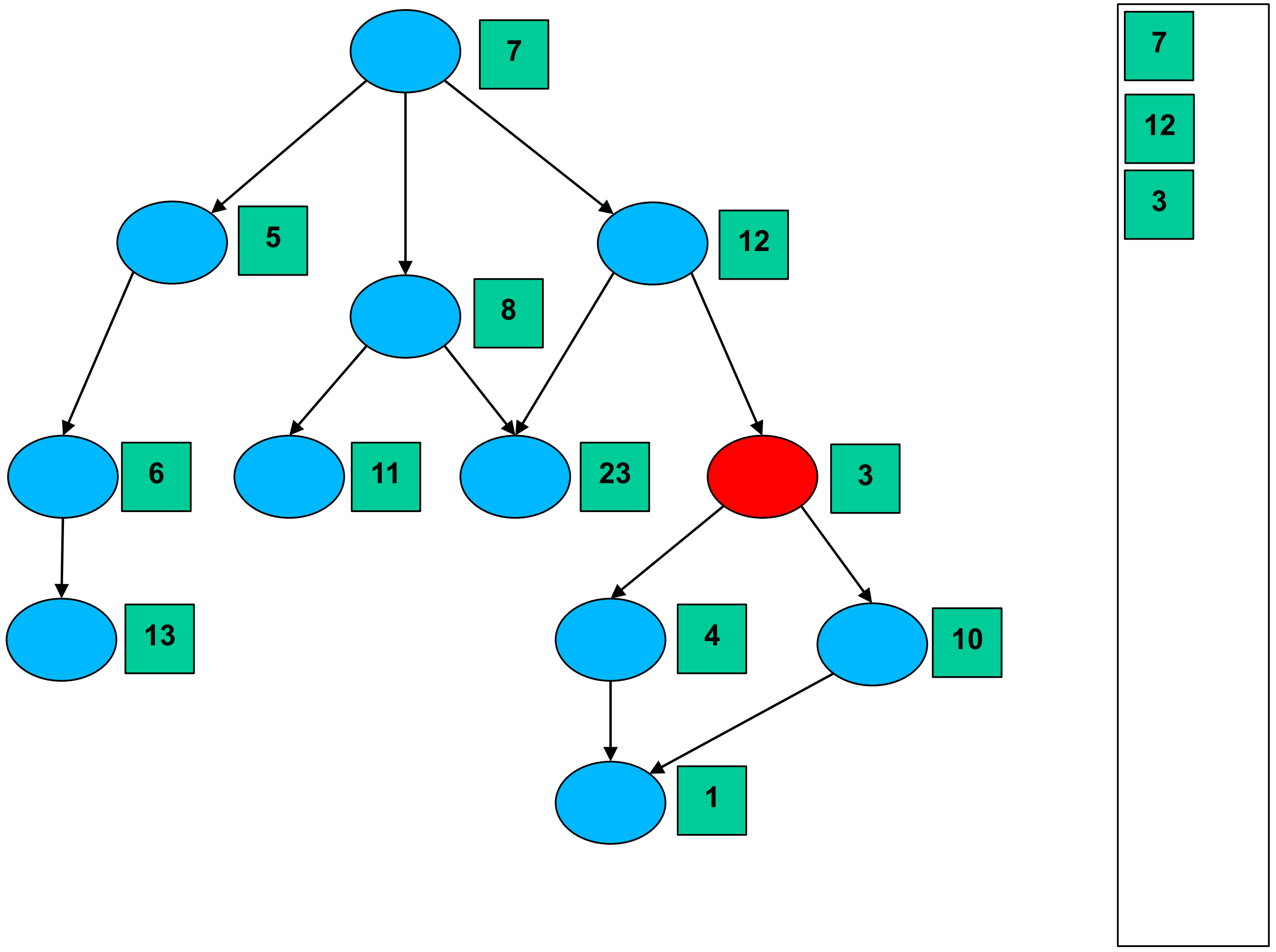
Function memory



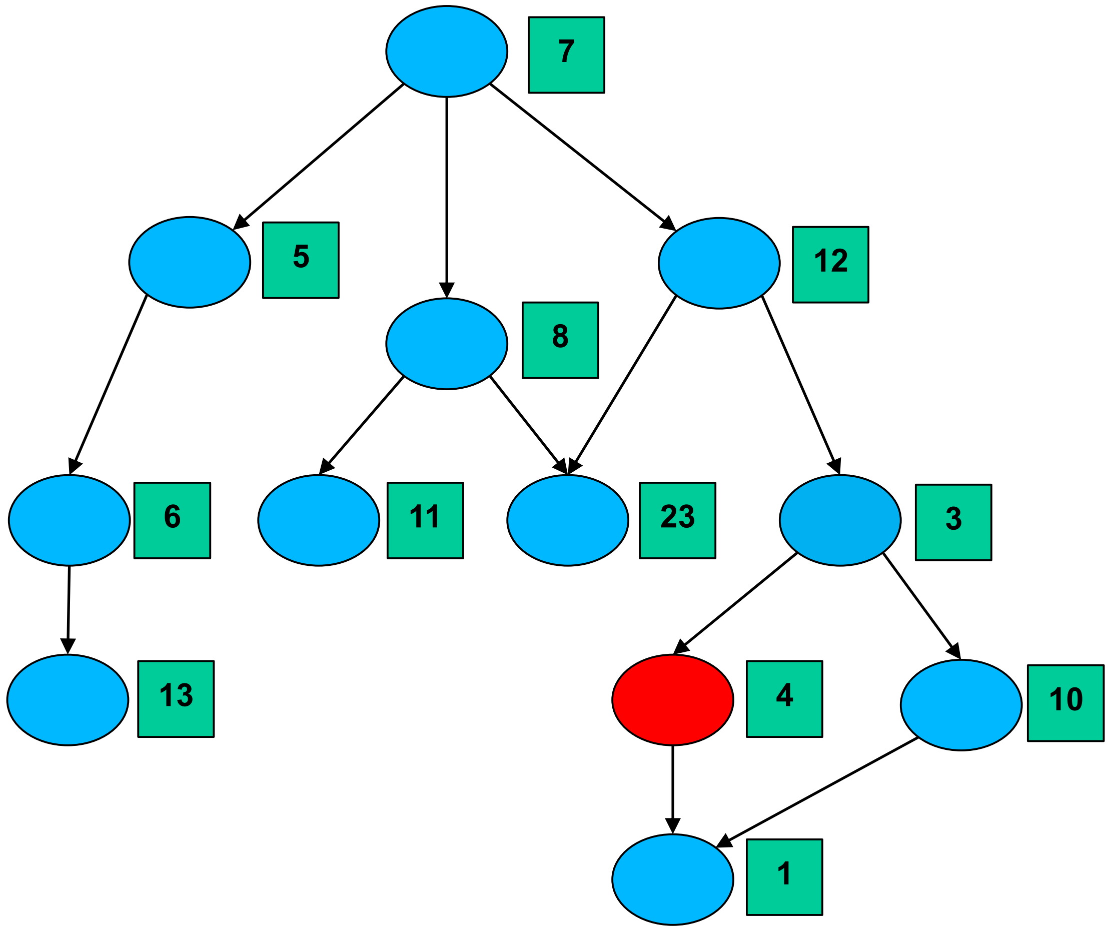
- A function needs memory for:
  - Storing return address
  - Local variables (including arguments)
  - Temporary storage

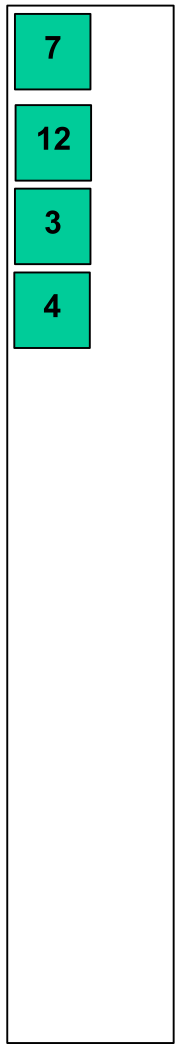
- Because of compiler optimizations, it is much harder to estimate how much memory a function will use.
- But we can have some general idea
- And we can get precise data from compiler (or some other tool), if we need it.

