

Exercise 1

- Draw the stack (as in the previous slide) right before `foo`'s returns.
- Assume that when `main` starts executing `SP=100`. Show the *actual* addresses for all arguments, return addresses, and local variables on the stack.

```
void foo(int a, int b, int c){
    int d;
    d = a + b;
    return;
}

int main(argc, argv){
    int e=6;
    foo(e,3,4);
}
```

High

100

96

92

80

Low

arg v
arg c
return addr
saved FP
e
f
c
b
a
return addr
FP
e
d

current FP

current SP

High

12

8

-4

0

-4

-8

-12

-16

Low

b
a
return address
old ebp ← ebp FP
esi
edi
ebx
c

```
int add (int a, int b) {
```

```
    int c;
```

```
    c = a + b;
```

```
    return c;
```

```
}
```

```
.global add
```

```
.text
```

```
add: #prolog
```

```
push %ebp      esp: top of stack
```

```
movl %esp, %ebp
```

```
push edi esi ebp ebx
```

```
subl $4, %esp
```

```
movl 12(%ebp), %ebx
```

```
addl 8(%ebp), %ebx
```

```
movl %ebx, -16(%ebp)
```

```
movl -16(%ebp), %eax  retval
```

```
addl $4, %esp
```

```
restore 3 values #pop ebx, edi, esi
```