CS107 Midterm, Winter 2022, Solutions

Problem 1: Bits and Bytes

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>

void print_binary(unsigned int num, int left_bit, int right_bit);

unsigned int bits_range(unsigned int num, int left_bit, int right_bit) {
    // bits are numbered from right to left:
    // bit 0 is the least significant, and bit 31 is the most significant
    // left_bit is the left-most bit in the number to include
    // right_bit is the right-most bit in the number to include

    unsigned int ALL_ONES = ~0;
    unsigned int shifted = num >> right_bit;

    // mask out bits above left_bit - right_bit
    unsigned int mask = ALL_ONES >> (32 - (left_bit - right_bit + 1));
    return shifted & mask;
}
```

Problem 2: Strings, pointers, and copying memory

```
void bracket_string(char *s, const char *substr, const char *bracket)
{
    size_t substr_len = strlen(substr);
    size_t bracket_len = strlen(bracket);

    char *substr_loc = strstr(s, substr);

    char *s_loc = substr_loc;
    if (s_loc) {
        // copy after the substring to the end
        // (otherwise we will copy over it)
        // must use memmove because the strings overlap
        size_t end_len = s + strlen(s) - s_loc + 1;
        memmove(s_loc + substr_len + 2 * bracket_len,
                s_loc + substr_len,
                end_len);

        // copy the bracket into str
        strcpy(s_loc, bracket);

        // copy substr
        s_loc += bracket_len;
        strcpy(s_loc, substr);

        // copy second bracket but don't null-terminate
        s_loc += substr_len;
        strncpy(s_loc, bracket, bracket_len);
    }
}
```

Problem 3: Dynamic memory management

```
void print_longest_line(FILE *file_pointer) {
    /* print out the longest line in a file, keeping track of only
     * a single line at a time. If there are no lines in the file
     * don't print anything.
     */
    char *longest = read_line(file_pointer);
    if (longest == NULL) {
        return;
    }
    char *line;
    while ((line = read_line(file_pointer)) != NULL) {
        if (strlen(longest) <= strlen(line)) {
            free(longest);
            longest = line;
        } else {
            free(line);
        }
    }
    printf("%s\n", longest);
    free(longest);
}
```

Problem 4: Generics and function pointers

```
typedef struct point {
    int x;
    int y;
} point;

void generic_sum(void *arr, int nelems, int width, void *total, void
(*add)(const void *, void *)) {
    for (int i = 0; i < nelems; i++) {
        void *elem = (char *)arr + i * width;
        add(elem, total);
    }
}

void add_char(const void *element, void *total) {
    char value = *(char *)element;
    *(int *)total += value;
}

void add_point_max(const void *element, void *total) {
    const point *p = element;
    int max;
    if (p->x > p->y) {
        max = p->x;
    } else {
        max = p->y;
    }
    *(int *)total += max;
}
```