

When a subroutine is called, the address of the next instruction is saved into the stack

```
0x00007fffffff4c0|+0x0000: 0x00000000004000d7 → <print_hello_world+6> pop rcx ← $rsp
```

When ret is executed, the next instruction is executed which is pop rcx which has been saved to the stack

```
→ 0x4000cb <hello_world_proc+27> ret
↳ 0x4000d7 <print_hello_world+6> pop rcx
```

When pop is executed, rsp increases by 8 bytes since it is a 64 bit executable

Before pop is executed

```
0x00007fffffff4c8|+0x0000: 0x0000000000000003 ← $rsp
0x00007fffffff4d0|+0x0008: 0x0000000000000001
```

After pop is executed

```
0x00007fffffff4d0|+0x0000: 0x0000000000000001 ← $rsp
```

Before loop is executed

```
$rcx : 0x3
```

After loop is executed

```
$rcx : 0x2
```

Counter is 0:

```
$rcx : 0x0
```

Program will not loop

```
0x4000d8 <print_hello_world+7> loop 0x4000d1 <print_hello_world>
→ 0x4000da <exit+0> mov eax, 0x3c
```