Function call graph and program stack



Function call graph and program stack

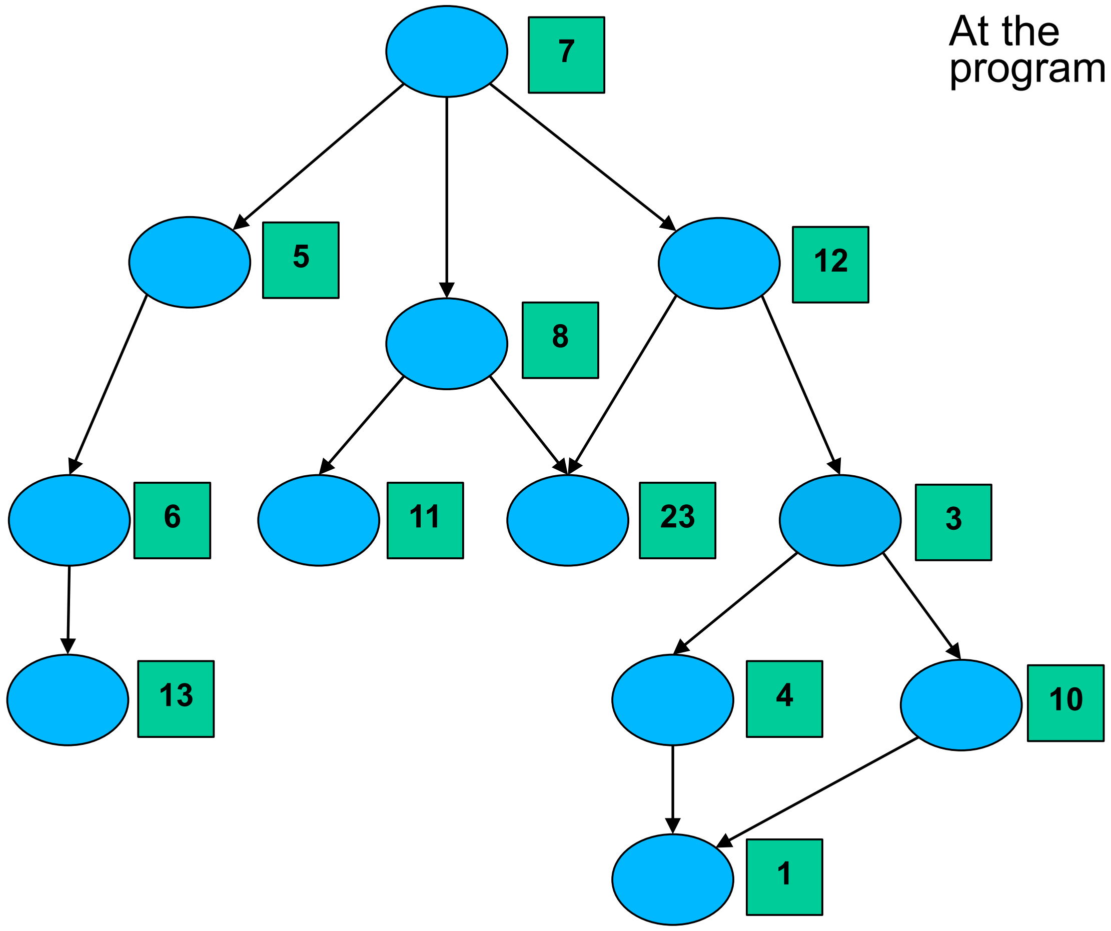




How big our stack has to be?

- Notice that we need to reserve memory for stack in advance.
- Two approaches for estimating how big stack has to be:
- Experimental
- Analytical
- Plus, stackless organization

