CS 61C Summer 2022 — C and Memory Management — Discussion 1

1 Pre-Check

This section is designed as a conceptual check for you to determine if you conceptually understand and have any misconceptions about this topic. Please answer true/false to the following questions, and if false, correct the statement to make it true:

1.1 True or False: C is a pass-by-value language.

1.2 The following is correct C syntax:
`int num = 43`

1.3 In compiled languages, the compile time is generally pretty fast, however the run-time is significantly slower than interpreted languages.

1.4 The correct way of declaring a character array is `char[] array`.

1.5 Bitwise and logical operations result in the same behaviour for given bitstrings.

1.6 Memory sectors are defined by the hardware, and cannot be altered.

1.7 When should you use the heap over the stack? Do they grow?

2 Memory Management

2.1 For each part, choose one or more of the following memory segments where the data could be located: **code, static, heap, stack**.

(a) Static variables

(b) Local variables

(c) Global variables

(d) Constants

(e) Machine Instructions

(f) Result of Dynamic Memory Allocation(`malloc` or `calloc`)

(g) String Literals

3 Bit-wise Operations

3.1 In C, we have a few bit-wise operators at our disposal:

- `AND` (&)
- `NOT` (∼)
- `OR` (|)
- `XOR` (∧)
- `SHIFT LEFT` (<<)
  - Example: `0b0001 << 2 = 0b0100`
- `SHIFT RIGHT` (>>)
  - Example: `0b0100 >> 2 = 0b0001`

a	**b**	**a & b**	**a | b**	**a ∧ b**	**∼a**
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

For your convenience, truth tables for the logical operators are provided above. With the binary numbers `a`, `b`, and `c` below, perform the following bit-wise operations:

```
a = 0b1000 1011
b = 0b0011 0101
c = 0b1111 0000
```

(a) `a & b`

(b) `a ∧ c`

(c) `a | 0`

(d) `a | (b >> 5)`

(e) ∼ (`(b | c) & a`)