



CS107, Lecture 15

Introduction to Assembly, Take II

Reading: B&O 3.1-3.4

[Ed Discussion](#)

Extra Practice 1

Fill in the blank to complete the C code that

```
int x = ...  
int *ptr = malloc(...);  
...  
___???___ = _????_;
```

1. mystery line compiles to this assembly
2. registers hold these values

```
mov %ecx, (%rax)
```

<val of x>

%ecx

<val of ptr>

%rax



Try subbing in <x> and <ptr>
with actual values, like 4
and 0x7fff80

Extra Practice 1

Fill in the blank to complete the C code that

```
int x = ...  
int *ptr = malloc(...);  
...  
___??_ = _???_;    *ptr = x;
```

```
mov %ecx, (%rax)
```

<val of x>

%ecx

<val of ptr>

%rax

Extra Practice 2

Fill in the blank to complete the C code that

1. generates this assembly
2. **results in** this register layout

```
long *arr = malloc(...);
```

```
...
```

```
long num = _____???
```

```
mov (%rdi, %rcx, 8),%rax
```

<val of num>

%rax

3

%rcx

<val of arr>

%rdi



Extra Practice 2

Fill in the blank to complete the C code that

1. generates this assembly
2. **results in** this register layout

```
long *arr = malloc(...);  
...  
long num = _____???
```

```
long num = arr[3];  
long num = *(arr + 3);  
long num = *(arr + y);
```

assume long y = 3;
declared earlier

```
mov (%rdi, %rcx, 8),%rax
```

<val of num>

%rax

3

%rcx

<val of arr>

%rdi

Extra Practice 3

Fill in the blank to complete the C code that

1. generates this assembly
2. has this register layout

```
char *str = malloc(...);  
long i = 2;  
____?___ = 'c';
```

```
movb $0x63, (%rcx,%rdx,1)
```

<val of str>

%rcx

2

%rdx



Extra Practice 3

Fill in the blank to complete the C code that

1. generates this assembly
2. has this register layout

```
char *str = malloc(...);
```

```
long i = 2;
```

```
___ ??? ___ = 'c';
```

```
str[i] = 'c';  
*(str + i) = 'c';
```

```
movb $0x63, (%rcx,%rdx,1)
```

<val of str>

%rcx

2

%rdx

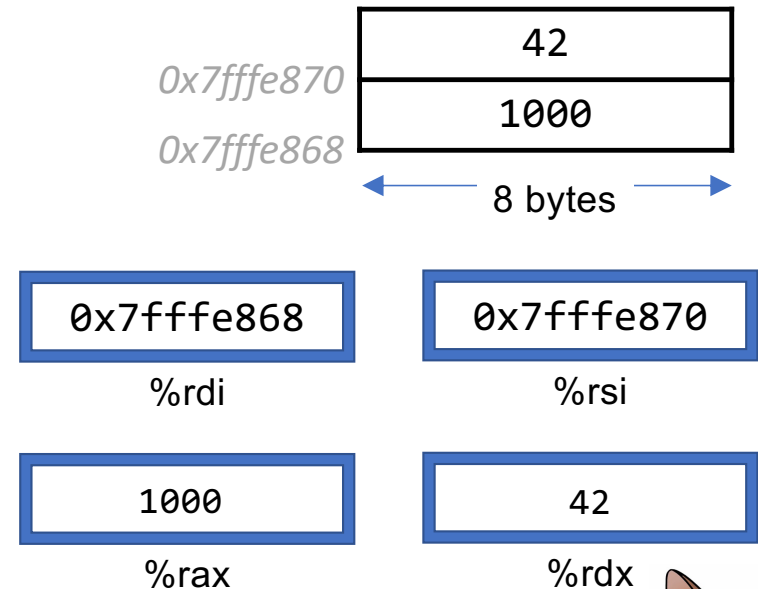
Bonus: Sneak peek into next week

- The below code is the **objdump** of a C function, **foo**.
 - foo keeps its 1st and 2nd parameters in registers **%rdi** and **%rsi**, respectively.

```
0x4005b6 <foo>      mov     (%rdi),%rax
0x4005b9 <foo+3>     mov     (%rsi),%rdx
0x4005bc <foo+6>     mov     %rdx,(%rdi)
0x4005bf <foo+9>     mov     %rax,(%rsi)
```

- What does this function do?
- What C code could have generated this assembly?

(Hints: make up C variable names as needed, assume all regs 64-bit)



Bonus: Sneak peek into next week

- The below code is the **objdump** of a C function, **foo**.
 - foo keeps its 1st and 2nd parameters in registers **%rdi** and **%rsi**, respectively.

```
0x4005b6 <foo>      mov     (%rdi),%rax
0x4005b9 <foo+3>      mov     (%rsi),%rdx
0x4005bc <foo+6>      mov     %rdx,(%rdi)
0x4005bf <foo+9>      mov     %rax,(%rsi)
```

```
void foo(long *xp, long *yp) {
    long a = *xp;
    long b = *yp;
    *yp = a;
    *xp = b;
    ...
}
```

