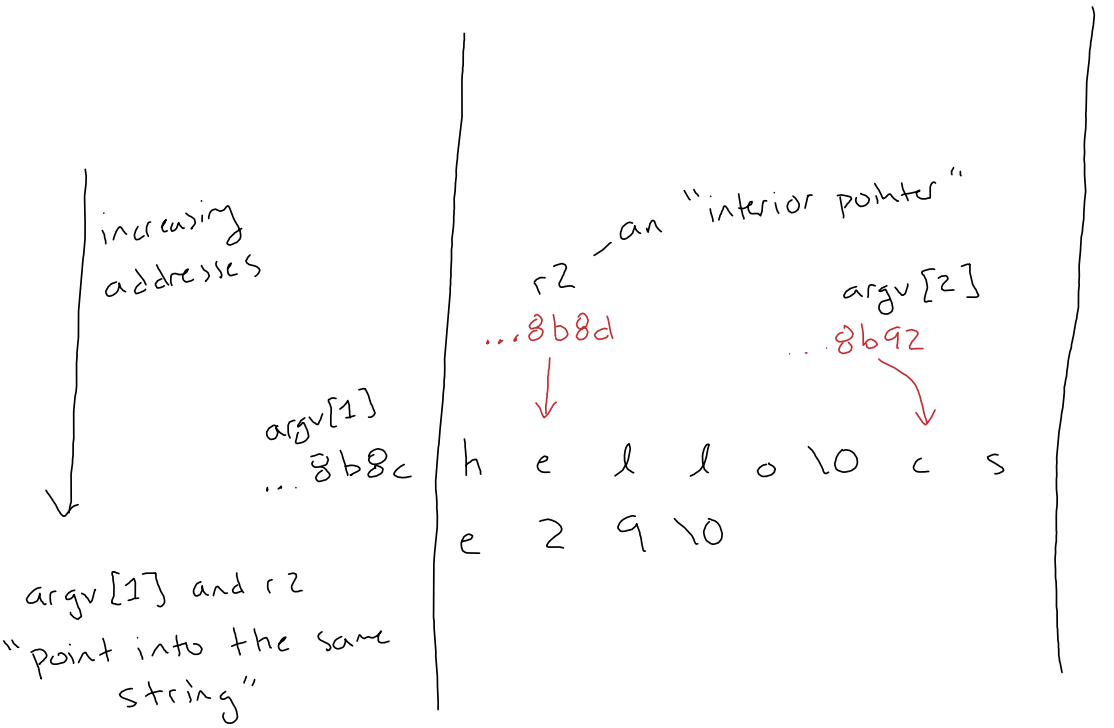
"slices" in Go
Rust

./strfunc hello csc29



./strfunc joe-had-surgery surge

1 2 3 4 5 6 7 8

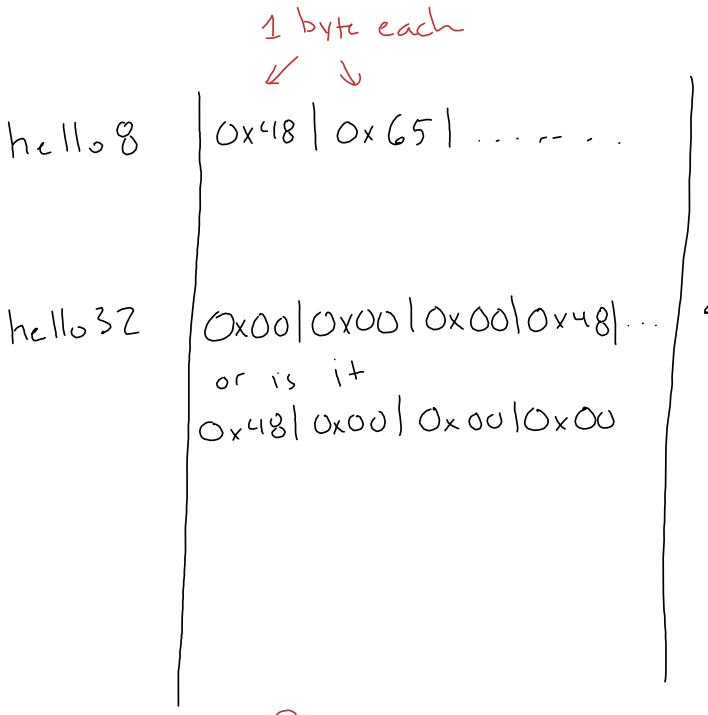
argv[1] 0x...6b82
r1 0x...6b8a ↘ 8 bytes later

```
uint32_t hello32[] = {0x48
```

gets expanded to 32 bits

0x00000048 is stored in memory

1 byte each



Big-endian

0x48000000

$4*16^7 + 8*16^6$ ....

Little-endian

0x00000048

$4*16 + 8 = 72$

"Endian-ness" how to interpret multi-byte numbers

Big-endian first byte in order is most significant

Little-endian first byte in order is least significant