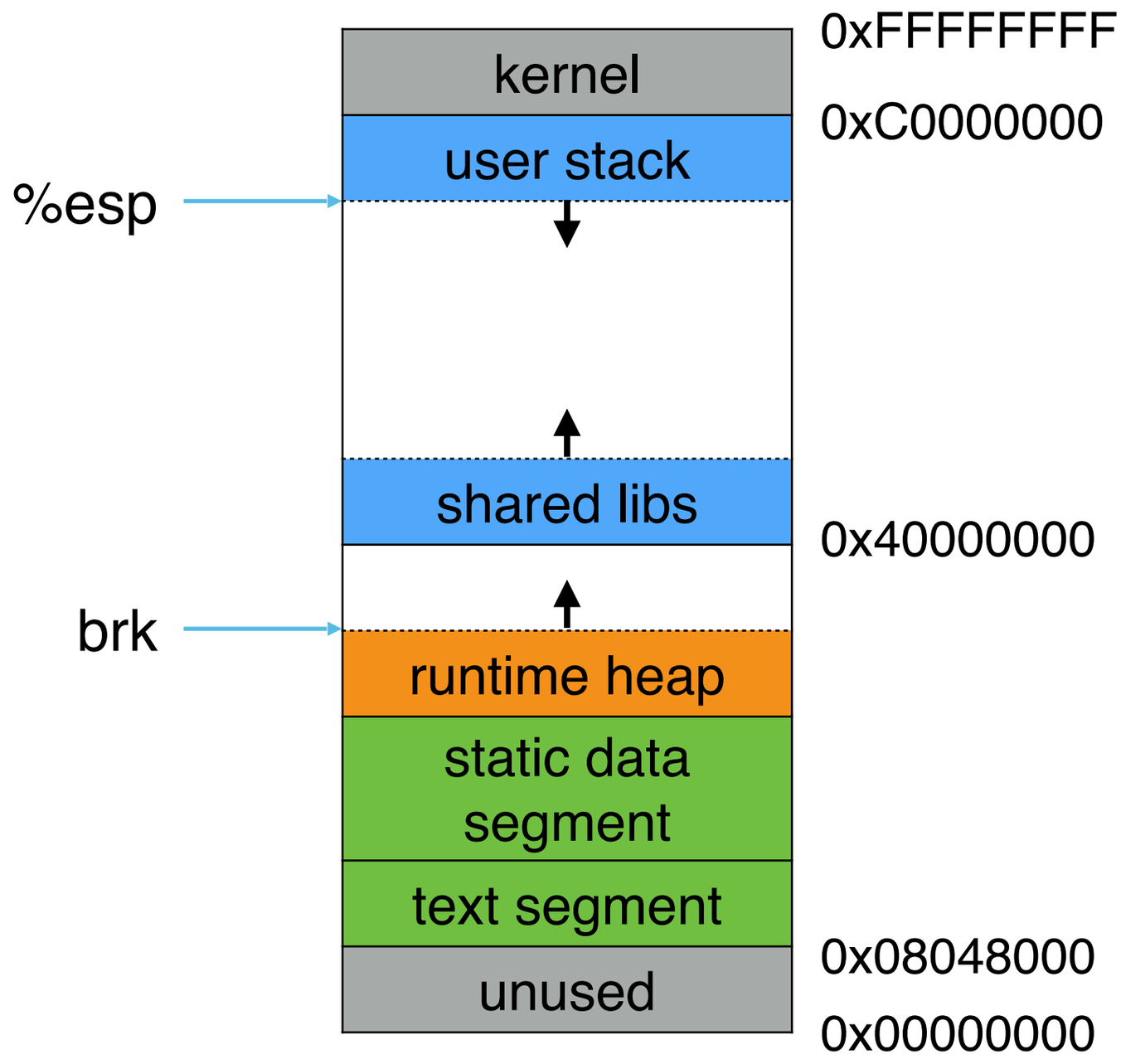
Linux process memory layout

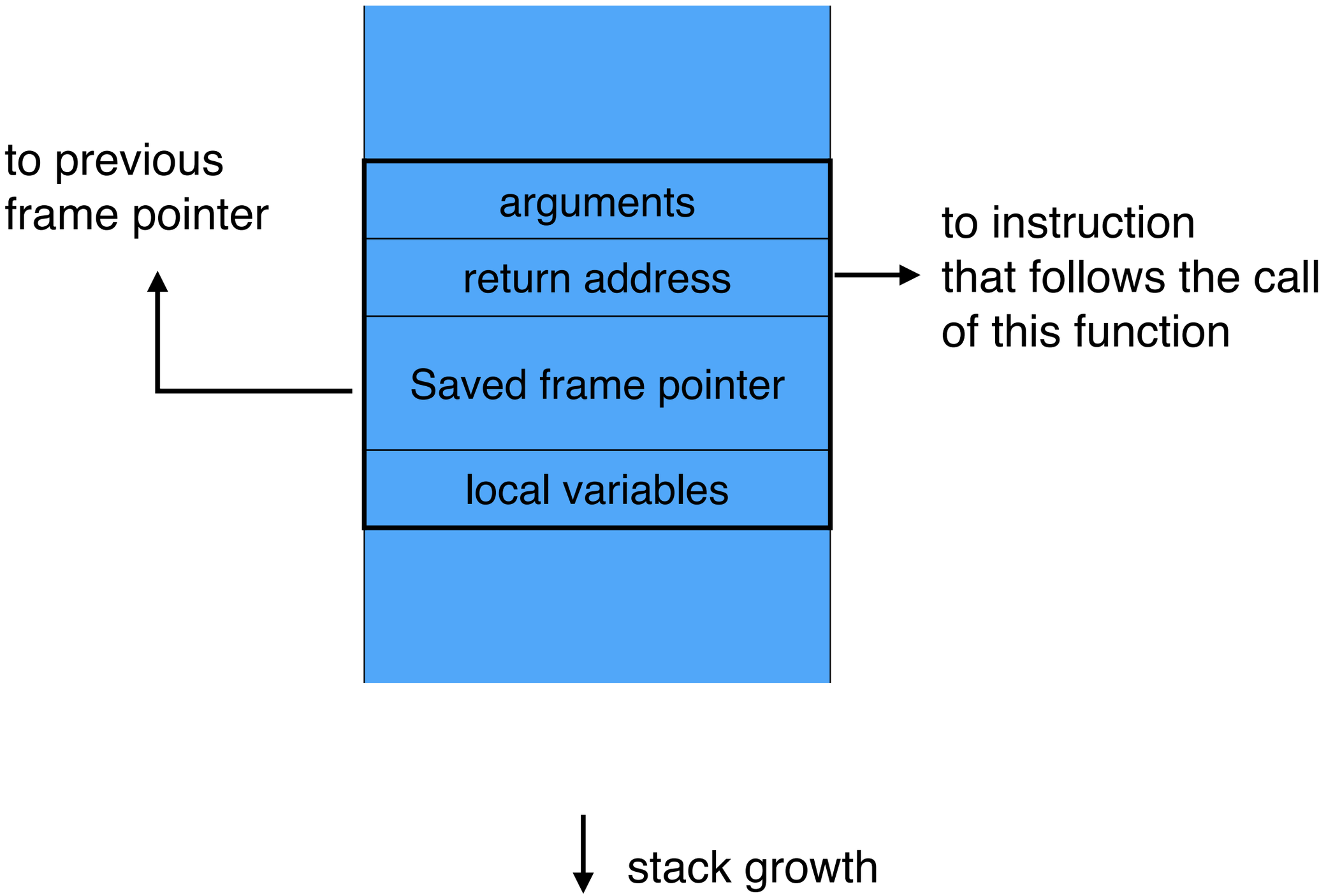
- Stack
- Heap
- Data segment
  - .data, .bss
- Text sement
  - Executable code



The Stack

- Stack divided into frames
  - Frame stores locals and args to called functions
- Stack pointer points to top of stack
  - x86: Stack grows down (from high to low addresses)
  - x86: Stored in %esp register
- Frame pointer points to caller's stack frame
  - Also called base pointer
  - x86: Stored in %ebp register

Stack frame


to previous
frame pointer
arguments
return address
Saved frame pointer
local variables
to instruction
that follows the call
of this function
stack growth

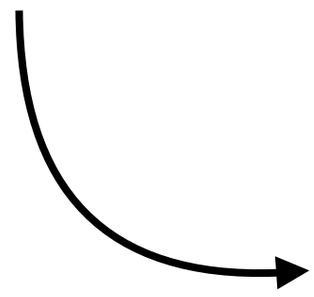
Example 0

```
int foobar(int a, int b, int c)
{
    int xx = a + 2;
    int yy = b + 3;
    int zz = c + 4;
    int sum = xx + yy + zz;

    return xx * yy * zz + sum;
}

int main()
{
    return foobar(77, 88, 99);
}
```

https://godbolt.org/z/3iFhjy

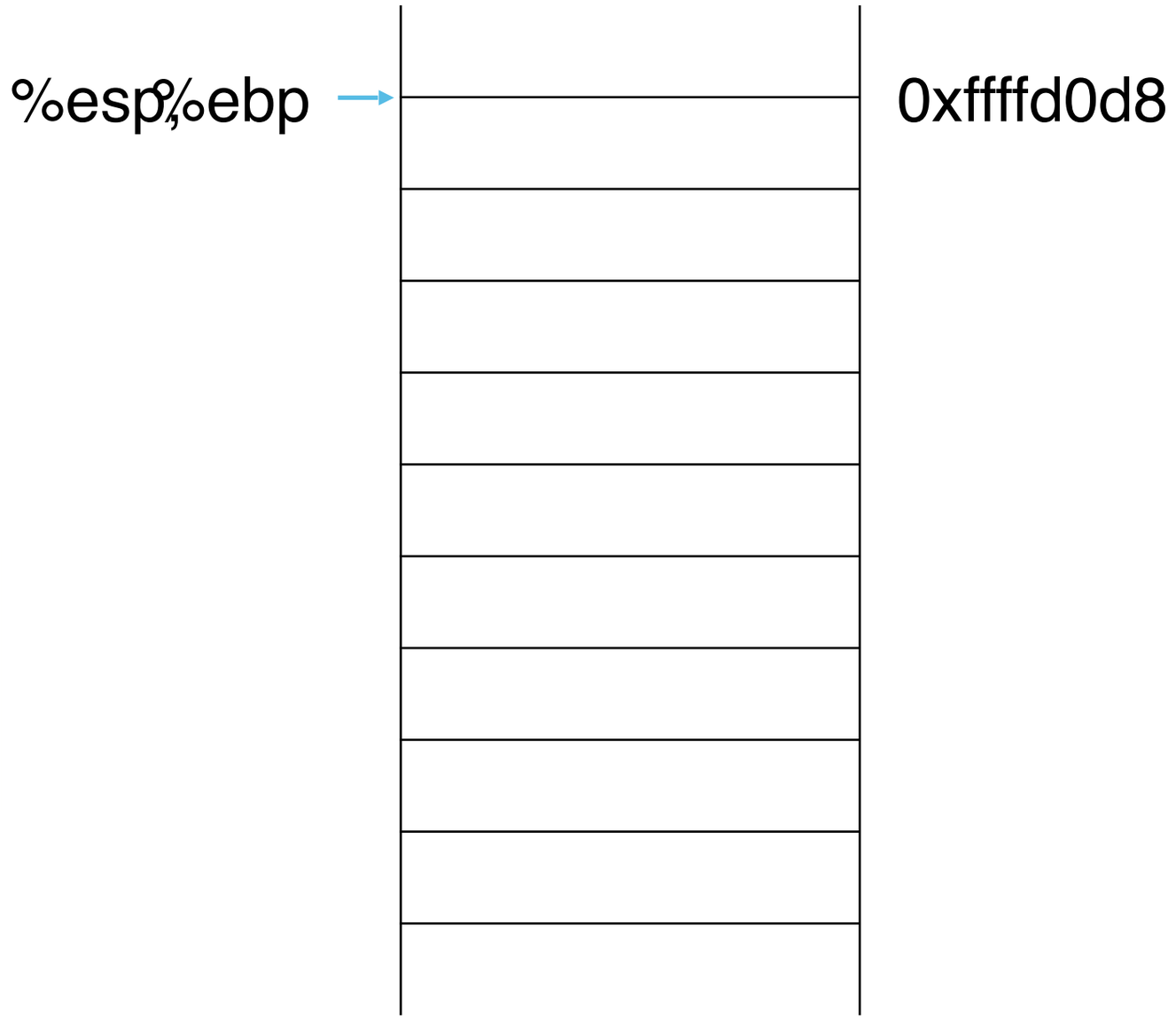
Compiled to x86

```c
int foobar(int a, int b, int c)
{
    int xx = a + 2;
    int yy = b + 3;
    int zz = c + 4;
    int sum = xx + yy + zz;

    return xx * yy * zz + sum;
}

int main()
{
    return foobar(77, 88, 99);
}
```

