Stack Example

a `sum( )` function

```
int sum(int x, int y) {
  return x+y;
}
```

same thing in assembly

```
int sum(int x, int y) {
  return x+y;
}
```

```
        .globl sum
        .type sum, @function
sum:
        pushl %ebp
        movl %esp, %ebp
        movl 8(%ebp), %eax
        addl 12(%ebp), %eax
        movl %esp, %ebp
        popl %ebp
        ret
```

a function that calls `sum( )`

```
        .section .text
        .globl another_func
        .type another_func, @function
another_func:
        pushl %ebp
        movl %esp, %ebp
        pushl $20
        pushl $10
        call sum
        addl $8, %esp # free stack space
                      # used by 10, 20
        movl %ebp, %esp
        popl %ebp
        ret
```

its C equivalent

```c
int another_func() {
  return sum(10, 20);
}
```

```
        .section .text
        .globl another_func
        .type another_func, @function
another_func:
        pushl %ebp
        movl %esp, %ebp
        pushl $20
        pushl $10
        call sum
        addl $8, %esp # free stack space
                      # used by 10, 20
        movl %ebp, %esp
        popl %ebp
        ret
```

main( )

We use a very simple main( )

```
void another_func();

int main(void) {
  another_func();
  return 0;
}
```

in another_func( ): 0x08048402

Before calling a function, push the arguments on the stack.

```
   0x080483ff <+0>:     push   %ebp
   0x08048400 <+1>:     mov    %esp,%ebp
=> 0x08048402 <+3>:     push   $0x14
   0x08048404 <+5>:     push   $0xa
   0x08048406 <+7>:     call   0x8048412 <sum>
   0x0804840b <+12>:    add    $0x8,%esp
   0x0804840e <+15>:    mov    %ebp,%esp
   0x08048410 <+17>:    pop    %ebp
   0x08048411 <+18>:    ret
```

EIP	0x08048402
ESP	0xffffd5f8
EBP	0xffffd5f8

stack	0xffffd5f8	0xffffd608

in another_func( ): 0x08048404

```
   0x080483ff <+0>:    push   %ebp
   0x08048400 <+1>:    mov    %esp,%ebp
   0x08048402 <+3>:    push   $0x14
=> 0x08048404 <+5>:    push   $0xa
   0x08048406 <+7>:    call   0x8048412 <sum>
   0x0804840b <+12>:   add    $0x8,%esp
   0x0804840e <+15>:   mov    %ebp,%esp
   0x08048410 <+17>:   pop    %ebp
   0x08048411 <+18>:   ret
```

EIP	0x08048404
ESP	0xffffd5f4
EBP	0xffffd5f8

stack

0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in another_func( ): 0x08048406

```
   0x080483ff <+0>:     push   %ebp
   0x08048400 <+1>:     mov    %esp,%ebp
   0x08048402 <+3>:     push   $0x14
   0x08048404 <+5>:     push   $0xa
=> 0x08048406 <+7>:     call   0x8048412 <sum>
   0x0804840b <+12>:    add    $0x8,%esp
   0x0804840e <+15>:    mov    %ebp,%esp
   0x08048410 <+17>:    pop    %ebp
   0x08048411 <+18>:    ret
```

EIP	0x08048406
ESP	0xffffd5f0
EBP	0xffffd5f8

stack

0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in another_func( ): 0x08048406

Calling the function does two things:

1. pushes the return address onto the stack
2. copies the address of the function into EIP

```
   0x080483ff <+0>:    push   %ebp
   0x08048400 <+1>:    mov    %esp,%ebp
   0x08048402 <+3>:    push   $0x14
   0x08048404 <+5>:    push   $0xa
=> 0x08048406 <+7>:    call   0x8048412 <sum>
   0x0804840b <+12>:   add    $0x8,%esp
   0x0804840e <+15>:   mov    %ebp,%esp
   0x08048410 <+17>:   pop    %ebp
   0x08048411 <+18>:   ret
```

EIP	0x08048406
ESP	0xffffd5f0
EBP	0xffffd5f8

stack

0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in sum( ): 0x08048412

Calling the function does two things:

1. pushes the return address onto the stack
2. copies the address of the function into EIP

sum( )

```
=> 0x08048412 <+0>:     push   %ebp
   0x08048413 <+1>:     mov    %esp,%ebp
   0x08048415 <+3>:     mov    0x8(%ebp),%eax
   0x08048418 <+6>:     add    0xc(%ebp),%eax
   0x0804841b <+9>:     mov    %esp,%ebp
   0x0804841d <+11>:    pop    %ebp
   0x0804841e <+12>:    ret
```

EIP	0x08048412
ESP	0xffffd5ec
EBP	0xffffd5f8

stack

0xffffd5ec	0x0804840b
0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in sum( ): 0x08048413

```
   0x08048412 <+0>:    push   %ebp
=> 0x08048413 <+1>:    mov    %esp,%ebp
   0x08048415 <+3>:    mov    0x8(%ebp),%eax
   0x08048418 <+6>:    add    0xc(%ebp),%eax
   0x0804841b <+9>:    mov    %esp,%ebp
   0x0804841d <+11>:   pop    %ebp
   0x0804841e <+12>:   ret
```

EIP	0x08048413
ESP	0xffffd5e8
EBP	0xffffd5f8

stack

0xffffd5e8	0xffffd5f8
0xffffd5ec	0x0804840b
0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in sum( ): 0x08048415

```
   0x08048412 <+0>:    push   %ebp
   0x08048413 <+1>:    mov    %esp,%ebp
=> 0x08048415 <+3>:    mov    0x8(%ebp),%eax
   0x08048418 <+6>:    add    0xc(%ebp),%eax
   0x0804841b <+9>:    mov    %esp,%ebp
   0x0804841d <+11>:   pop    %ebp
   0x0804841e <+12>:   ret
```

