# CS 3: Introduction to Software Design

## Pointers Exercises

## Warmup

### fread Prototype Reminder

```
size_t fread(void *ptr, size_t size, size_t nitems, FILE *stream);
```

### Fill In The Blanks!

```
void read_one_1() {
    Char *  c = malloc(sizeof(char));
    fread( c , sizeof(char), 1, stdin);
    printf("I got: %c\n", *c );
}
```

c , *c , &c

~~void *~~ →

```
void read_one_2() {
    Char  c = 'X';
    fread(    , sizeof(char), 1, stdin);
    printf("I got: %c\n",    );
}
```

Char * t = malloc(...)

t * → t

&d

(char *)*

~~*d~~

Char

Char * c;

"Char *" ← "char" "

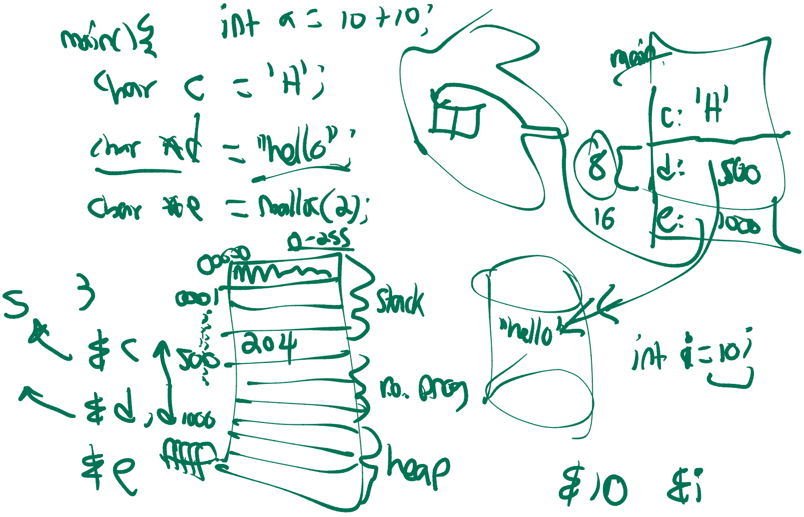
"Char &"

## More &

```
int to_nibble(char *bin) {
    char *endptr = NULL;
    char *dup = strndup(bin, 4);
    int result = strtol(dup,    , 2);
    if (    != endptr) {
        free(dup);
        return -1;
    }
    free(dup);
    return result;
}
```

```
long strtol(char *str, char **endptr, int base)
------------------------------------------------
If endptr is not NULL, strtol() stores the
address of the first invalid character in
*endptr.  If there were no digits at all,
however, strtol() stores the original value of
str in *endptr.  (Thus, if *str is not `\0' but
**endptr is `\0' on return, the entire string
was valid.)
```

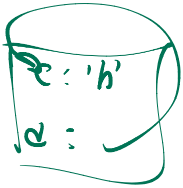
```
main(){      int a = 10+10;
    char c = 'H';
    char *d = "hello";
    char *e = malloc(2);
}
```



```
int i = 10;
```

&10   &i

```
char c = 'h';
char *d = &c;
```



```
int16_t                                        int16_t
~~char~~ *arr = malloc(sizeof(~~char~~) * 10);
```

&arr

```
int32_t *b = ....
        b = 76
```

sizeof(char)

*arr    arr[0]    *(arr + 0)

&(b[2])    arr[1]    *(arr + 1)

b = 76 (which is a heap addr)

arr[x] ≈ addr(arr) + sizeof(char) * x

$\&(b[2]) \mapsto \&*(b+2) \mapsto$ "b+2"