CSC 209H1 Y 2012 Midterm
Duration — 50 minutes
Aids allowed: none

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

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

Lecture Section: 1 **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: _____/ 3

2: _____/ 6

3: _____/ 6

4: _____/ 4

TOTAL: _____/19

Question 1. [3 MARKS]

Part (a) [2 MARKS]

I would like to have two filenames `a` and `b` that refer to the same file. Also, if I remove one of `a` or `b`, I still want to be able to access the file through the other filename. Which type of link should I use? Explain why the other type of link would not work.

Part (b) [1 MARK]

What would be printed by the following shell command? (' is single quote.)

```
echo '*.c'
```

Question 2. [6 MARKS]

Part (a) [2 MARKS]

The `date` program produces a line containing six space-separated fields representing the current date and time, like this:

```
greywolf:~$ date
Sat Jun 23 14:09:41 EDT 2012
```

The fourth field here contains three colon-separated fields: the current hour, the current minute, and the current second.

Write a shell script that, when run, prints **only** the current hour. For the `date` call above, your script would print `14`.

Part (b) [4 MARKS]

Calling `date` with the `-r` option allows you to obtain the last modification date and time of a file, like this:

```
greywolf:~$ date -r vgm1.csv
Wed Apr  4 12:17:02 EDT 2012
```

Write a shell script that supports the following synopsis:

```
filedates [-n] [file]...
```

That is, the script takes an optional `-n` and then **zero or more** filenames on the commandline. If `-n` is not provided, it prints each filename and its last modification date on the same line. If `-n` **is** provided, then only last modification dates are printed. If multiple files are provided, the information for each one is printed on a separate line. Here are two examples:

```
greywolf:~$ filedates vgm1.csv
vgm1.csv Wed Apr  4 12:17:02 EDT 2012
greywolf:~$ filedates -n vgm1.csv vgm2.csv
Wed Apr  4 12:17:02 EDT 2012
Thu Apr  5 14:42:44 EDT 2012
```

Don't error-check the commandline: all filenames are guaranteed to exist and be readable, and no option besides the optional `-n` will be provided.

Question 3. [6 MARKS]

Consider the following `typedef`, which creates a new type called `Name`.

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

`nl` is a linked list of `Name`s. Write a function `tostring` that returns a string consisting of each string in `nl` concatenated together with spaces between each pair of strings. For example, if `nl` has three nodes that, in order, have `s` components of `celes`, `terra`, and `locke`, the resulting string will be `celes terra locke`. You must `malloc` exactly the right amount of memory for the resulting string. Since you don't know in advance how much memory to `malloc`, traverse `nl` twice.

```
char *tostring (const Name *nl) {
```

Question 4. [4 MARKS]

Part (a) [2 MARKS]

The following code exhibits a memory leak. Can it be fixed by adding a `free` call **after** the existing code? If so, add it below the comment; if not, explain why not. Assume the `malloc` call succeeds.

```
int *x = NULL;
int y = 9;
x = malloc(sizeof(int));
*x = 15;
x = &y;
/*Possible call of free here?*/
```

Part (b) [2 MARKS]

Give the output produced by the following program assuming that `SIGINT` arrives at the location specified by the comment. Assume that the program runs without error; error-handling has been removed to make the code shorter.

```
void catch(int code) {
  printf("Caught signal\n");
}

int main(void) {
  struct sigaction oldact, newact;
  sigset_t oldset, set;
  sigemptyset (&newact.sa_mask);
  newact.sa_flags = 0;
  newact.sa_handler = catch;
  sigaction(SIGINT, &newact, &oldact);
  sigfillset (&set);
  sigprocmask (SIG_SETMASK, &set, &oldset);
  printf ("Setup complete\n");
  /*SIGINT is delivered here*/
  sigaction(SIGINT, &oldact, NULL);
  printf ("About to terminate\n");
  return 0;
}
```

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)

int sigaction(int sig, const struct sigaction *act, struct sigaction *oldact)
int sigprocmask(int how, const sigset_t *set, sigset_t *oldset)
int sigemptyset(sigset_t *set)
int sigfillset(sigset_t *set)
int sigaddset(sigset_t *set, int signo)
int sigdelset(sigset_t *set, int signo)
int sigismember(const sigset_t *set, int signo)
```

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?

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