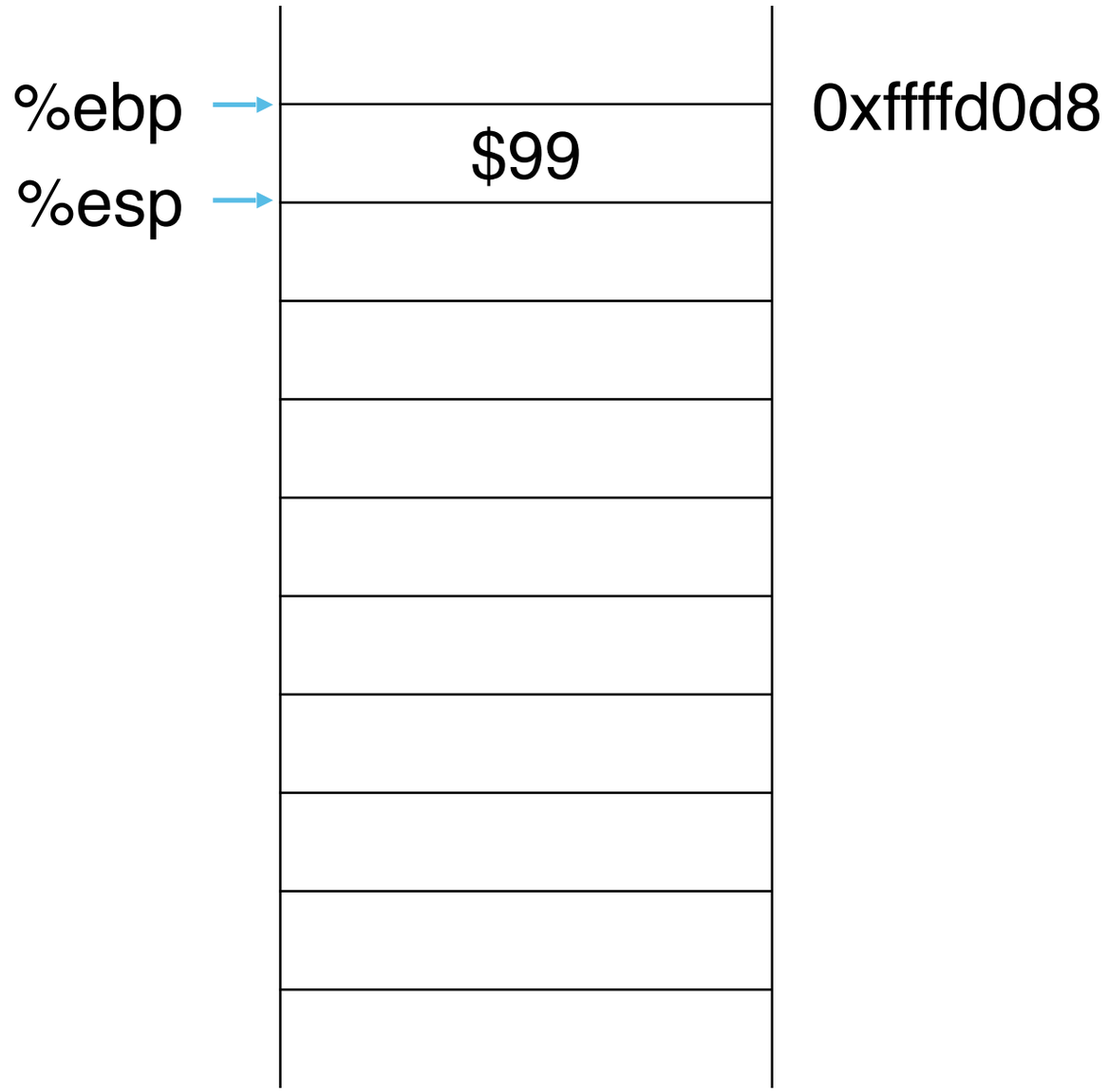


```asm
foobar(int, int, int):
        pushl   %ebp
        movl    %esp, %ebp
        subl    $16, %esp
        movl    8(%ebp), %eax
        addl    $2, %eax
        movl    %eax, -4(%ebp)
        movl    12(%ebp), %eax
        addl    $3, %eax
        movl    %eax, -8(%ebp)
        movl    16(%ebp), %eax
        addl    $4, %eax
        movl    %eax, -12(%ebp)
        movl    -4(%ebp), %edx
        movl    -8(%ebp), %eax
        addl    %eax, %edx
        movl    -12(%ebp), %eax
        addl    %edx, %eax
        movl    %eax, -16(%ebp)
        movl    -4(%ebp), %eax
        imull   -8(%ebp), %eax
        imull   -12(%ebp), %eax
        movl    %eax, %edx
        movl    -16(%ebp), %eax
        addl    %edx, %eax
        leave
        ret
main:
        pushl   %ebp
        movl    %esp, %ebp
        pushl   $99
        pushl   $88
        pushl   $77
        call    foobar(int, int, int)
        addl    $12, %esp
        nop
        leave
        ret
```

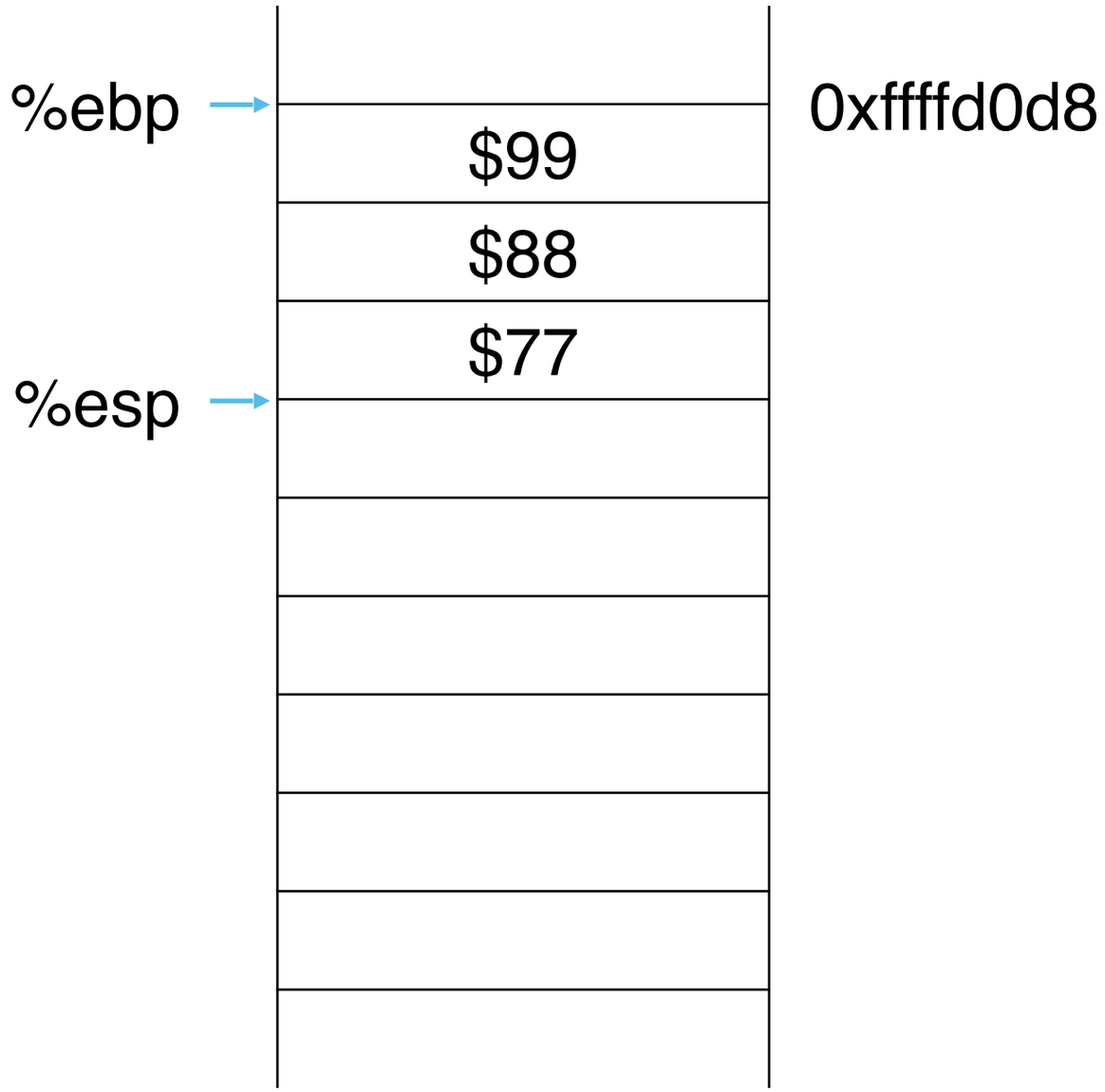
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



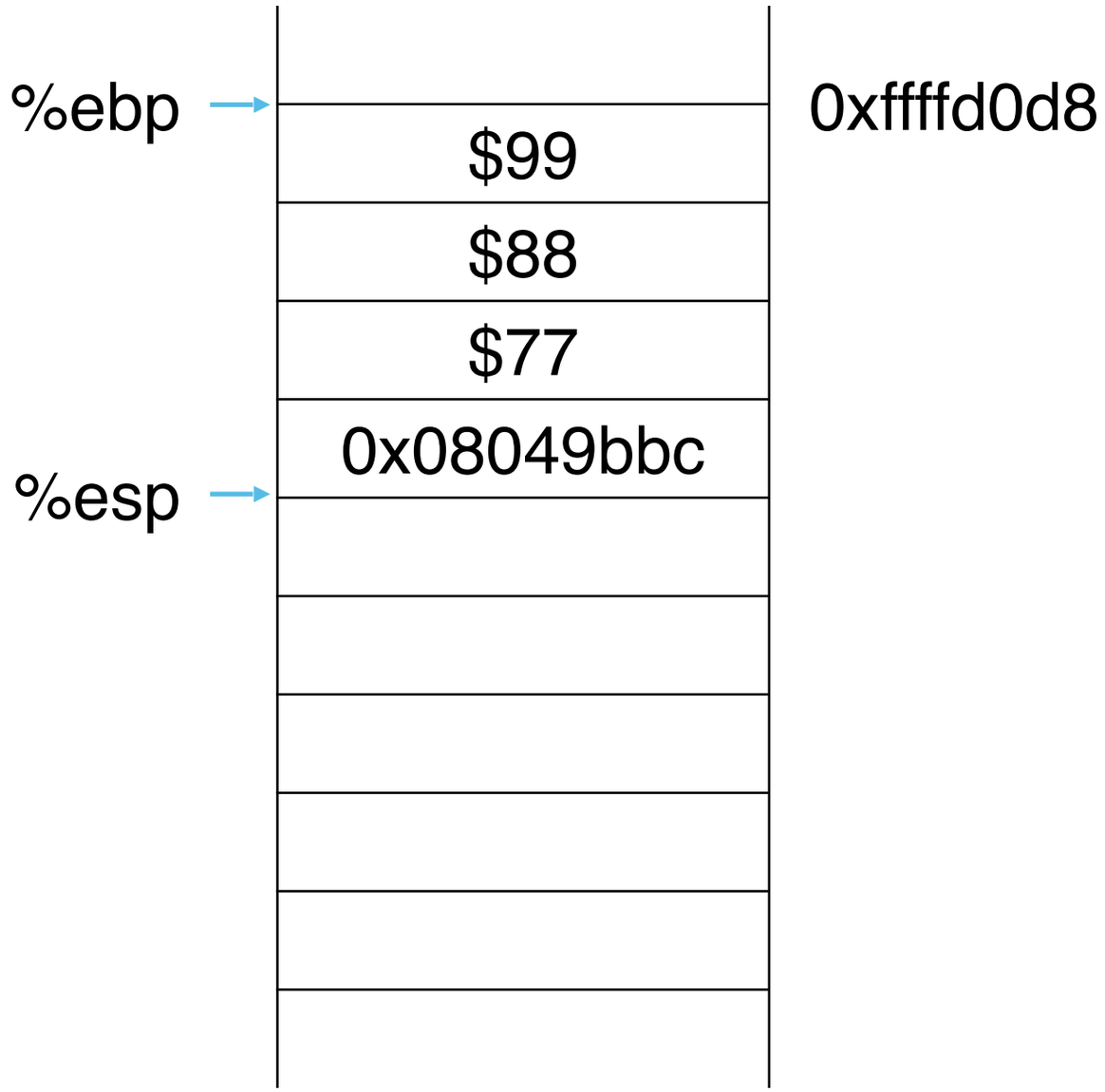
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```