Experimental



At the program start

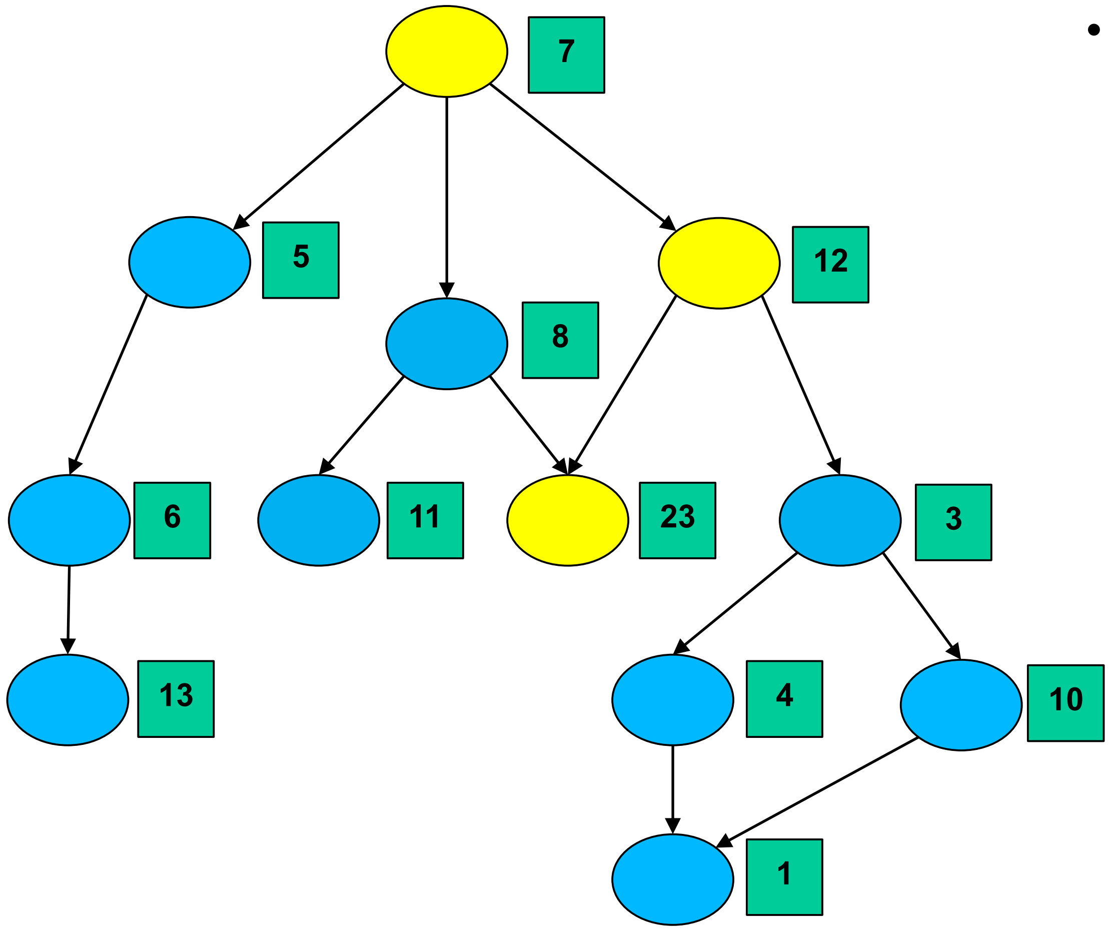
```
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
```

At the end

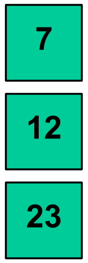
```
0xF123
0x5235
0x2134
0xA43E
0x35B4
0x2232
0x7CCE
0xFEA3
0x7593
0xA234
0xCF56
0xDD37
0x88A9
0xCF2C
0x9146
0xBA34
0x7593
0xA234
0xCF56
0xDD37
0x7593
0xA234
0xCF56
0xDD37
0x8821
0x32CD
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
0xBABA
```

