This exam is open book and open notes. The test has four (4) questions. You should spend no more than 12 minutes per question for the 50-minute exam. Put your name on every page, and write out and sign the Honor Code pledge before turning in the test.

```
I pledge my honor that I have not violated the Honor Code during this examination.
```

<table>
<thead>
<tr>
<th>Question</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25 pts</td>
</tr>
<tr>
<td>2</td>
<td>25 pts</td>
</tr>
<tr>
<td>3</td>
<td>25 pts</td>
</tr>
<tr>
<td>4</td>
<td>25 pts</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
</tr>
</tbody>
</table>
QUESTION 1: Itsy Bitsy (25 POINTS)

1a) What does `printf("%d", 64 & 63)` print to stdout? (5 points)

1b) What decimal integer has an eight-bit two’s complement representation of 1111110? (5 points)

1c) What is the largest unsigned integer that can be represented in eight bits? Please give your answer as a base-8 (octal) number. (5 points)

1d) What does `printf("%d", (0532 << 3)/64)` print to stdout? (5 points)

1e) What does `printf("%d", ~(76 ^ 15))` print to stdout? (5 points)
QUESTION 2: Bugsy (25 POINTS)

Assume that all necessary header files (e.g., stdio.h, stdlib.h, string.h, and assert.h) and a main() function have been included.

2a) This function should return true (1) if the string “dog” appears anywhere within the input string s (and false, 0, otherwise). Identify the bug in the code and give an example input string where the bug would manifest itself. (6 points)

```c
int findDog(char *s) {
    char *dog = "dog";

    for (; *s != \0; s++)
        if (*dog == *s)
            return 1;

    return 0;
}
```

2b) This function should print every other character of the input (i.e., for an input of "0123...", the output should be "13..."). When does it produce the wrong answer? Rewrite the code to fix the bug. (6 points)

```c
void q2b(void) {
    while (getchar() != EOF)
        putchar(getchar());
}
```
2c) This function should return the *maximum value in an array `a` of `n` integers*. When does this code return the wrong value? Modify the code to correct the bug. (6 points)

```c
int q2c(int *a, int n) {
    int currmax = 0, i;
    assert(a != NULL);
    assert(n > 0);
    for (i = 0; i < n; i++)
        if (a[i] > currmax)
            currmax = a[i];
    return currmax;
}
```

2d) This function should print the differences between consecutive elements in an array `a` of `n` integers. How would the bug manifest itself? Find and fix the bug in this function. (7 points)

```c
void q2d(int *a, int n) {
    int i;

    for (i = 0; i < n; i++)
        printf("%d ", a[i+1] - a[i]);
    printf("\n");
}
```
QUESTION 3: That Thing That You Do (25 POINTS)

Answer the following questions in one short phrase --- long answers will have points deducted.

3a) State concisely how this function changes the string s. (6 pts)

```c
void q3a(char *s) {
    char *p;
    char *q;
    char c;
    int len = strlen(s);

    for (p = s, q = s + len - 1; p < q; p++, q-- ) {
        c = *p;
        *p = *q;
        *q = c;
    }
}
```

3b) State concisely what this C function returns, assuming the condition in the assert() statement is always true. (7 pts)

```c
int q3b(char *s) {
    int i = 0;

    while (*s != '\0') {
        assert(('0' <= *s) && (*s <= '9'));
        i = 10 * i + (*s++ - '0');
    }

    return i;
}
```
3c) Assume the input to the function is a string with a first and last name, separated by a single space. What does the function print for the argument “Alan Turing”, and what does the function do? (6 pts)

```c
void q3c(char *s) {
    int i = 1;
    while (s[i] != ' ')
        i++;
    printf("%s, %c\n", s+i+1, *s);
}
```

3d) State concisely what binary inputs (and only those binary inputs) this four-state DFA accepts. Assume all input characters are either ‘0’ or ‘1’, and that the leftmost state is the initial state and the rightmost state is the only accept state. (6 pts)
QUESTION 4: Stacking (25 POINTS)

This question concerns a generic stack module. The stack.h file is listed below.

```c
typedef struct Stack *Stack_T;
Stack_T Stack_new(void);
void    Stack_free(Stack_T s);
int     Stack_push(Stack_T s, const void *item);
void   *Stack_top(Stack_T s);
void   *Stack_pop(Stack_T s);
int     Stack_isEmpty(Stack_T s);
int     Stack_areEqual(Stack_T s1, Stack_T s2,
                         int (*cmp)(const void *item1, const void *item2));
```

4a) What is the difference between the definition and the declaration of an identifier such as “struct Stack”? (4 pts)

4b) Suppose the implementation stack.c changes the representation of a stack from an array to a linked list. Does the client software client.c (that has a “#include stack.h”) need to be compiled again to generate client.o? Why or why not? (2 pts)
4c) The `Stack_push()` function has an item parameter of type “`const void *`”. Why is a `void` pointer used rather than (say) an `int` or `char` pointer? Why is “`const`” used? (4 pts)

4d) The `Stack_pop()` and `Stack_top()` functions return the top item in the stack, with or without removing that item, respectively. Why do these functions have a return type of “`void *`” rather than “`const void *`”? (3 pts)
4e) The `Stack_areEqual()` function in `stack.c` relies on the client to provide a comparison function `*cmp()`. Why can’t `Stack_areEqual()` simply consist of `return(s1 == s2)`? (5 pts)

4f) `Stack_top()` returns a pointer to the top item in the stack, leaving the contents of the stack intact, and returns a NULL pointer if the stack is empty. If `Stack_top()` had not been provided as part of the interface to the Stack module, the client code could have its own `Stack_top()` function, built using only the other functions in the module interface. Provide a correct implementation of `Stack_top()` that relies only of the other functions in the module interface. (7 pts)