EIP	0x08048415
ESP	0xffffd5e8
EBP	0xffffd5e8

stack

0xffffd5e8	0xffffd5f8
0xffffd5ec	0x0804840b
0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in sum( ): 0x08048418

```
   0x08048412 <+0>:    push   %ebp
   0x08048413 <+1>:    mov    %esp,%ebp
   0x08048415 <+3>:    mov    0x8(%ebp),%eax
=> 0x08048418 <+6>:    add    0xc(%ebp),%eax
   0x0804841b <+9>:    mov    %esp,%ebp
   0x0804841d <+11>:   pop    %ebp
   0x0804841e <+12>:   ret
```

EIP	0x08048418
ESP	0xffffd5e8
EBP	0xffffd5e8

stack

0xffffd5e8	0xffffd5f8
0xffffd5ec	0x0804840b
0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in sum( ): 0x0804841b

```
   0x08048412 <+0>:    push   %ebp
   0x08048413 <+1>:    mov    %esp,%ebp
   0x08048415 <+3>:    mov    0x8(%ebp),%eax
   0x08048418 <+6>:    add    0xc(%ebp),%eax
=> 0x0804841b <+9>:    mov    %esp,%ebp
   0x0804841d <+11>:   pop    %ebp
   0x0804841e <+12>:   ret
```

EIP	0x0804841b
ESP	0xffffd5e8
EBP	0xffffd5e8

stack

0xffffd5e8	0xffffd5f8
0xffffd5ec	0x0804840b
0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in sum( ): 0x0804841d

```
   0x08048412 <+0>:    push   %ebp
   0x08048413 <+1>:    mov    %esp,%ebp
   0x08048415 <+3>:    mov    0x8(%ebp),%eax
   0x08048418 <+6>:    add    0xc(%ebp),%eax
   0x0804841b <+9>:    mov    %esp,%ebp
=> 0x0804841d <+11>:   pop    %ebp
   0x0804841e <+12>:   ret
```

EIP	0x0804841d
ESP	0xffffd5e8
EBP	0xffffd5e8

stack

0xffffd5e8	0xffffd5f8
0xffffd5ec	0x0804840b
0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in sum( ): 0x0804841e

```
   0x08048412 <+0>:    push   %ebp
   0x08048413 <+1>:    mov    %esp,%ebp
   0x08048415 <+3>:    mov    0x8(%ebp),%eax
   0x08048418 <+6>:    add    0xc(%ebp),%eax
   0x0804841b <+9>:    mov    %esp,%ebp
   0x0804841d <+11>:   pop    %ebp
=> 0x0804841e <+12>:   ret
```

EIP	0x0804841e
ESP	0xffffd5ec
EBP	0xffffd5f8

stack

0xffffd5ec	0x0804840b
0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in another_func( ): 0x0804840b

```
   0x080483ff <+0>:     push   %ebp
   0x08048400 <+1>:     mov    %esp,%ebp
   0x08048402 <+3>:     push   $0x14
   0x08048404 <+5>:     push   $0xa
   0x08048406 <+7>:     call   0x8048412 <sum>
=> 0x0804840b <+12>:    add    $0x8,%esp
   0x0804840e <+15>:    mov    %ebp,%esp
   0x08048410 <+17>:    pop    %ebp
   0x08048411 <+18>:    ret
```

EIP	0x0804840b
ESP	0xffffd5f0
EBP	0xffffd5f8

stack

0xffffd5f0	0x0000000a
0xffffd5f4	0x00000014
0xffffd5f8	0xffffd608

in another_func( ): 0x0804840e

```
   0x080483ff <+0>:     push   %ebp
   0x08048400 <+1>:     mov    %esp,%ebp
   0x08048402 <+3>:     push   $0x14
   0x08048404 <+5>:     push   $0xa
   0x08048406 <+7>:     call   0x8048412 <sum>
   0x0804840b <+12>:    add    $0x8,%esp
=> 0x0804840e <+15>:    mov    %ebp,%esp
   0x08048410 <+17>:    pop    %ebp
   0x08048411 <+18>:    ret
```

EIP	0x0804840e
ESP	0xffffd5f8
EBP	0xffffd5f8

stack		
	0xffffd5f8	0xffffd608

in another_func( ): 0x08048410

```
   0x080483ff <+0>:     push   %ebp
   0x08048400 <+1>:     mov    %esp,%ebp
   0x08048402 <+3>:     push   $0x14
   0x08048404 <+5>:     push   $0xa
   0x08048406 <+7>:     call   0x8048412 <sum>
   0x0804840b <+12>:    add    $0x8,%esp
   0x0804840e <+15>:    mov    %ebp,%esp
=> 0x08048410 <+17>:    pop    %ebp
   0x08048411 <+18>:    ret
```

EIP	0x08048410
ESP	0xffffd5f8
EBP	0xffffd5f8

stack	0xffffd5f8	0xffffd608

in another_func( ): 0x08048411

```
   0x080483ff <+0>:    push   %ebp
   0x08048400 <+1>:    mov    %esp,%ebp
   0x08048402 <+3>:    push   $0x14
   0x08048404 <+5>:    push   $0xa
   0x08048406 <+7>:    call   0x8048412 <sum>
   0x0804840b <+12>:   add    $0x8,%esp
   0x0804840e <+15>:   mov    %ebp,%esp
   0x08048410 <+17>:   pop    %ebp
=> 0x08048411 <+18>:   ret
```

EIP	0x08048411
ESP	0xffffd5fc
EBP	0xffffd608

stack