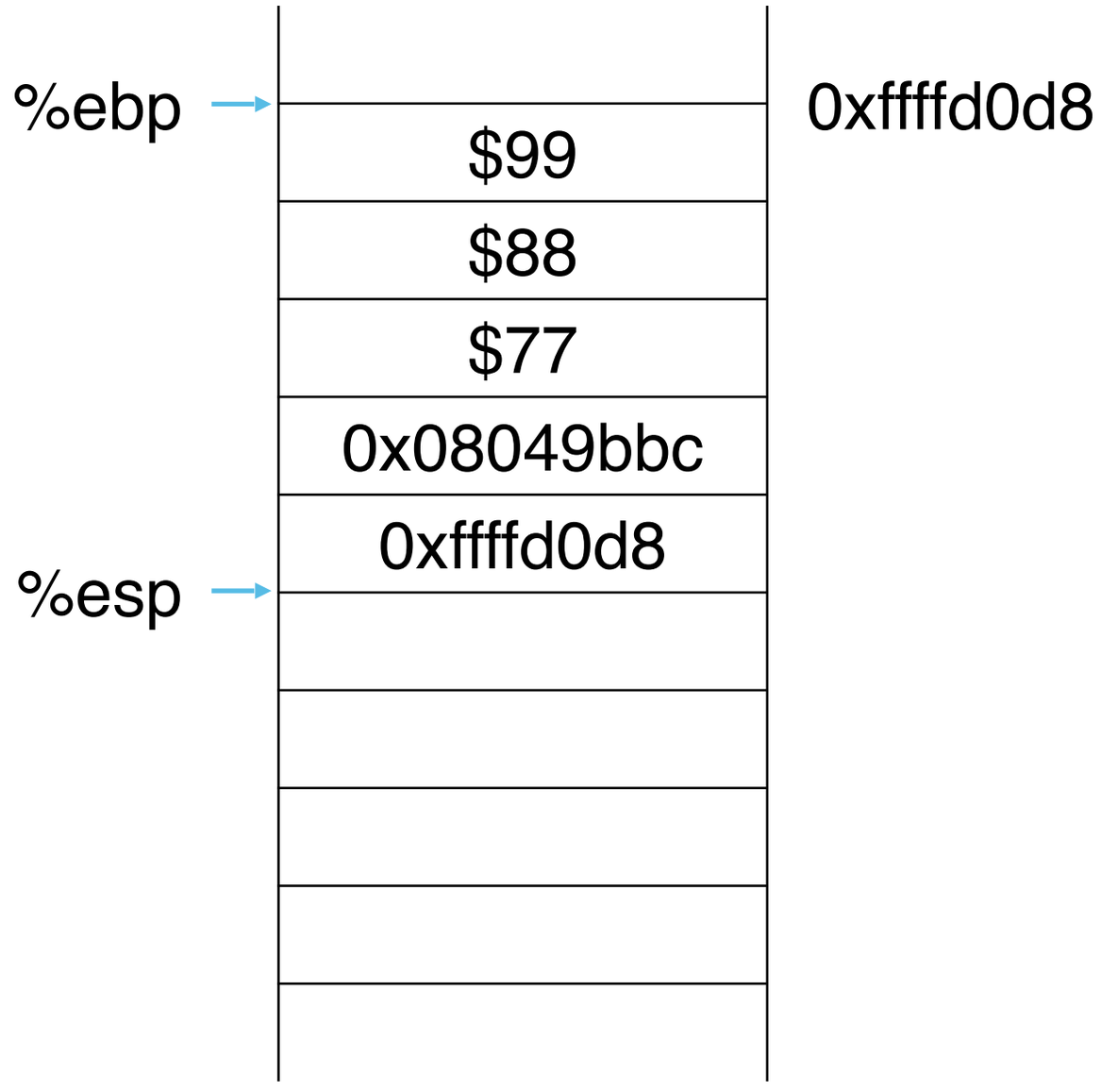


```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```

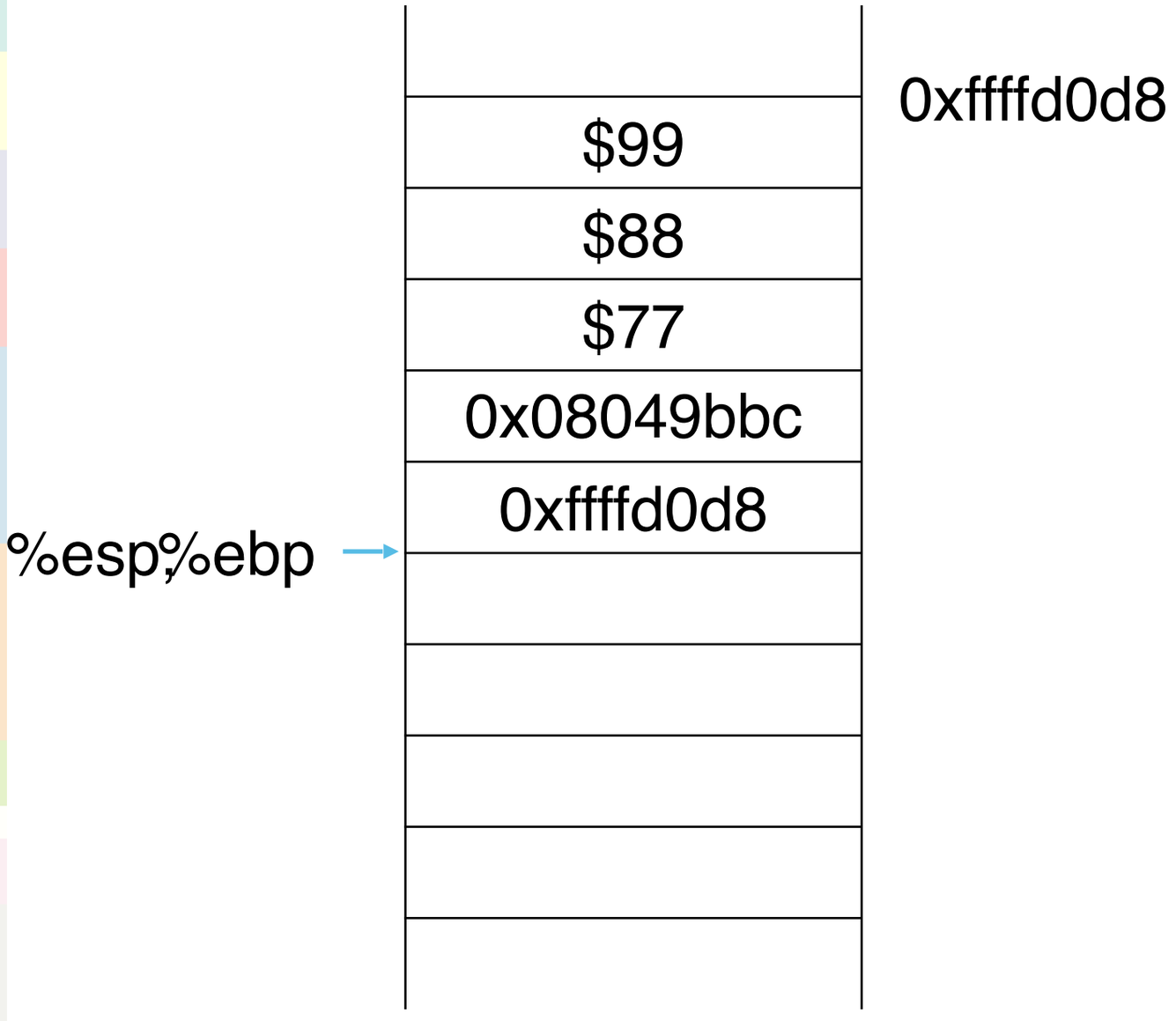


%eip = 0x08049ba7

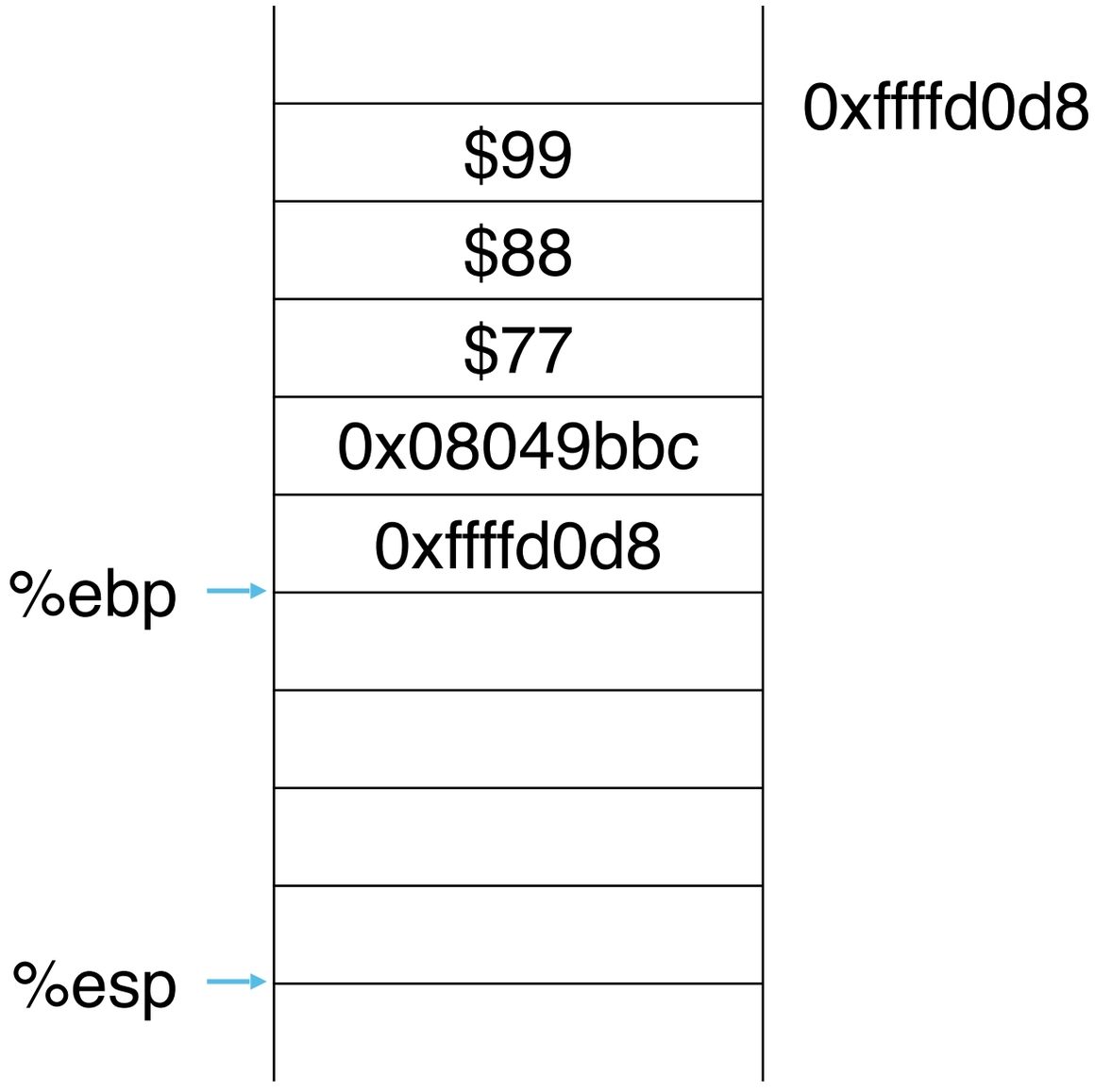
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



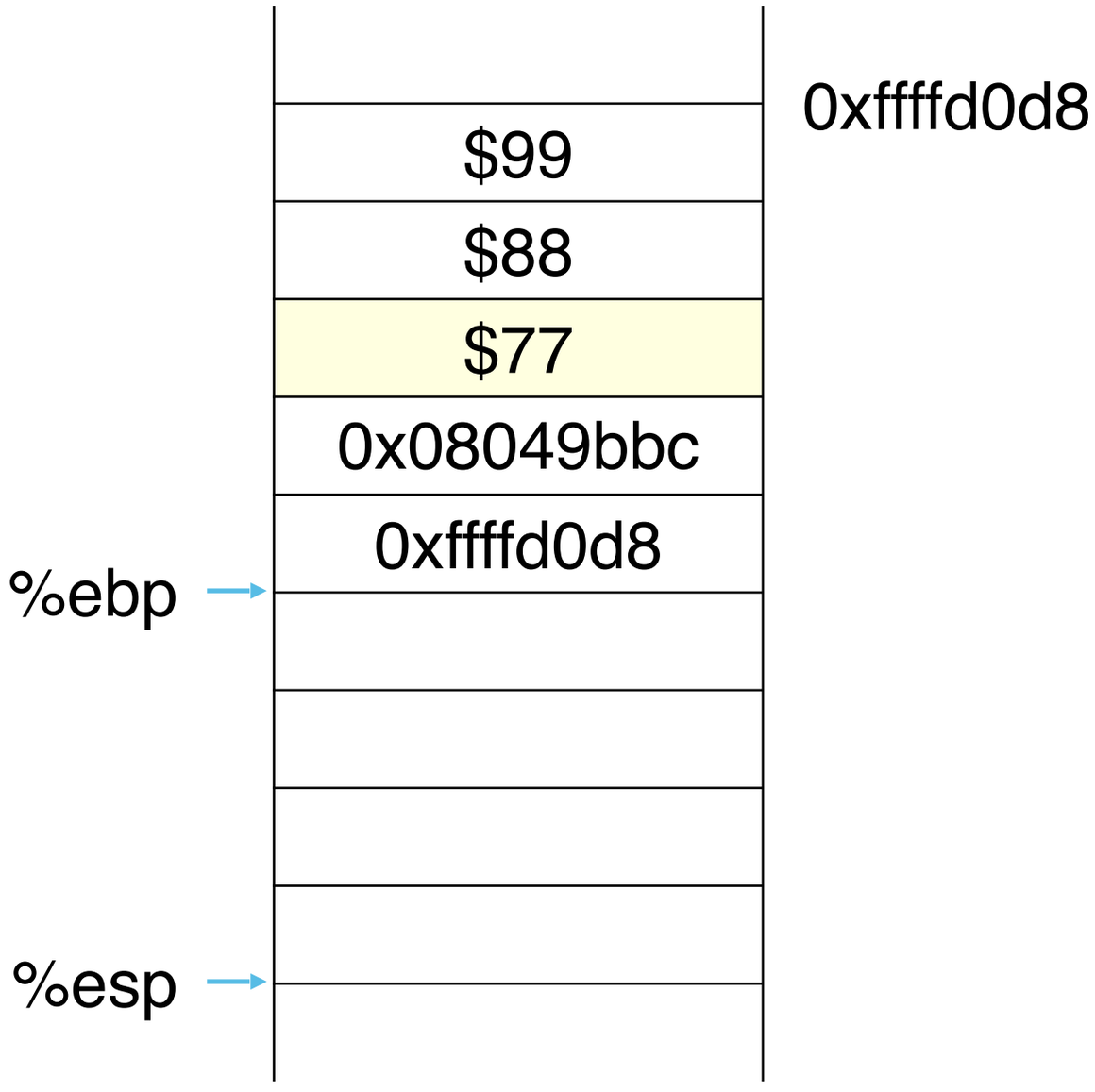
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



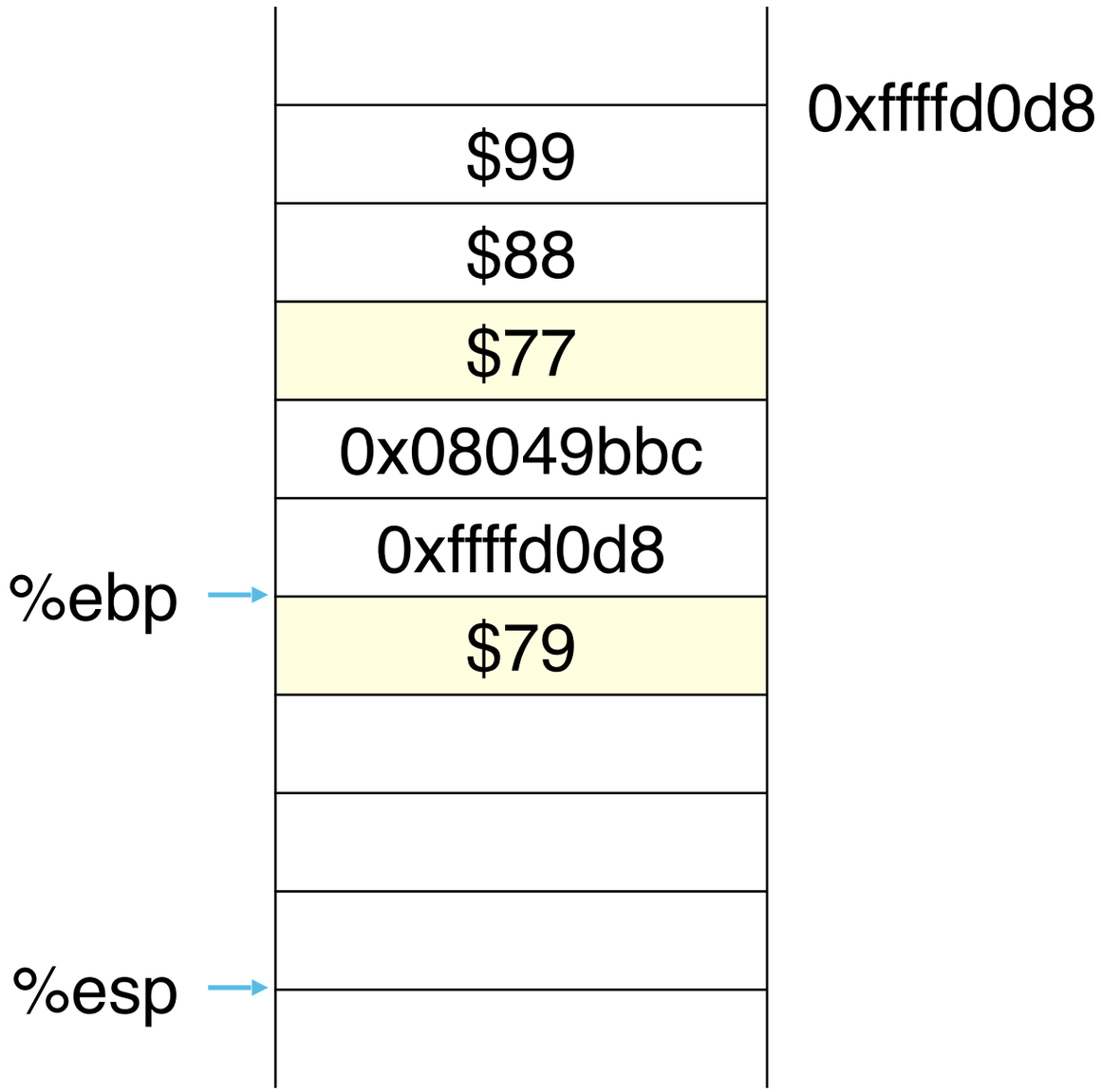
