

NAME:

Login name:

**Computer Science 217
Midterm Exam
October 27, 2005
10am-10:50am**

This test has six (6) questions – five regular and one “extra credit.” 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.”

Question	Score
1 (15 pts)	
2 (15 pts)	
3 (15 pts)	
4 (20 pts)	
5 (35 pts)	
6 (5 bonus)	
Total	

QUESTION 1: Modulo Arithmetic and Character Output (15 POINTS)

Consider the following function:

```
void f(unsigned int n)
{
    do {
        putchar('0' + (n % 10));
    } while (n /= 10);
    putchar('\n');
}
```

1a) What is the output of `f(837)`? (5 points)

1b) Give a concise, one-sentence description of what this function does. (5 points)

1c) In which cases (if any) would the alternative implementation below produce a different output than the original implementation? What outputs would the two versions give? (5 points)

```
void f(unsigned int n)
{
    for ( ; n; n /= 10)
        putchar('0' + (n % 10));
    putchar('\n');
}
```

QUESTION 2: Pointers and Strings (15 POINTS)

Consider the following program:

```
#include <stdio.h>

void f(char *s) {
    char *p = s;

    while (*s)
        s++;

    for (s--; s>p; s--,p++) {
        char c = *s;
        *s = *p;
        *p = c;
    }
}

int main() {
    char a[20] = "feeling lucky";

    f(a);
    printf("%s\n", a);
    return 0;
}
```

2a) What does the program print to standard output? (5 points)

2b) In function `f()`, why are `p` and `s` “`char *`” rather than “`const char *`”? (5 points)

2c) Give a concise, one-sentence description of what function `f()` does. (5 points)

QUESTION 3: Short Answer (15 POINTS, 3 points each)

3a) What does `printf("%d, %d\n", 5/3, 5 % 3)` print to standard output?

3b) Rewrite the C declaration "`float (*f)[25]`" in English. For example, the declaration "`int *x`" would be rewritten in English as "`x` is a pointer to an integer."

3c) Rewrite the C declaration "`float *(*foo[7])(int *)`" in English. For example, the declaration "`int *x`" would be rewritten in English as "`x` is a pointer to an integer."

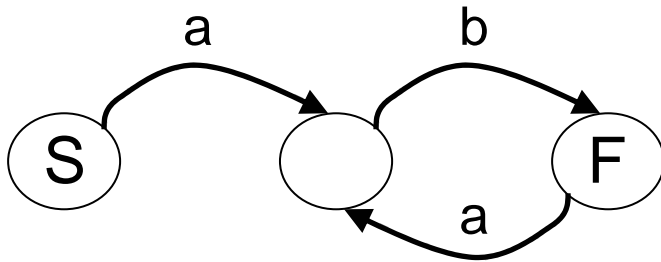
3d) In the memory layout of a UNIX process, what is the difference between the heap and the stack sections? What is stored in the heap and what is stored in the stack? What is the difference between the data and BSS sections?

3e) In the memory layout for a UNIX process, why does the heap grow from the top down and the stack from the bottom up, instead of both growing from the top down or both growing from the bottom up?

QUESTION 4: Deterministic Finite Automata (20 POINTS)

A lexical analyzer is one of the first components of a compiler and is responsible for recognizing the top-level elements of the program. The operations of the lexical analyzer can be represented using a deterministic finite automaton (DFA). For this question, draw a DFA that determines whether a string represents a floating-point number which may include a sign, an integer portion, a fractional portion, and an exponent. The DFA analyzes one character at a time and reaches the end of the string in a “success” state for inputs such as “-34”, “78.1”, “+298.3”, “-34.7e-1”, “34.7E-1”, “7.”, “.7”, and “999.99e99”. The DFA ends in “failure” for inputs such as “abc”, “-e9”, “1e”, “+”, “17.9A”, “0.38+”, “.”, and “38.38f9”.

Your DFA diagram should include a start state (labeled S) and one or more success states (labeled F), and show only the valid transitions between states; that is, your diagram should not include arcs to states that can never lead to a valid input. You can use the symbol “#” as a wildcard that matches any one-digit number (e.g., ‘0’, ‘1’, ..., ‘9’). As an example, consider a DFA that reports “success” for an input that repeats the characters “ab” one or more times and reports “failure” otherwise. That DFA would be drawn as:



The inputs “ab”, “abab”, and “ababab” would lead to a valid state, whereas “a”, “aba”, “abc”, “789”, or “cabab” would not. On the next page, draw a DFA that recognizes floating-point numbers. Please be neat.

QUESTION 5: Abstract Data Types (35 POINTS)

A queue is a first-in-first-out data structure. The Queue ADT offers a simple interface to clients that creates a new queue, checks if a queue is empty, adds an item to the end of a queue, and removes an item from the beginning of a queue. The `queue.h` file specifies the interface, and the `queue.c` file has the code for implementing a Queue using a linked list, where the “head” points to the first element in the list and the “tail” points to the last element. First, `queue.h` has

```
#ifndef QUEUE_INCLUDED
#define QUEUE_INCLUDED

typedef struct Queue_t *Queue_T;

extern Queue_T Queue_new(void);
extern int Queue_empty(Queue_T queue);
extern void Queue_add(Queue_T queue, void* item);
extern void* Queue_remove(Queue_T queue);

#endif
```

Then, `queue.c` has

```
#include <stdlib.h>
#include <assert.h>
#include "queue.h"

struct list {
    void* item;
    struct list *next;
};

struct Queue_t {
    struct list *head;
    struct list *tail;
};

Queue_T Queue_new(void) {
    Queue_T queue = malloc(sizeof *queue);
    assert(queue != NULL);
    queue->head = NULL;
    queue->tail = NULL;

    return queue;
}
```

along with the code for `Queue_empty()`, `Queue_add()`, and `Queue_remove()`.

QUESTION 5 (continued)

5a) Why is `item` defined as `void*` instead of a specific type like `Item_T` or `int*`? (5 points)

5b) Why is `Queue_t` declared in `queue.c`, rather than in `queue.h`? (5 points)

5c) Write the code for `int Queue_empty(Queue_T queue)`, taking care to check using an `assert()` that the input parameter is valid. The function returns `1` if the queue is empty and `0` otherwise. (5 points)

5d) Write the code for `void Queue_add(Queue_T queue)` that adds the element to the queue, allocating memory as needed. The code should include any necessary `assert()` checks. (10 points)

QUESTION 5 (continued)

5e) Write the code for `void* Queue_remove(Queue_T queue)` that removes the first element in the queue and returns the associated item, returning memory as needed. The code should include any necessary `assert()` checks. (10 points)

QUESTION 6: C Puzzle (5 BONUS POINTS)

The code below was written to print twenty dash ('-') characters in a row. However, the code has a bug and prints dash characters indefinitely.

```
int i, n=20;

for (i=0; i < n; i--)
    printf("-");
```

Identify two ways to change just one character in the code (i.e., adding, removing, or modifying a single character) such that the resulting code produces the correct output. Providing one correct answer will result in half credit for the question. There are three known answers. *This question may take some time and is only worth five extra-credit points, so you should complete the rest of the exam to your satisfaction before working on this question.*