26

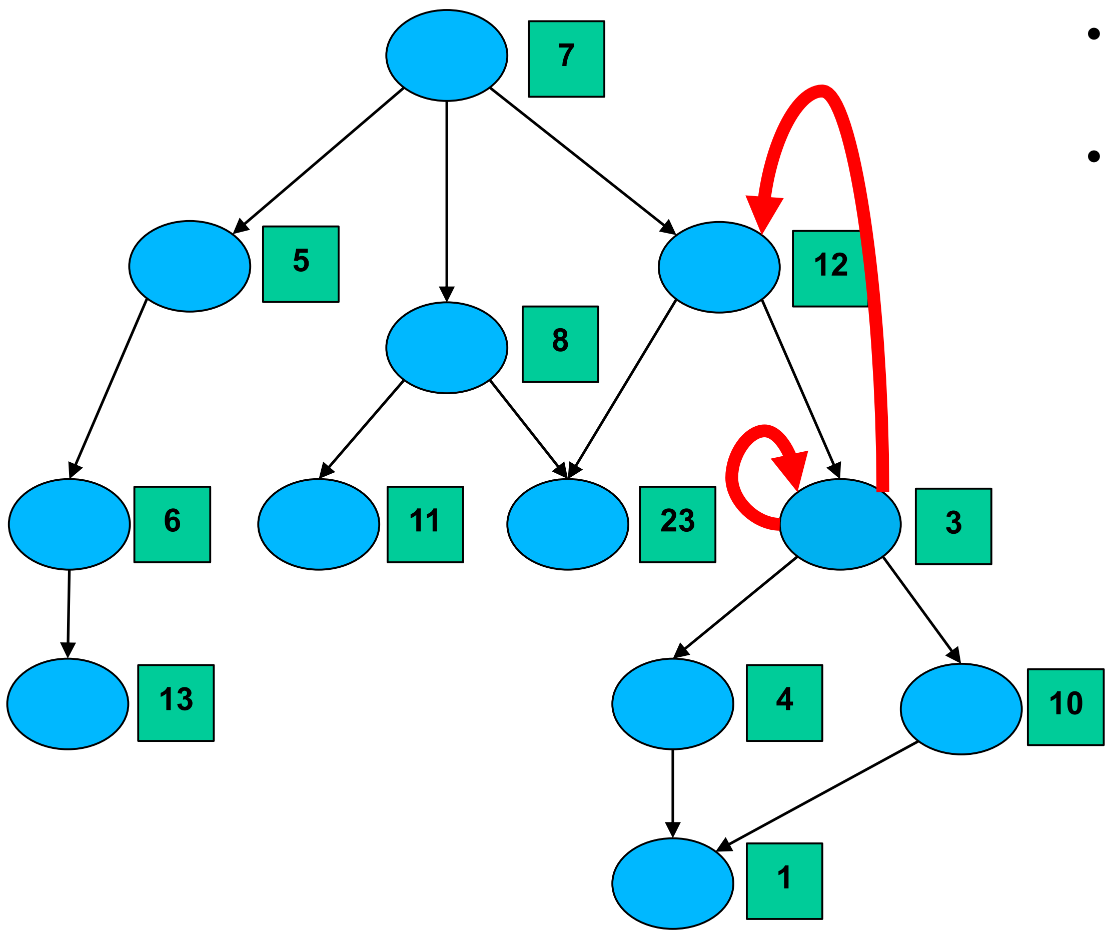
Analytical



- Need to know whole function call graph – which is hard to do



Analytical

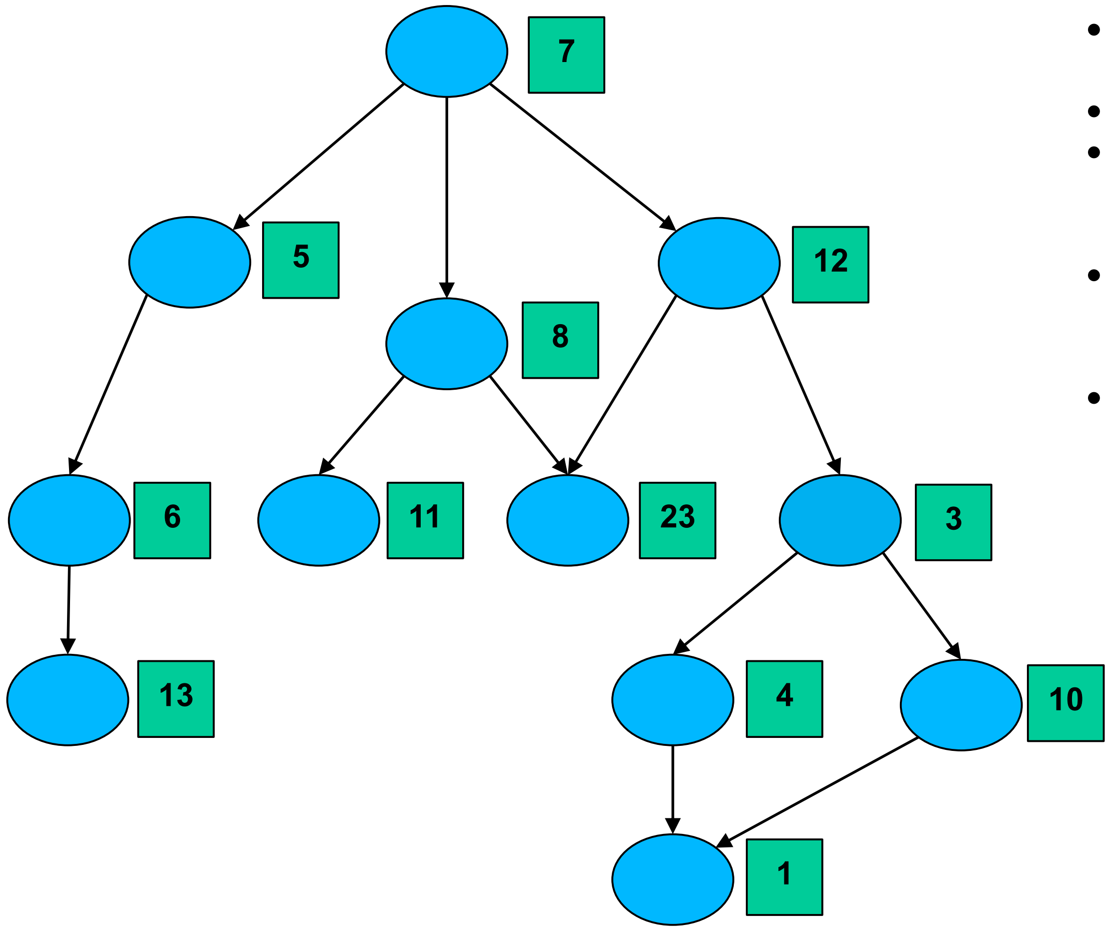


- Need to know whole function call graph – which is hard to do
- Additional problems in case of recursion and indirect calls (through pointers)

```
void (*p)(int x);
...
p(17);
```