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



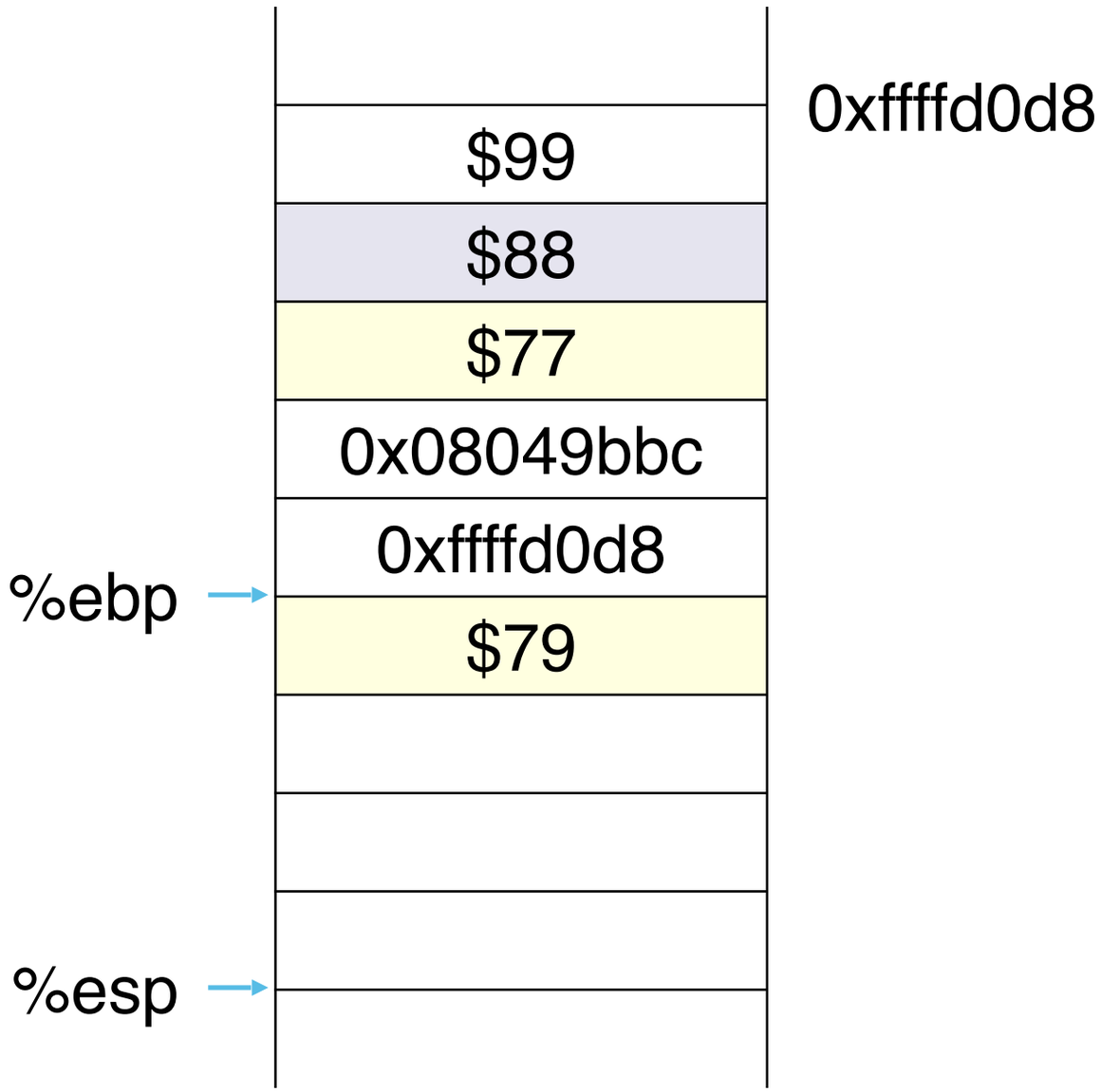
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



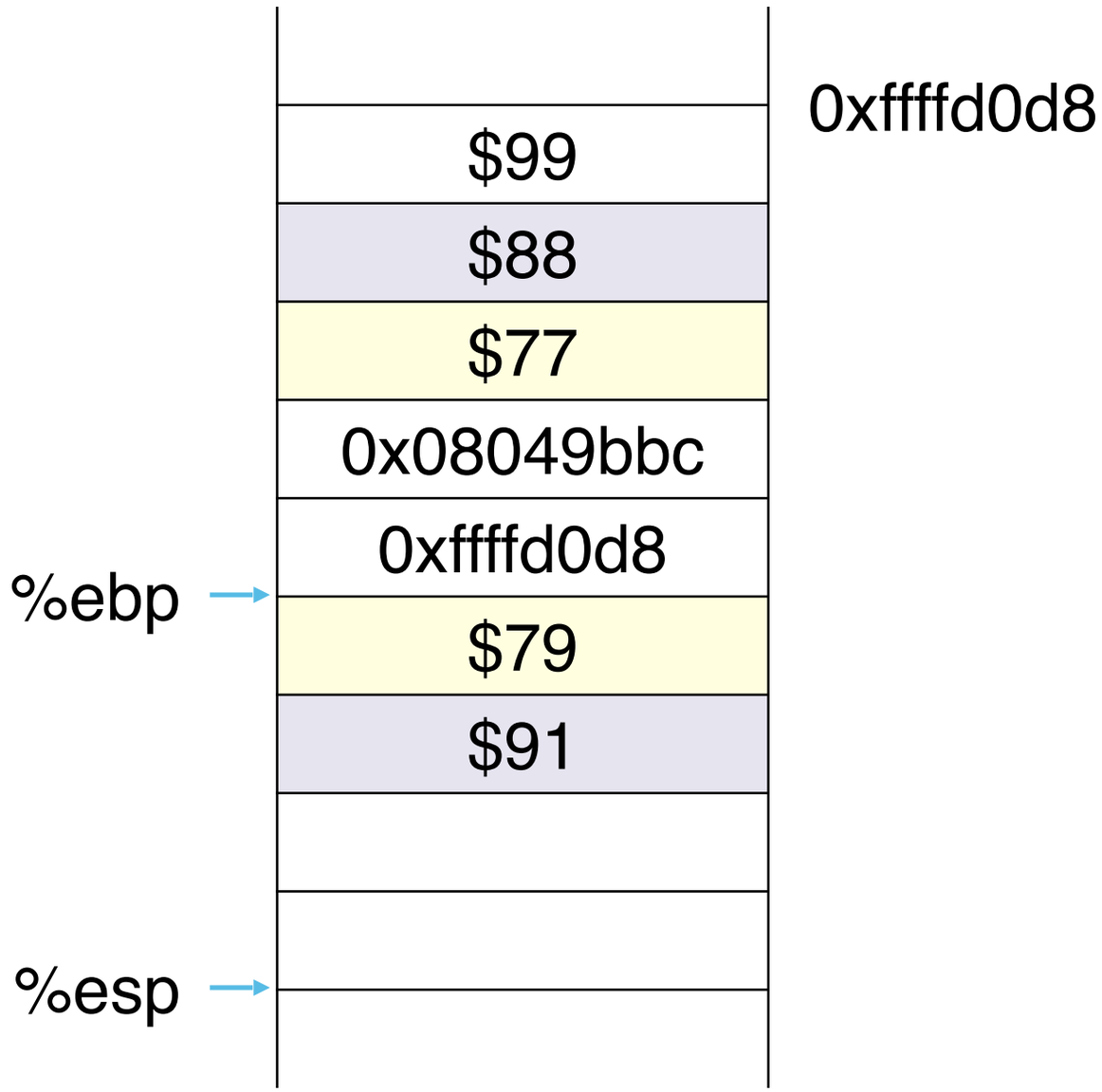
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



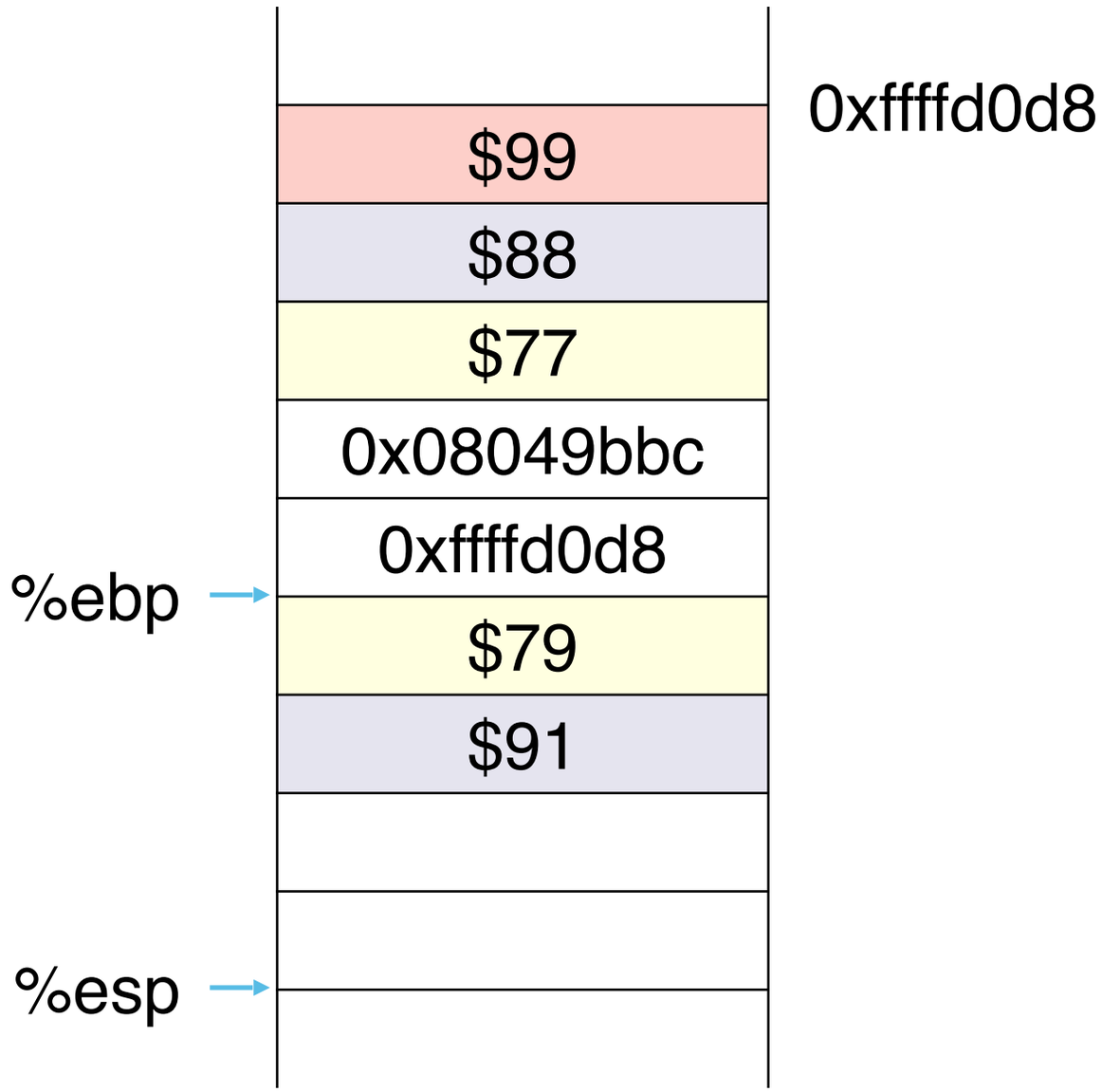
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



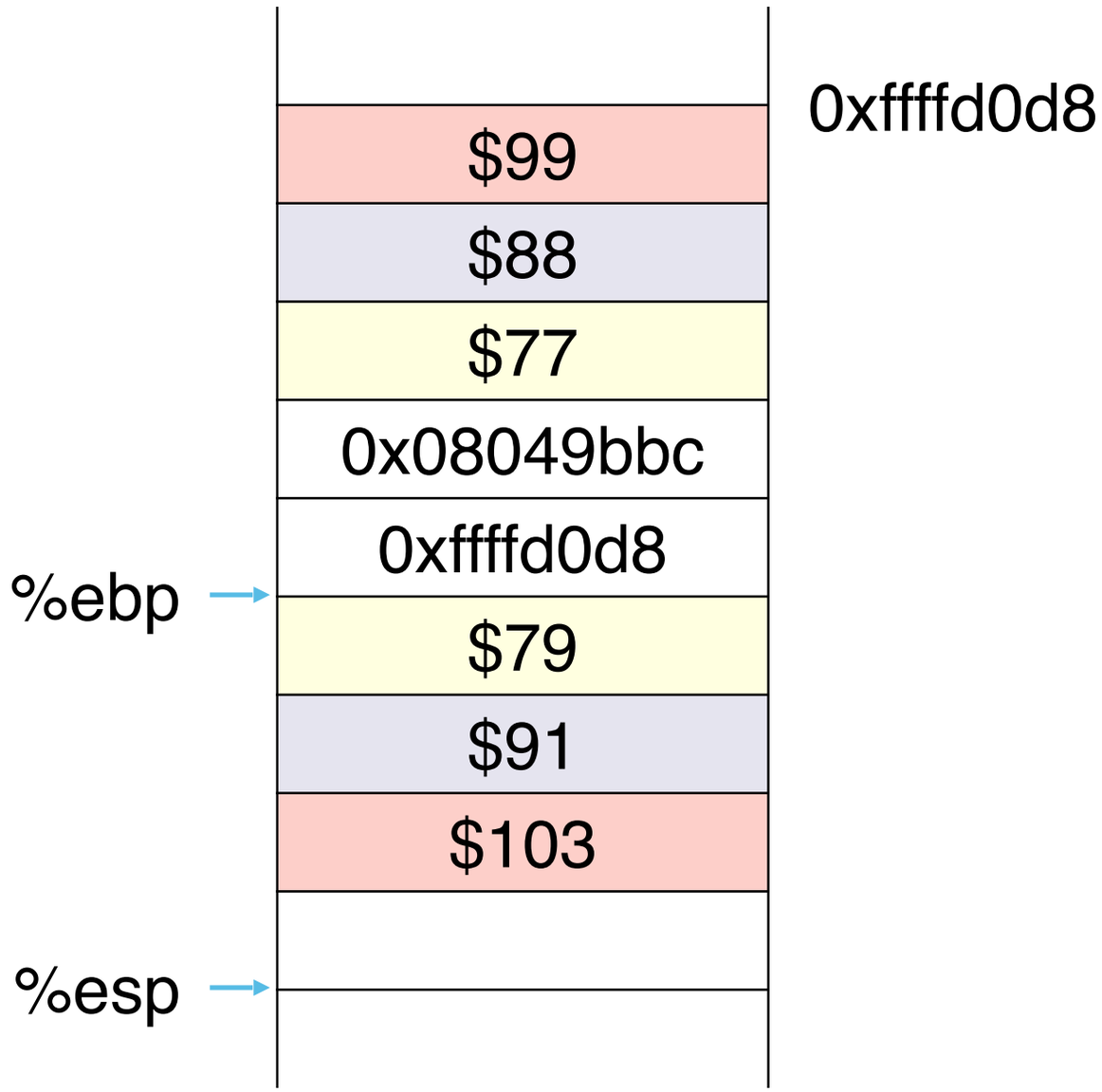
