CSM 61C Spring 2019 — Number Representation and C

Mentoring 1: January 20, 2019

1 Number Representation Warm-Up

1.1 What is the range of integers represented by a n-bit binary number? Your answers should include expressions that use 2^n.

(a) Unsigned:

(b) Two's Complement:

(c) One's Complement:

(d) Bias (with bias b):

1.2 How many unique integers can be represented in each case?

(a) Unsigned:

(b) Two's Complement:

(c) One's Complement:

(d) Bias (with bias b):

2 Memory Addresses

2.1 Consider the C code here, and assume the malloc call succeeds. Rank the following values from 1 to 5, with 1 being the least, right before bar returns. Use the memory layout from class; Treat all addresses as unsigned numbers.

```
#include <stdlib.h>

int FIVE = 5;

int bar(int x) {
    return x * x;
}

int main(int argc, char *argv[]) {
    int *foo = malloc(sizeof(int));
    if (foo) free(foo);
    bar(10); //  snapshot just before it returns
    return 0;
}
```

```
foo:   _______
&foo:  _______
FIVE:  _______
&FIVE: _______
&x:    _______
```

2.2 Consider the following C program:

```
int a = 5;
int main()
{
    int b = 0;
    char* s1 = cs61c;
    char s2[] = cs61c;
    char* c = malloc(sizeof(char) * 100);
    return 0;
};
```

For each of the following values, state the location in the memory layout where they are stored. Answer with code, static, heap, or stack.

(a) `s1`

(b) `s2`

(c) `s1[0]`

(d) `s2[0]`

(e) `c[0]`

(f) `a`

3 Linked Lists Revisited

3.1 Fill out the declaration of a singly linked linked-list node below.

```
typedef struct node {
    int value;
    _____________ next; // pointer to the next element
} sll_node;
```

3.2 Let's convert the linked list to an array. Fill in the missing code.

```
int* to_array(sll_node *sll, int size) {
    int i = 0;
    int *arr = ________________;
    while (sll) {
        arr[i] = ___________;
        sll = ___________;
        _______;
    }
    return arr;
}
```

3.3 Finally, complete the function `delete_even()` that will delete every second element of the list. For example, given the lists below:

```
Before: Node 1  Node 2  Node 3  Node 4
After: Node 1  Node 3
```

Calling `delete_even()` on the list labeled "Before" will change it into the list labeled "After". All list nodes were created via dynamic memory allocation.

```
void delete_even(sll_node *s11) {
    sll_node *temp;
    if (!sll || !sll->next) {
        return;
    }
    temp = ________;
    sll->next = ________;
    free(________);
    delete_even(________);
}
```