? ☹

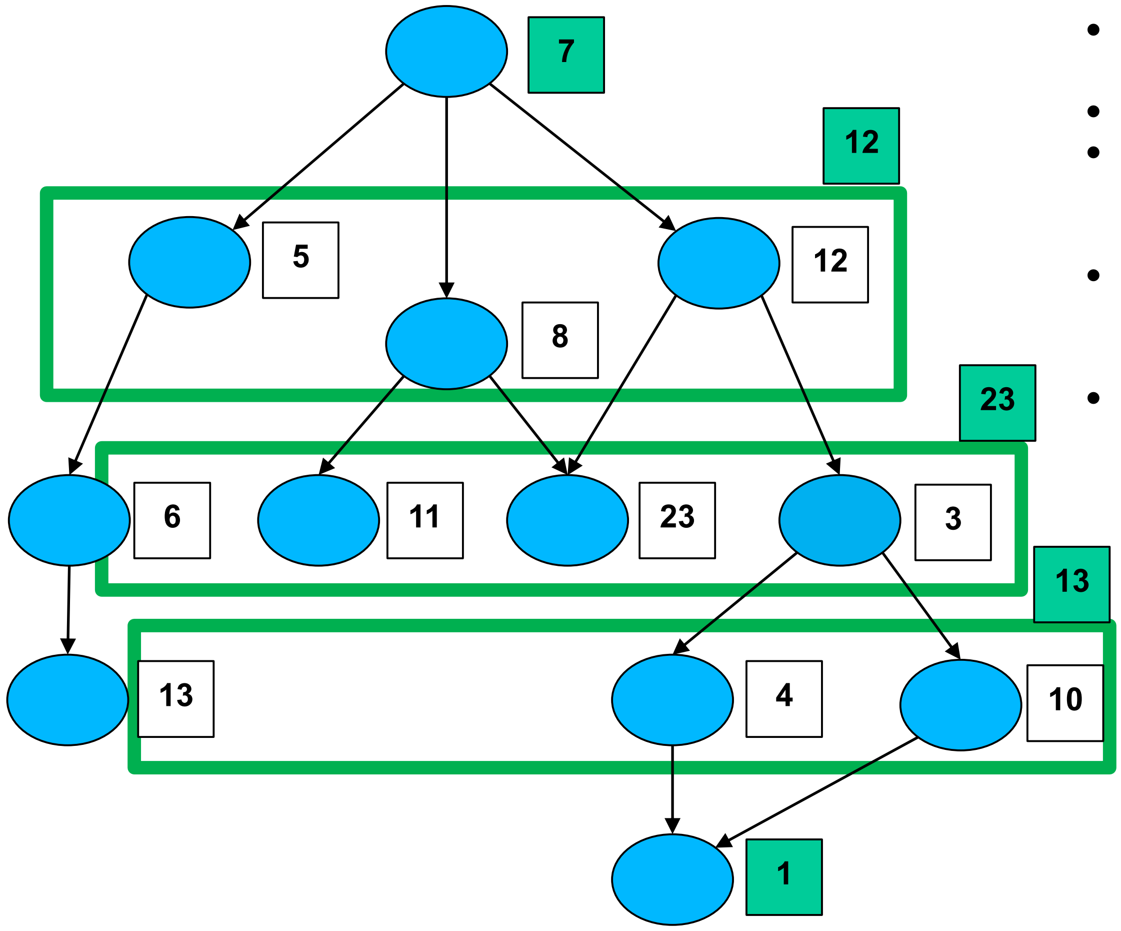
Stackless organization



- Every function has its own memory on a fixed address
- Recursion is not possible
- Indirect calls are harder to handle (although not impossible)
- Memory usage is bigger than with stack