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



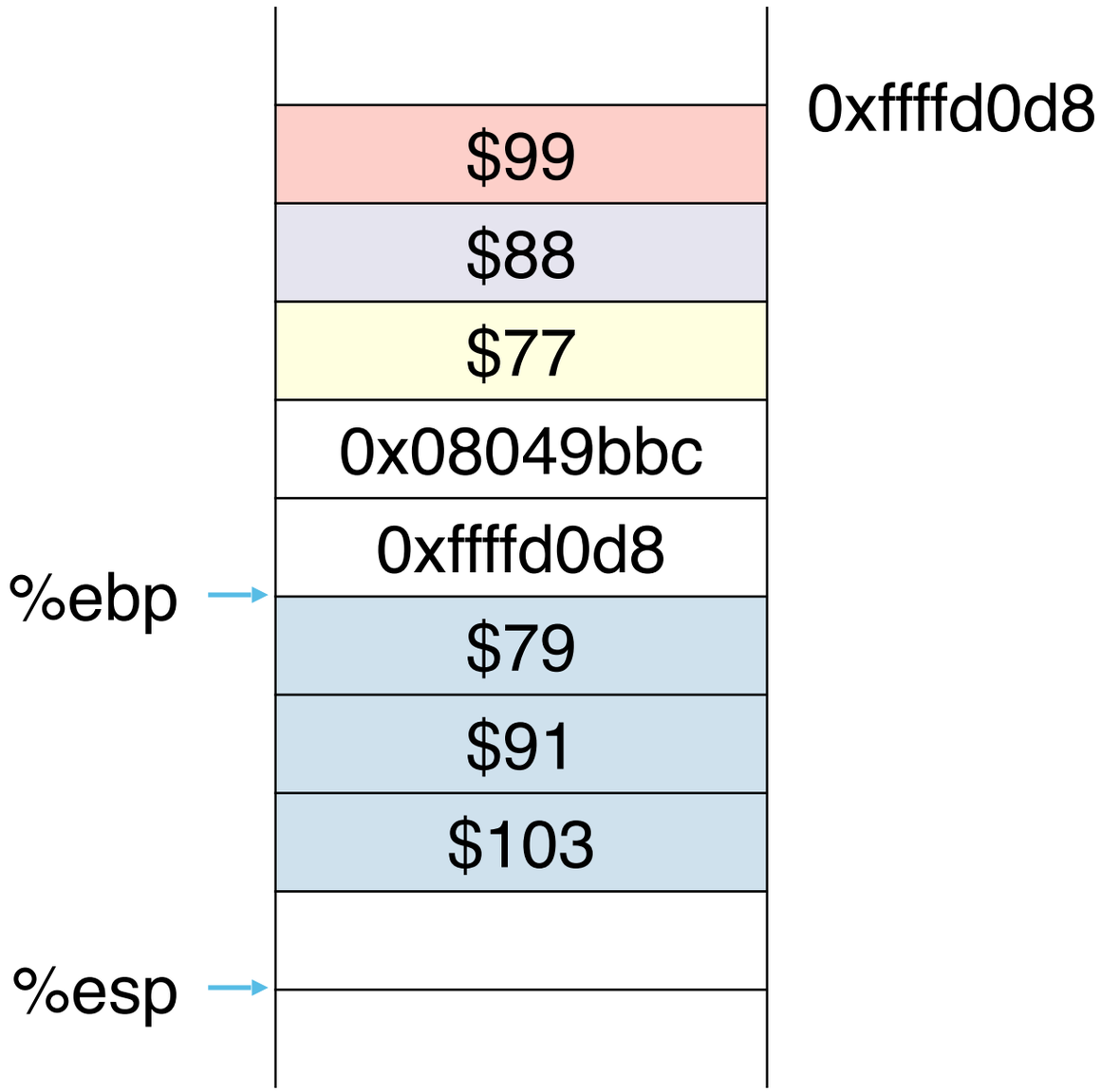
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



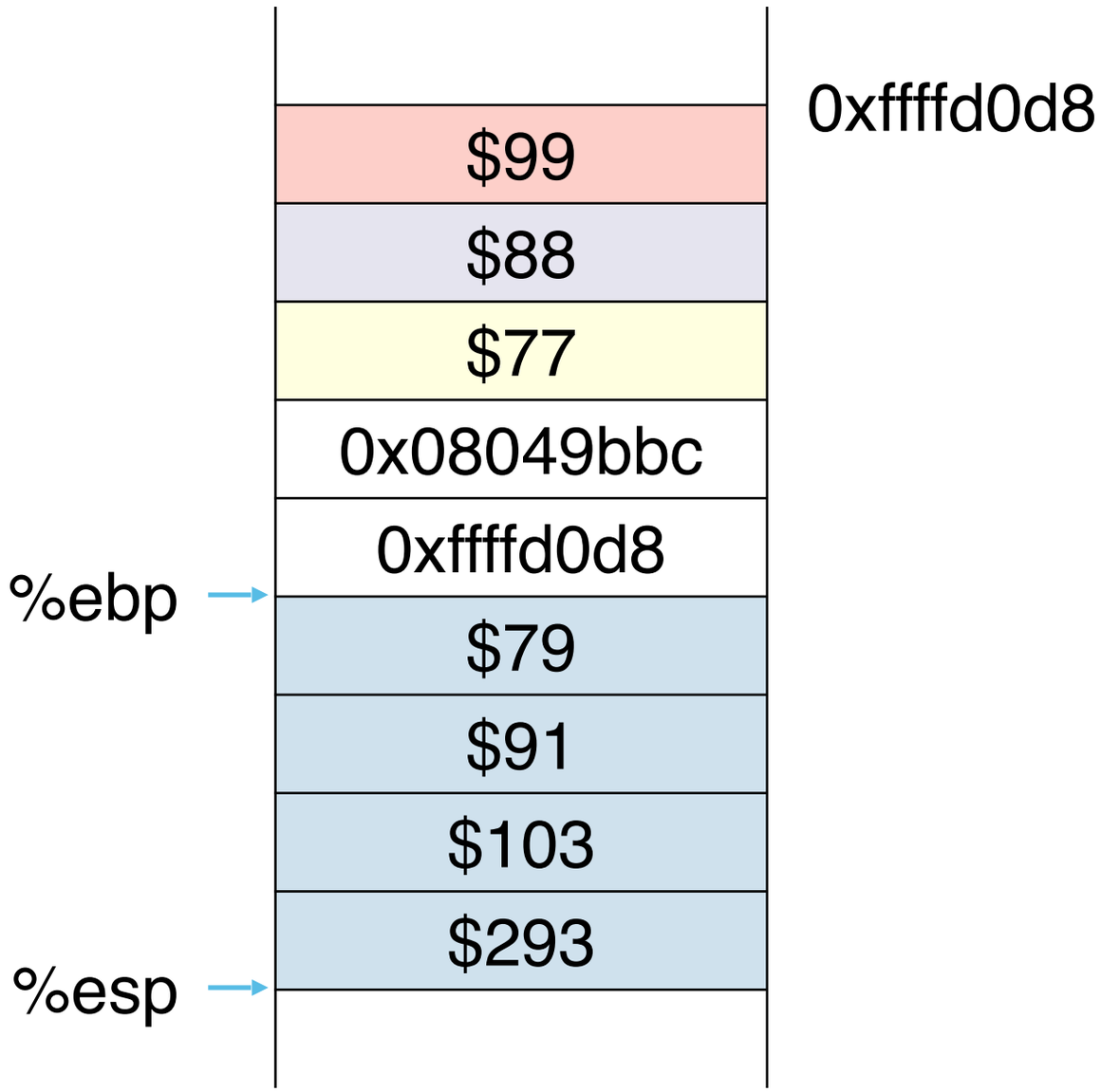
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



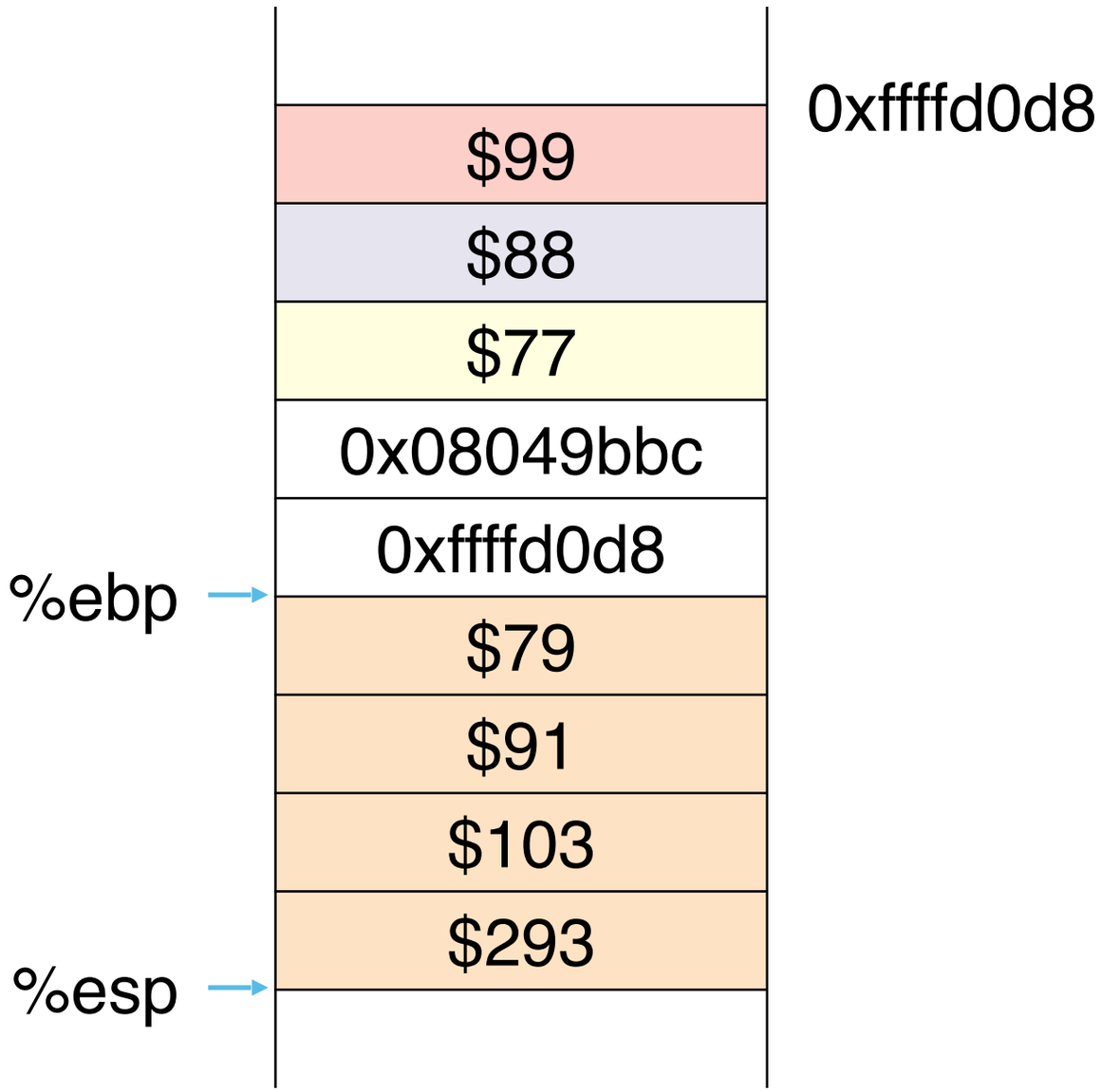
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```

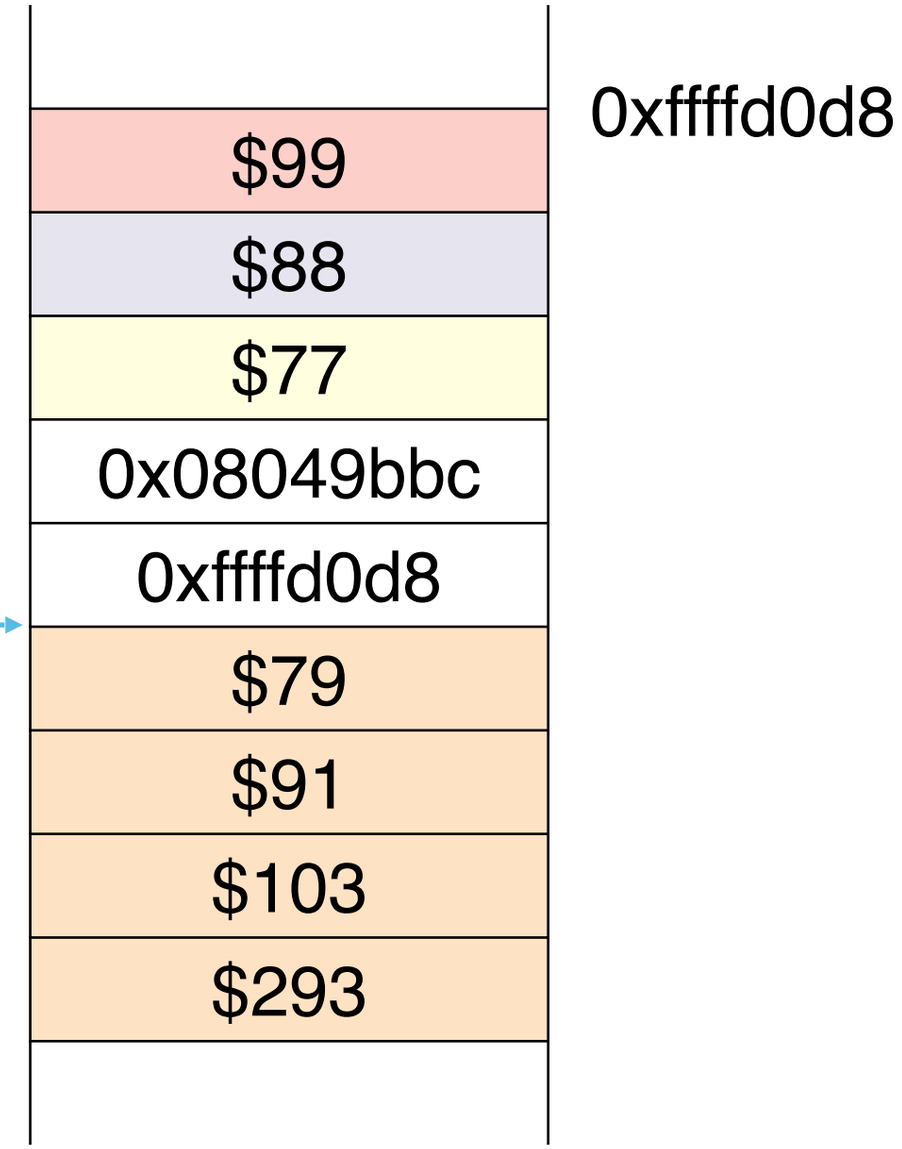


