

CS61c Spring 2014 Discussion 1 – C

1 C Introduction

C is syntactically very similar to Java, but there are a few key differences of which to be wary:

- C is function oriented, not object oriented, so no objects for you.
- C does not automatically handle memory for you.
 - In the case of stack memory (things allocated in the “usual” way), a datum is garbage immediately after the function in which it was defined returns.
 - In the case of heap memory (things allocated with `malloc` and friends), data is freed only when the programmer explicitly frees it.
 - In any case, allocated memory always holds garbage until it is initialized.
- C uses pointers explicitly. `*p` tells us to use the value that `p` points to, rather than the value of `p`, and `&x` gives the address of `x` rather than the value of `x`.

There are other differences of which you should be aware, but this should be enough for you to get your feet wet.

2 At Least There Are Comments.

Write the following functions so that they perform according to the provided comment.

1.

```
/* The first function you write in any language.
 * Prints the string "Hello World\n" to standard output. */
void hello_world() {
```
2.

```
/* Divides and takes the floor of a value exterior to this function by 2^POW.
 * Does not use the division function. */
void div(int *y, unsigned int pow) {
```
3.

```
/* For each bit position i in [0, sizeof(int)*8) calls hello_world i times
 * iff the ith bit of the value X points to is set. */
void HI_HI_HI_HI(int *x) {
```

4.

```
/* Computes and returns the nth fibonacci number, using an iterative approach. */
int fib_iter(unsigned int n) {
```

3 Uncommented Code? Yuck!

The following functions work correctly (note, this does not mean intelligently), but have no comments. Document the code to prevent it from causing further confusion.

1.

```
/*
 *
 */
int foo(int *arr, size_t n) {
    return n ? arr[0] + foo(arr + 1, n - 1) : 0;
}
```

2.

```
/*
 *
 */
int bar(int *arr, size_t n) {
    int sum = 0, i;

    for (i = n; i > 0; i--) {
        sum += !arr[i - 1];
    }

    return ~sum + 1;
}
```

3.

```
/*
 *
 */
void baz(int x, int y) {
    x = x ^ y;
    y = x ^ y;
    x = x ^ y;
}
```

4 Programming with Pointers

Write the following functions so that they perform according to the provided comment. Not all questions are guaranteed to be soluble.

1.

```
/* Swaps the value of two ints outside of this function. */
```

2. `/* Increments the value of an int outside of this function by one. */`

3. `/* Returns a buffer for N ints. */`

4. `/* Returns the number of bytes in a string. Does not use strlen. */`

5. `/* Returns the number of elements in an array ARR of ints. */`

5 Problem?

The following code segments may contain either logic or syntax errors. Find them.

1.

```
/* Returns the sum of all the elements in SUMMANDS. */
int sum(int* summands) {
    int sum = 0;
    for (int i = 0; i < sizeof(summands); i++)
        sum += *(summands + i);
    return sum;
}
```

2.

```
/* Increments all the letters in the string STRING, held in an array of length N.
 * Does not modify any other memory which has been previously allocated. */
void increment(char* string, int n) {
    for (int i = 0; i < n; i++)
        *(string + i)++;
}
```

3.

```
/* Copies the string SRC to DST. */
void copy(char* src, char* dst) {
    while (*dst++ = *src++);
}
```