- But, we are absolutely sure that we have enough memory

Stackless organization

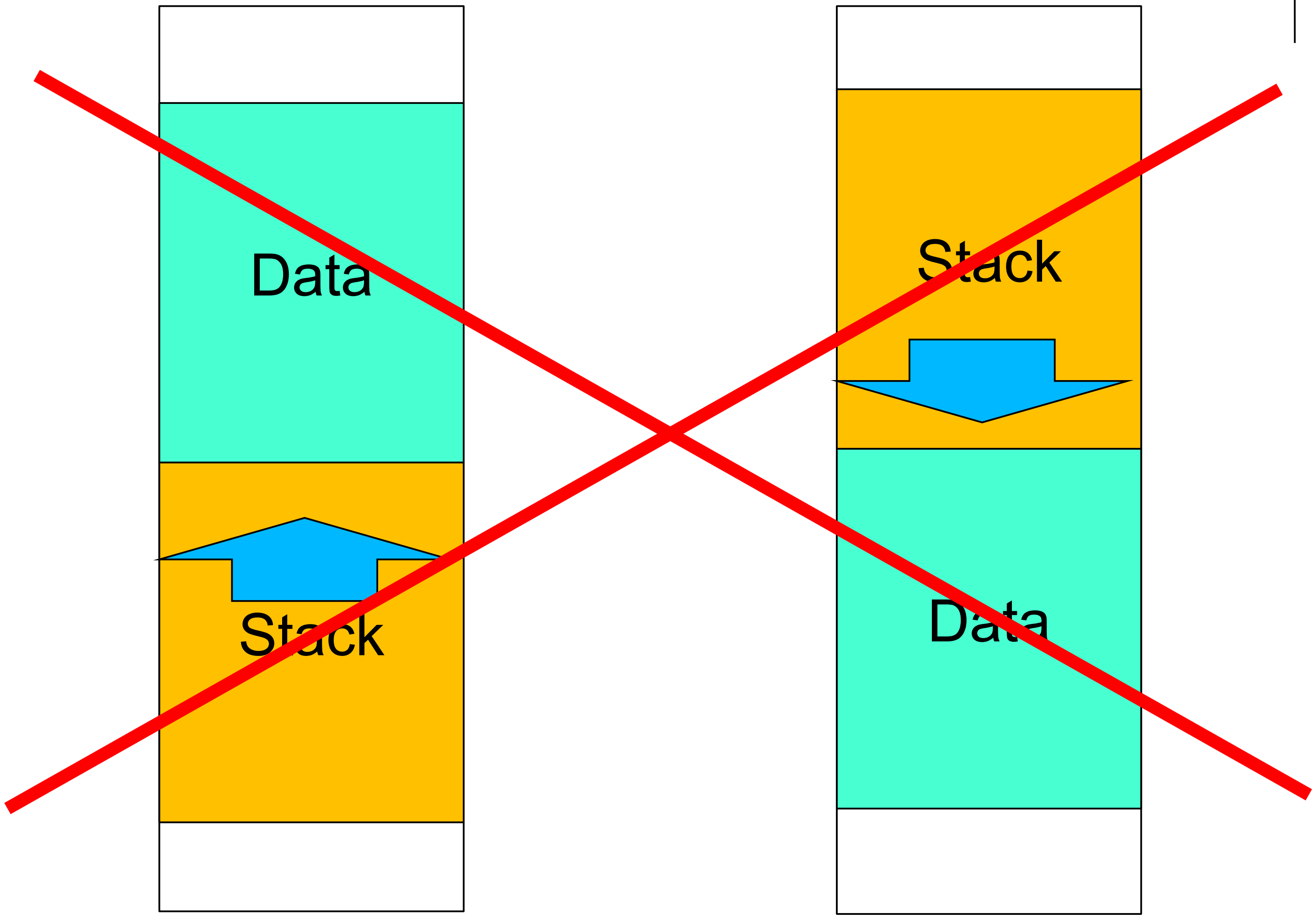


- Every function has its own memory on a fixed address
- Recursion is not possible
- Indirect calls are harder to handle (although not impossible)
- Memory usage is bigger than with stack

- But, we are absolutely sure that we have enough memory

Memory usage can be optimized by overlapping memories of functions that can not be on the same call line.

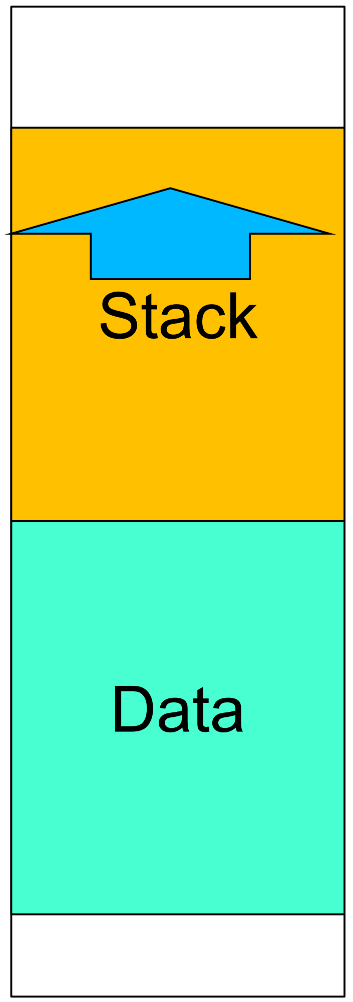
Where to place stack?



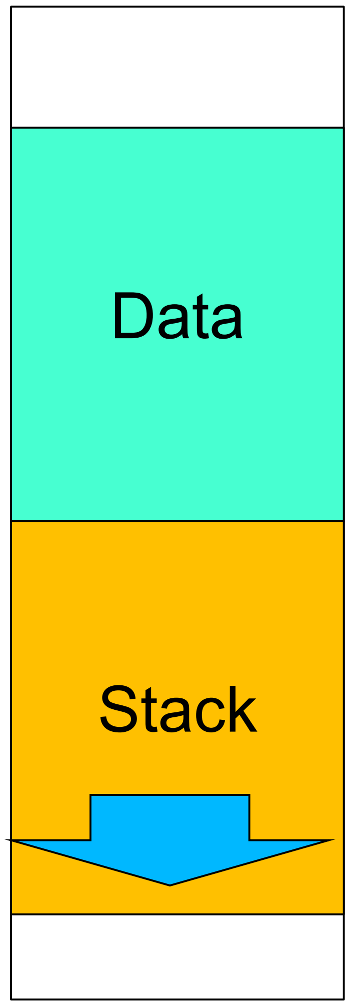
Stack grows upwards

Stack grows downwards

Where to place stack?



Stack grows upwards



Stack grows downwards