```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```

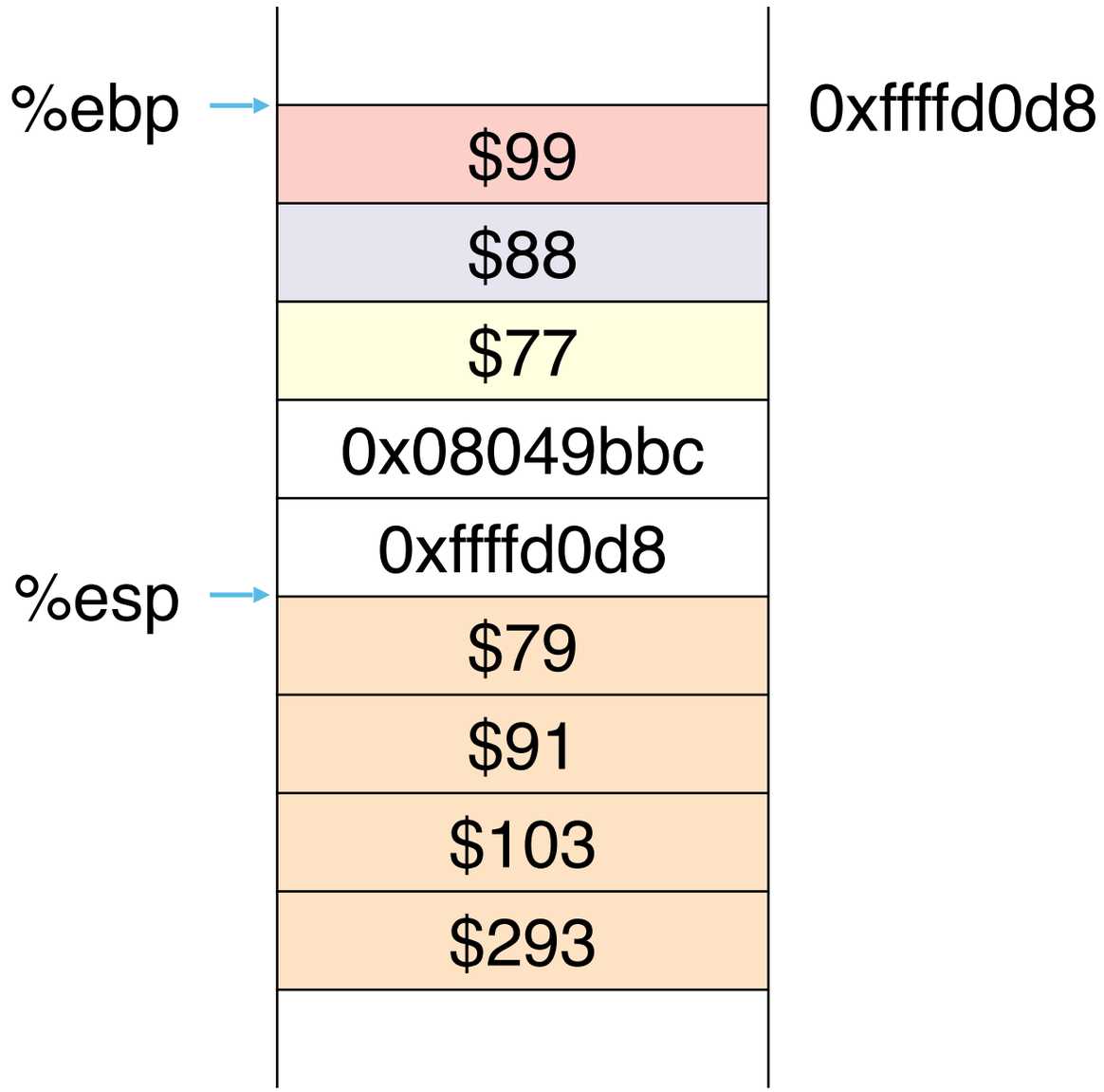


```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```

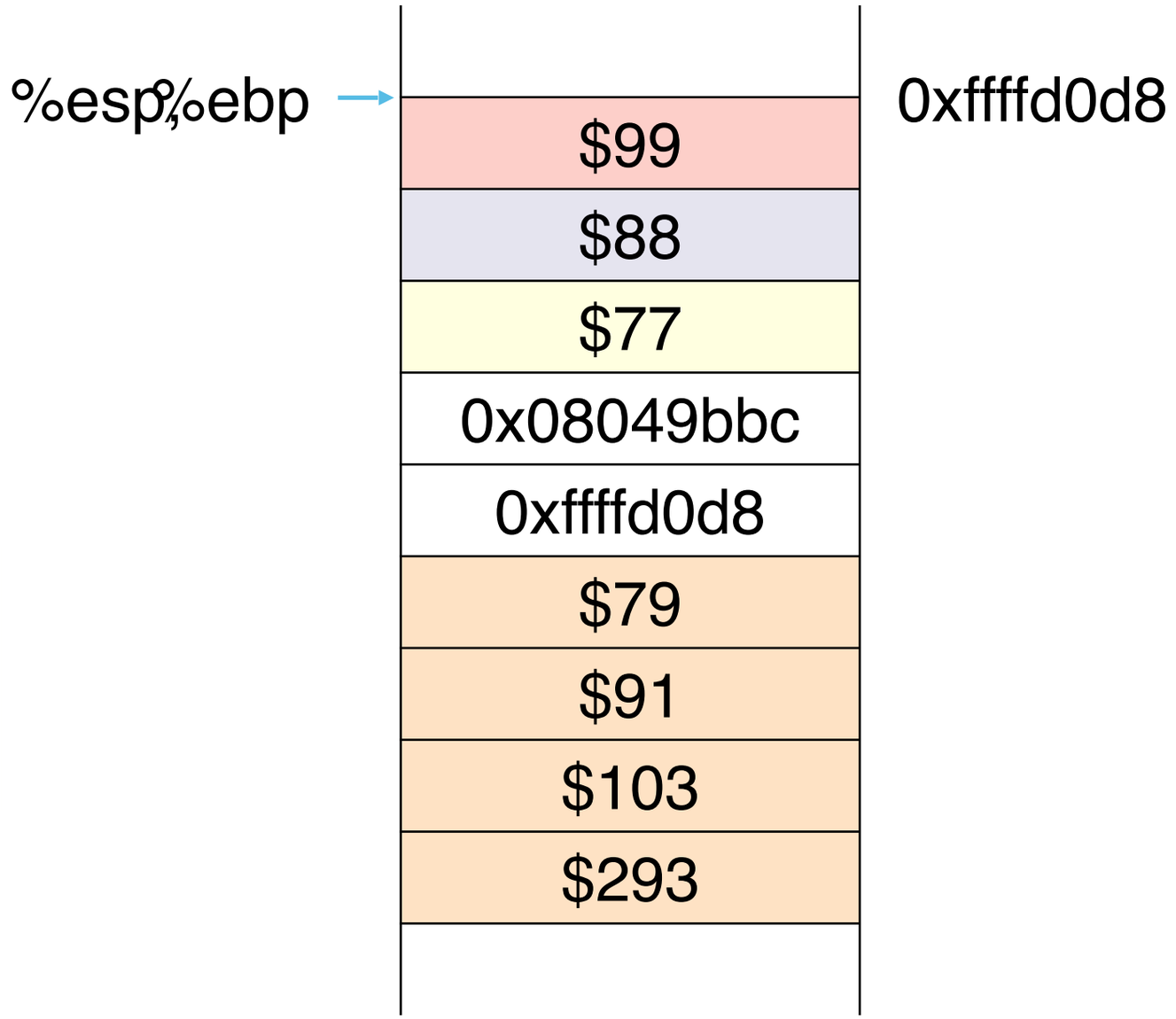
%esp%ebp →



```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



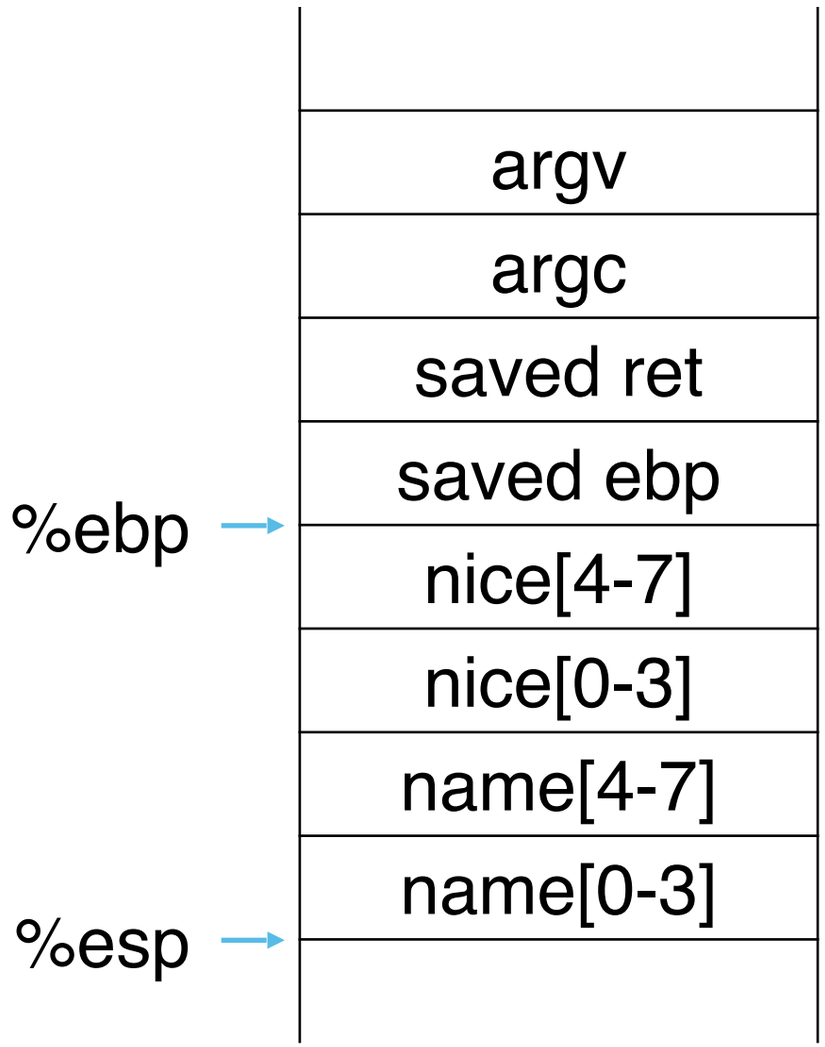
```
1   foobar(int, int, int):
2           pushl   %ebp
3           movl    %esp, %ebp
4           subl    $16, %esp
5           movl    8(%ebp), %eax
6           addl    $2, %eax
7           movl    %eax, -4(%ebp)
8           movl    12(%ebp), %eax
9           addl    $3, %eax
10          movl    %eax, -8(%ebp)
11          movl    16(%ebp), %eax
12          addl    $4, %eax
13          movl    %eax, -12(%ebp)
14          movl    -4(%ebp), %edx
15          movl    -8(%ebp), %eax
16          addl    %eax, %edx
17          movl    -12(%ebp), %eax
18          addl    %edx, %eax
19          movl    %eax, -16(%ebp)
20          movl    -4(%ebp), %eax
21          imull   -8(%ebp), %eax
22          imull   -12(%ebp), %eax
23          movl    %eax, %edx
24          movl    -16(%ebp), %eax
25          addl    %edx, %eax
26          leave
27          ret
28  main:
29          pushl   %ebp
30          movl    %esp, %ebp
31          pushl   $99
32          pushl   $88
33          pushl   $77
34          call    foobar(int, int, int)
35          addl    $12, %esp
36          nop
37          leave
38          ret
```



