

MAIN:

```
push ecx  
call SUB  
mov edx, eax
```

SUB:

```
push ebp  
mov ebp, esp  
mov eax, [ebp+8]  
mov esp, ebp  
pop ebp  
ret
```

EAX = 0x00000000

ECX = 0x00000001

ESP = 0xbffff77C

EBP = 0xbffff788

EIP = MAIN

Stack

ADDRESS	VALUE
0xbffff770	
0xbffff774	
0xbffff778	
0xbffff77C	0x12345678

MAIN:
push ecx
call SUB
mov edx, eax

SUB:
push ebp
mov ebp, esp
mov eax, [ebp+8]
mov esp, ebp
pop ebp
ret

EAX = 0x00000000
ECX = 0x00000001
ESP = 0xbffff778
EBP = 0xbffff788
EIP = addr of push ecx

Stack

ADDRESS	VALUE
0xbffff770	
0xbffff774	
0xbffff778	0x00000001
0xbffff77C	0x12345678

MAIN:

push ecx

call SUB

mov edx, eax

SUB:

push ebp

mov ebp, esp

mov eax, [ebp+8]

mov esp, ebp

pop ebp

ret

EAX = 0x00000000

ECX = 0x00000001

ESP = 0xbffff774

EBP = 0xbffff788

EIP = addr of call SUB

Stack

ADDRESS	VALUE
0xbffff770	
0xbffff774	addr of mov edx, eax
0xbffff778	0x00000001
0xbffff77C	0x12345678

MAIN:

```
push ecx  
call SUB  
mov edx, eax
```

SUB:

⇒ push ebp
mov ebp, esp
mov eax, [ebp+8]
mov esp, ebp
pop ebp
ret

EAX = 0x00000000

ECX = 0x00000001

ESP = 0xbffff770

EBP = 0xbffff788

EIP = addr of push ebp


Stack

ADDRESS	VALUE
0xbffff770	0xbffff788
0xbffff774	addr of mov edx, eax
0xbffff778	0x00000001
0xbffff77C	0x12345678

MAIN:

```
push ecx
call SUB
mov edx, eax
```

SUB:

```
push ebp
 mov ebp, esp
mov eax, [ebp+8]
mov esp, ebp
pop ebp
ret
```

EAX = 0x00000000

ECX = 0x00000001

ESP = 0xbffff770

EBP = 0xbffff770

**EIP = addr of
mov ebp, esp**

Stack

ADDRESS	VALUE
0xbffff770	0xbffff788
0xbffff774	addr of mov edx, eax
0xbffff778	0x00000001
0xbffff77C	0x12345678

MAIN:

```
push ecx  
call SUB  
mov edx, eax
```

SUB:

```
push ebp  
mov ebp, esp  
→ mov eax, [ebp+8]  
mov esp, ebp  
pop ebp  
ret
```

EAX = 0x00000001

ECX = 0x00000001

ESP = 0xbffff770

EBP = 0xbffff770

**EIP = addr of
mov eax, [ebp+8]**


Stack

ADDRESS	VALUE
0xbffff770	0xbffff788
0xbffff774	addr of mov edx, eax
0xbffff778	0x00000001
0xbffff77C	0x12345678

MAIN:

```
push ecx  
call SUB  
mov edx, eax
```

SUB:

```
push ebp  
mov ebp, esp  
mov eax, [ebp+8]  
 mov esp, ebp  
pop ebp  
ret
```

EAX = 0x00000001

ECX = 0x00000001

ESP = 0xbffff770

EBP = 0xbffff770

**EIP = addr of
mov esp, ebp**

Stack

ADDRESS	VALUE
0xbffff770	0xbffff788
0xbffff774	addr of mov edx, eax
0xbffff778	0x00000001
0xbffff77C	0x12345678

MAIN:

```
push ecx  
call SUB  
mov edx, eax
```

SUB:

```
push ebp  
mov ebp, esp  
mov eax, [ebp+8]  
mov esp, ebp  
pop ebp  
ret
```

EAX = 0x00000001

ECX = 0x00000001

ESP = 0xbffff774

EBP = 0xbffff788

EIP = addr of pop ebp

Stack

ADDRESS	VALUE
0xbffff770	0xbffff788
0xbffff774	addr of mov edx, eax
0xbffff778	0x00000001
0xbffff77C	0x12345678

MAIN:

```
push ecx  
call SUB  
mov edx, eax
```

SUB:

```
push ebp  
mov ebp, esp  
mov eax, [ebp+8]  
mov esp, ebp  
pop ebp
```

ret



EAX = 0x00000001

ECX = 0x00000001

ESP = 0xbffff778

EBP = 0xbffff788

EIP = addr of ret

Stack

ADDRESS	VALUE
0xbffff770	0xbffff788
0xbffff774	addr of mov edx, eax
0xbffff778	0x00000001
0xbffff77C	0x12345678

MAIN:

push ecx

call SUB

 **mov edx, eax**

SUB:

push ebp

mov ebp, esp

mov eax, [ebp+8]

mov esp, ebp

pop ebp

ret

EAX = 0x00000001

ECX = 0x00000001

ESP = 0xbffff778

EBP = 0xbffff788

**EIP = addr of
mov edx, eax**

Stack

ADDRESS	VALUE
0xbffff770	0xbffff788
0xbffff774	addr of mov edx, eax
0xbffff778	0x00000001
0xbffff77C	0x12345678