CSC 209H5 S 2013 Midterm
Duration — 50 minutes
Aids allowed: none

Student Number: |_|_|_|_|_|_|_|_|_|_|_|

Last Name: _____ **First Name:** _____

Lecture Section: L101 **Instructor: Daniel Zingaro**

---

*Do **not** turn this page until you have received the signal to start.*
(Please fill out the identification section above, **write your name on the back of the test**, and read the instructions below.)
Good Luck!

This midterm consists of 4 questions on 8 pages (including this one). *When you receive the signal to start, please make sure that your copy is complete.* If you use any space for rough work, indicate clearly what you want marked.

1: _____/ 5

2: _____/ 6

3: _____/ 4

4: _____/ 5

TOTAL: _____/20

Question 1. [5 MARKS]

Part (a) [2 MARKS]

In no more than a few sentences, explain what is done by the following code. Give an overall description of the code's purpose, not a line-by-line explanation of the code. Assume that `in` exists.

```
#!/bin/bash

while read line; do
  wc -l < $line
done <in >out
```

Part (b) [1 MARK]

Write one line of Bash code that writes the text `SEM` to standard error.

Part (c) [2 MARKS]

Briefly explain how the shell uses `fork` and an `exec` call to run new processes.

Question 2. [6 MARKS]

Write a shell script that supports the following synopsis:

```
numnewer reffile [dir]...
```

`reffile` is the name of a file in the current directory, and `[dir]...` is zero or more names of directories. `numnewer` produces as output a single integer giving the total number of files in the directories `[dir]...` that are newer than `reffile`.

As an example, consider this `ls` call:

```
$ ls *
numnewer  xyzzy

dir1:
a

dir2:
b  c
```

Imagine that `b` is newer than `xyzzy`, but that `a` and `c` are older than `xyzzy`. If I invoke
`numnewer xyzzy dir1 dir2`
then the output is the integer `1`.

Do not write any error-checking code for the `numnewer` call; i.e. assume that `reffile` is provided and exists. In addition, you may assume that each directory exists and that it contains only regular files (no subdirectories).

Question 3. [4 MARKS]

Here is a declaration for `struct node`:

```
struct node {
  char s[30];
  struct node *next;
};
```

`head` is a pointer to the first node in a linked list. Write function `findnode` that returns `1` if the node pointed to by `find` is found in the linked list, and `0` otherwise.

```
int findnode(const struct node *head, const struct node *find) {
```

The following shows sample calls for `findnode`. Note that `n3` is **not** in the list.

```
int main(void) {
  struct node *n1 = malloc(sizeof(struct node));
  struct node *n2 = malloc(sizeof(struct node));
  struct node *n3 = malloc(sizeof(struct node));
  strcpy(n1->s, "dan");
  strcpy(n2->s, "dan");
  strcpy(n3->s, "dan");

  n1->next = n2;
  n2->next = NULL;
  n3->next = NULL;

  printf("%d\n", findnode(n1, n2)); //outputs 1
  printf("%d\n", findnode(n1, n3)); //outputs 0
  return 0;
}
```

Question 4. [5 MARKS]

This question has two parts. Below is a partial implementation of function `copystrs`. `s` is an array of `max+1` bytes, and `strs` is an array of strings. `strs[0]` is the first string, `strs[1]` is the second string, etc. `strs` is guaranteed to include a `NULL` pointer to indicate that no further elements are initialized.

The function is designed to copy characters from each successive string in `strs`, until there is no room remaining in `s`. There is a sample call below the function that I encourage you to understand before continuing.

Part (a) [3 MARKS] Write the code for the missing loop. The code inside the loop is responsible for copying characters into `s`, updating `max`, and advancing through `strs`. Be sure that you null-terminate `s`. **You must use `strcat` or `strncat` in your solution.**

```
void copystrs(char *s, char **strs, int max) {
  s[0] = '\0'; // part B
  while (*strs != NULL && max > 0) {
    // copy into s, update max, advance strs









  }
}

int main(void) {
  char s[6];
  char *strs[4] = {"ab", "cd", "ef", NULL};
  copystrs(s, strs, 5);
  printf("%s\n", s); // outputs abcde
  return 0;
}
```

Part (b) [2 MARKS] The first line of code with the `// part B` comment puts a null character at the beginning of `s`. Suppose this line were removed. Would your function still work as expected? Briefly explain.

C function prototypes

```
int printf(const char *format, ...)

char *strchr(const char *s, int c) //Search from left
char *strrchr(const char *s, int c) //Search from right
char *strstr(const char *s1, const char *s2) //Search for s2

size_t strlen(const char *s)
char *strncat(char *dest, const char *src, size_t n)
int strncmp(const char *s1, const char *s2, size_t n)
char *strncpy(char *dest, const char *src, size_t n)
```

Shell `test` Operators

`-a`, `-o`	and, or
`-eq`, `-ne`	equality and inequality on ints
`-gt`, `-lt`, `-ge`, `-le`	greater-than, less-than, etc. on ints
`=`, `!=`	equality and inequality on strings
`-z`	Empty string?
`-f`	File is a regular file?
`-d`	File is a directory?

[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]

Last Name: _____ **First Name:** _____