%eip = 0x08049bbc

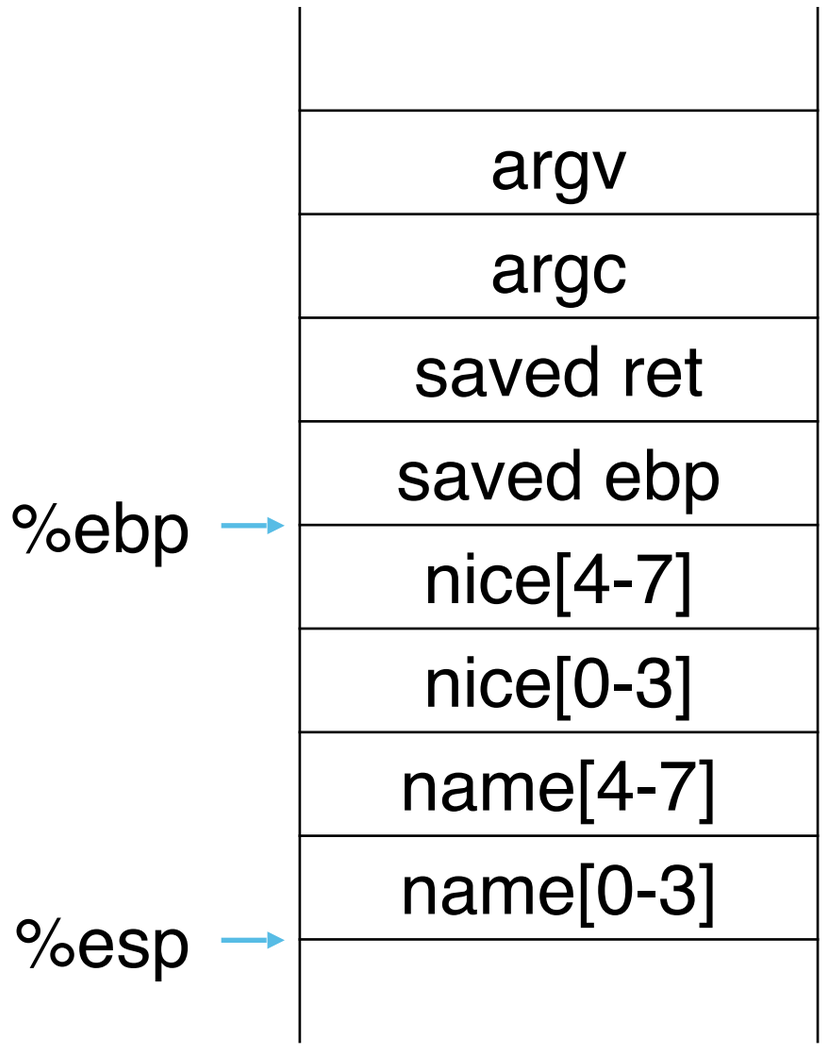
Example 1

```c
#include <stdio.h>
#include <string.h>

int main(int argc, char**argv) {
  char nice[] = "is nice.";
  char name[8];
  gets(name);
  printf("%s %s\n",name,nice);
  return 0;
}
```



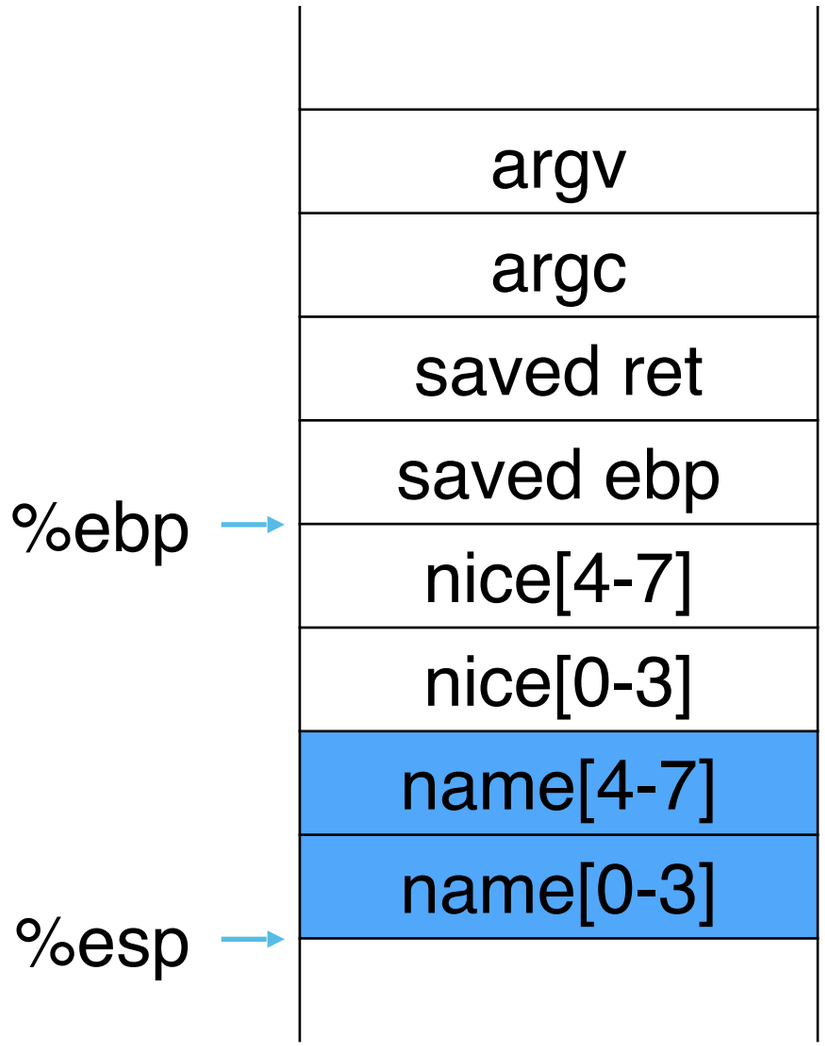
Example 1

```
#include <stdio.h>
#include <string.h>

int main(int argc, char**argv) {
  char nice[] = "is nice.";
  char name[8];
  gets(name);
  printf("%s %s\n",name,nice);
  return 0;
}
```



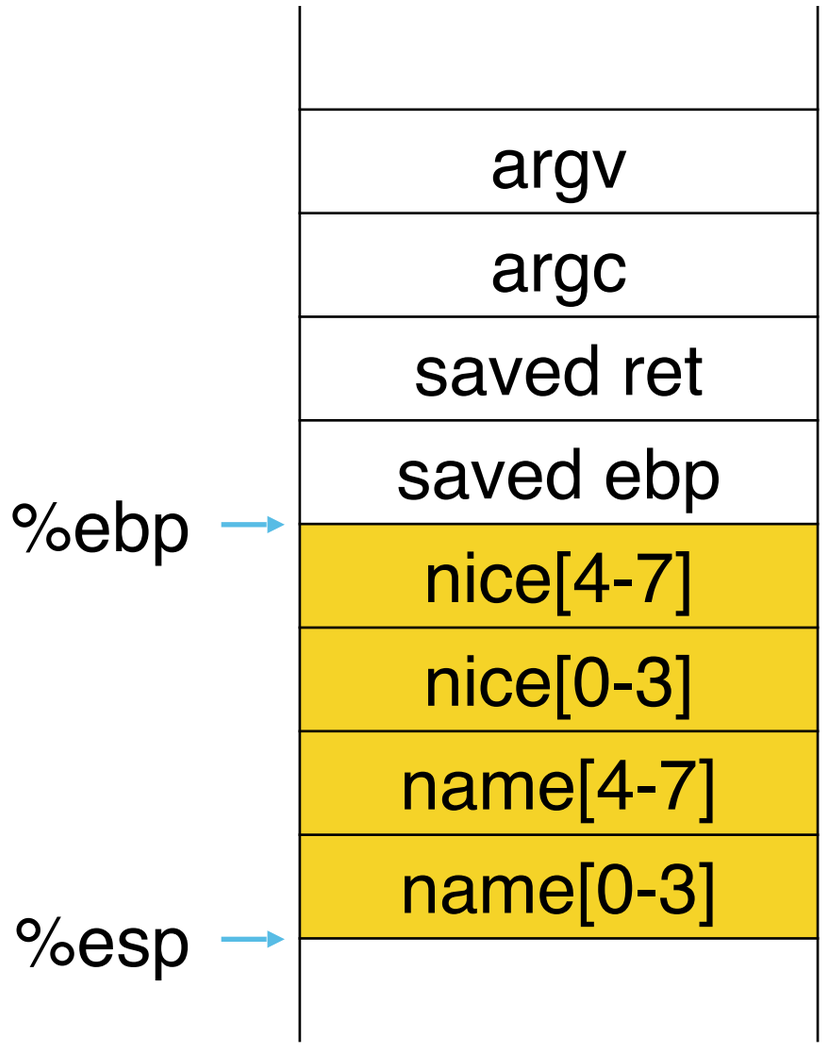
Example 1

```
#include <stdio.h>
#include <string.h>

int main(int argc, char**argv) {
  char nice[] = "is nice.";
  char name[8];
  gets(name);
  printf("%s %s\n",name,nice);
  return 0;
}
```



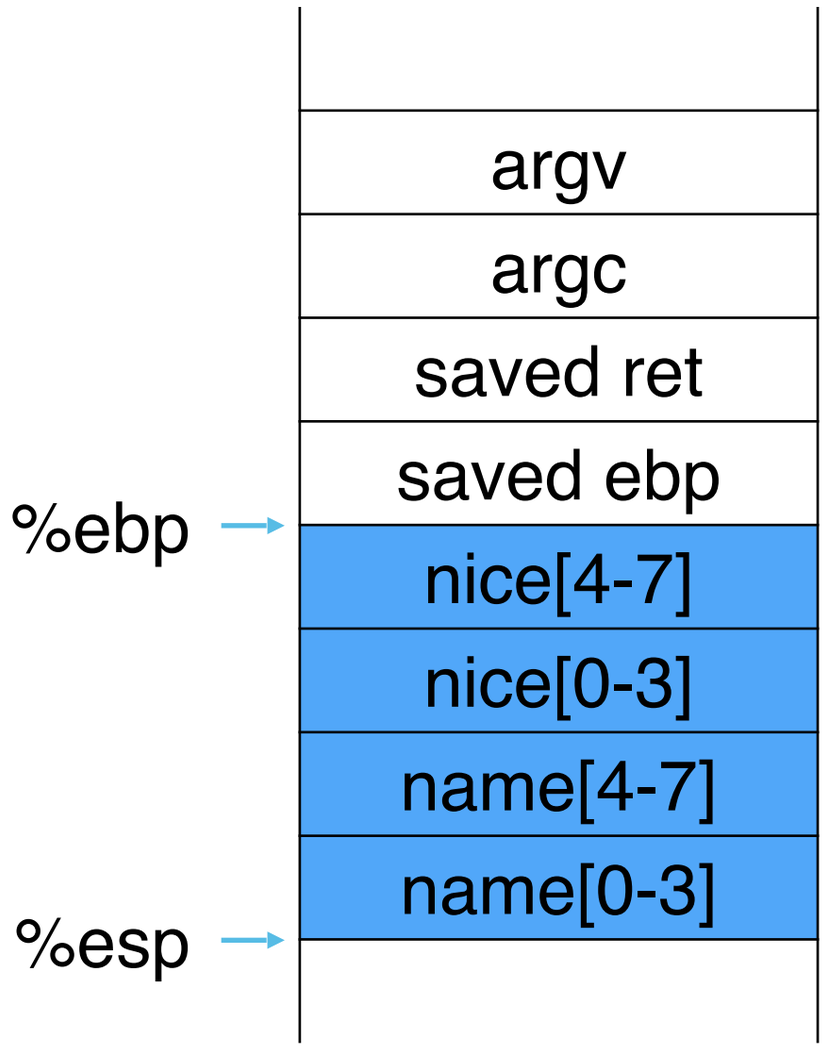
Example 1

```
#include <stdio.h>
#include <string.h>

int main(int argc, char**argv) {
  char nice[] = "is nice.";
  char name[8];
  gets(name);
  printf("%s %s\n",name,nice);
  return 0;
}
```



Example 1

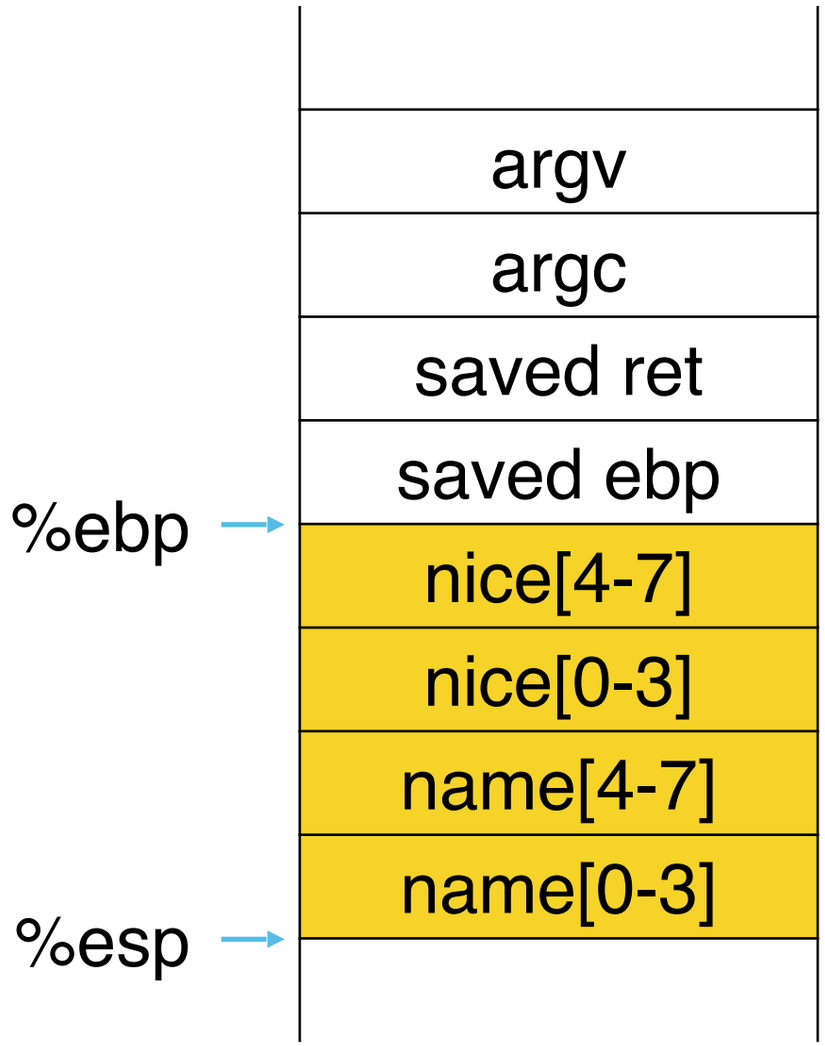
```
#include <stdio.h>
#include <string.h>

int main(int argc, char**argv) {
  char nice[] = "is nice.";
  char name[8];
  gets(name);
  printf("%s %s\n",name,nice);
  return 0;
}
```



What happens if we read a long name?

Example 1

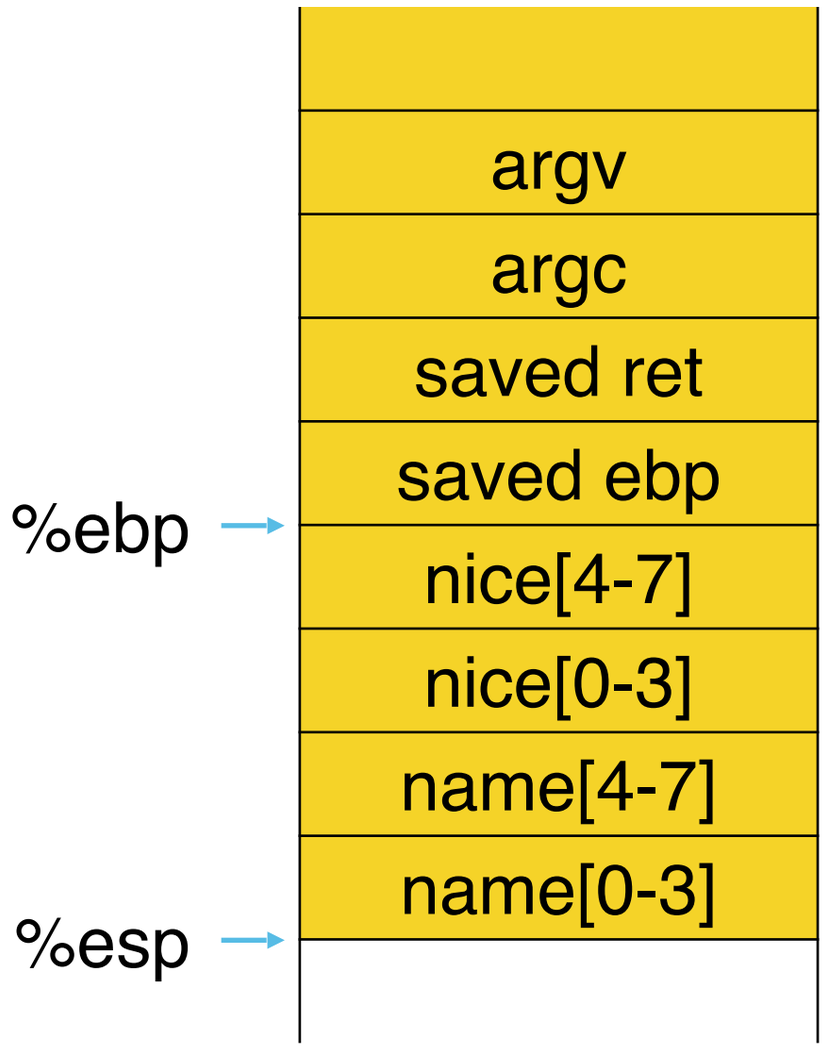
```
#include <stdio.h>
#include <string.h>

int main(int argc, char**argv) {
  char nice[] = "is nice.";
  char name[8];
  gets(name);
  printf("%s %s\n",name,nice);
  return 0;
}
```



What happens if we read a long name?

Example 1

```
#include <stdio.h>
#include <string.h>

int main(int argc, char**argv) {
  char nice[] = "is nice.";
  char name[8];
  gets(name);
  printf("%s %s\n",name,nice);
  return 0;
}
```

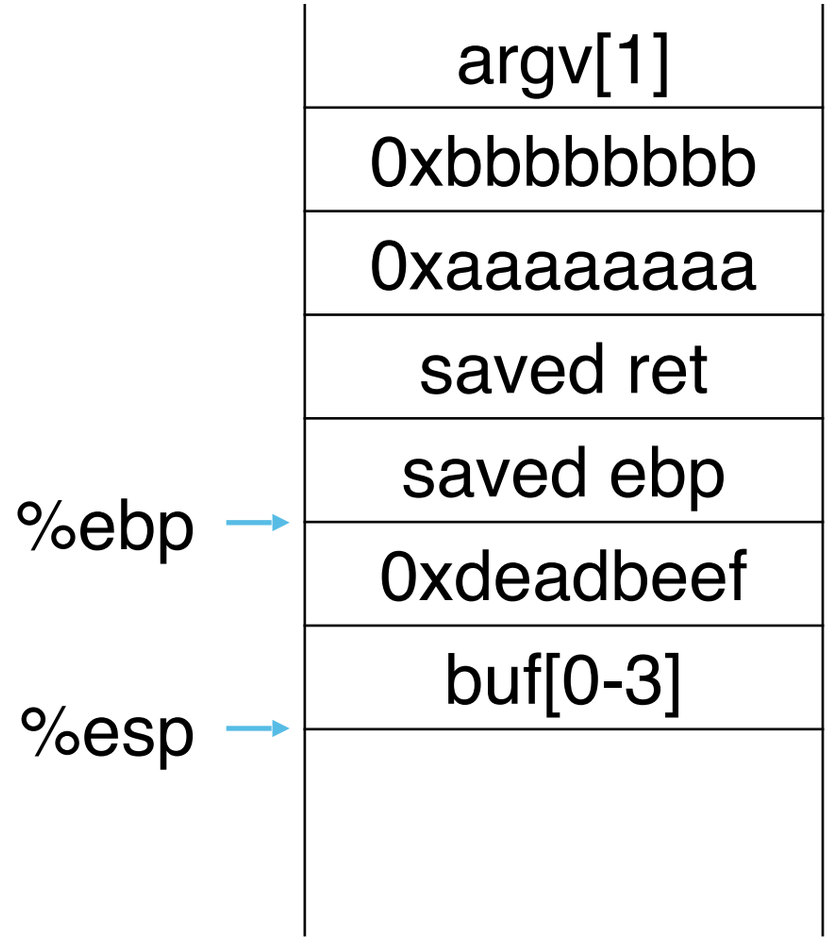


If not null terminated can read more of the stack

Let's run this program!

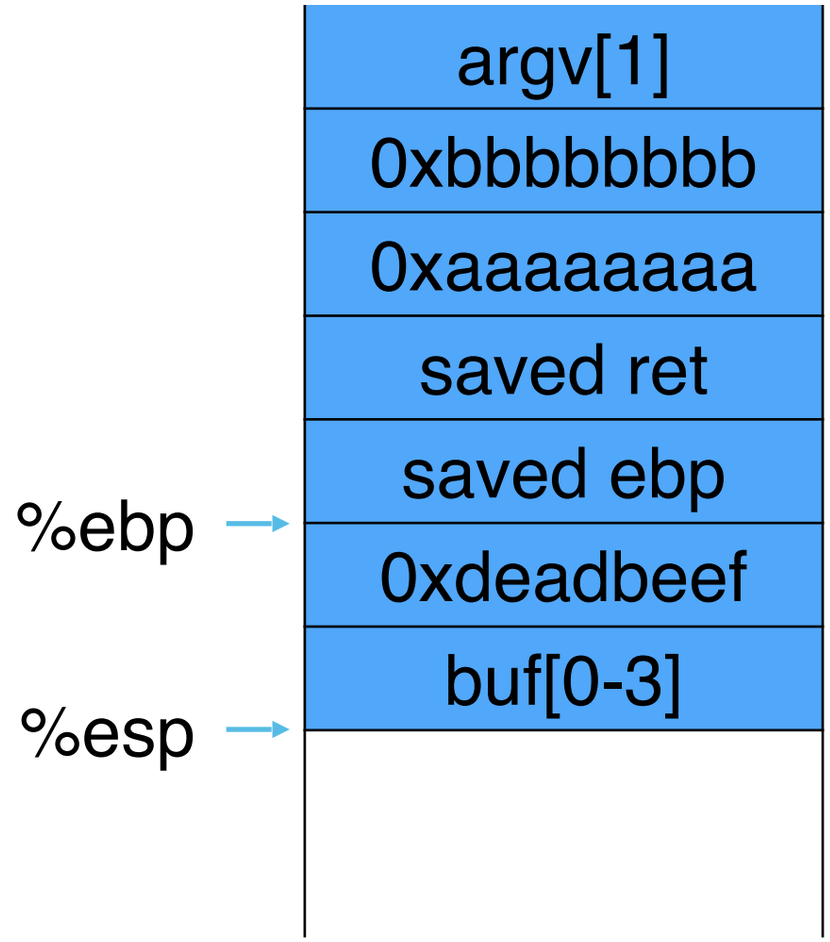
Example 2

```
#include <stdio.h>
#include <string.h>

void foo() {
  printf("hello all!!\n");
  exit(0);
}

void func(int a, int b, char *str) {
  int c = 0xdeadbeef;
  char buf[4];
  strcpy(buf,str);
}

int main(int argc, char**argv) {
  func(0xaaaaaaaa,0xbbbbbbbb,argv[1]);
  return 0;
}
```



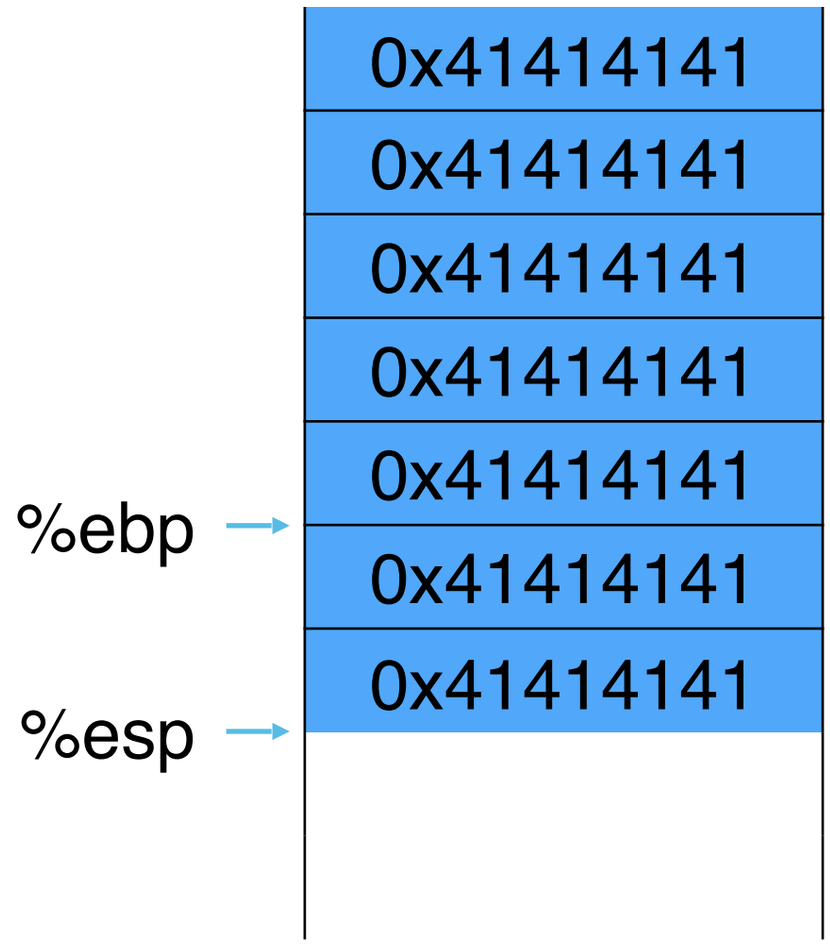
Example 2

```
#include <stdio.h>
#include <string.h>

void foo() {
  printf("hello all!!\n");
  exit(0);
}

void func(int a, int b, char *str) {
  int c = 0xdeadbeef;
  char buf[4];
  strcpy(buf,str);
}

int main(int argc, char**argv) {
  func(0xaaaaaaaa,0xbbbbbbbb,argv[1]);
  return 0;
}
```



Example 2

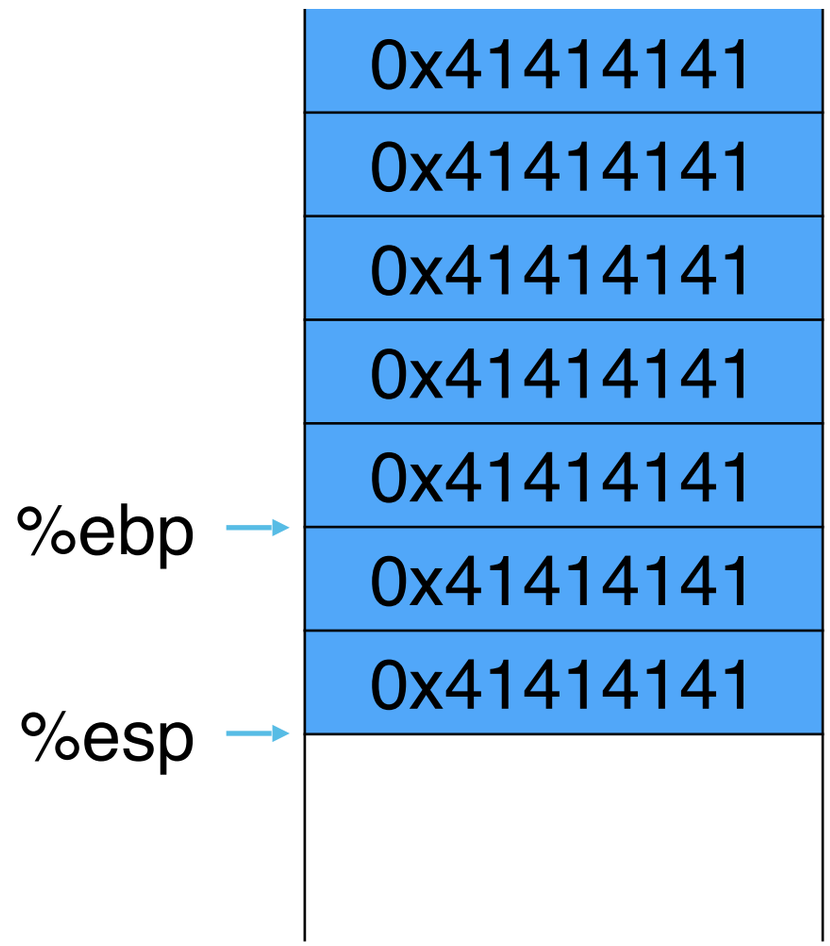
```
#include <stdio.h>
#include <string.h>

void foo() {
  printf("hello all!!\n");
  exit(0);
}

void func(int a, int b, char *str) {
  int c = 0xdeadbeef;
  char buf[4];
  strcpy(buf,str);
}

int main(int argc, char**argv) {
  func(0xaaaaaaaa,0xbbbbbbbb,argv[1]);
  return 0;
}
```



If first argument to program is “AAAAAAAA…”

Example 2

```
#include <stdio.h>
#include <string.h>

void foo() {
  printf("hello all!!\n");
  exit(0);
}

void func(int a, int b, char *str) {
  int c = 0xdeadbeef;
  char buf[4];
  strcpy(buf,str);
}

int main(int argc, char**argv) {
  func(0xaaaaaaaa,0xbbbbbbbb,argv[1]);
  return 0;
}
```



Example 2

```
#include <stdio.h>
#include <string.h>

void foo() {
  printf("hello all!!\n");
  exit(0);
}

void func(int a, int b, char *str) {
  int c = 0xdeadbeef;
  char buf[4];
  strcpy(buf,str);
}

int main(int argc, char**argv) {
  func(0xaaaaaaaa,0xbbbbbbbb,argv[1]);
  return 0;
}
```