CSC 209H1 S 2013 Midterm Test
Duration — 50 minutes
Aids allowed: none

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

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

Lecture Section: L5101 **Instructor: Craig**

---

*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.* Comments are not required except where indicated, although they may help us mark your answers. They may also get you part marks if you can't figure out how to write the code.
If you use any space for rough work, indicate clearly what you want marked.

1: _____/ 9
2: _____/ 5
3: _____/ 4
4: _____/12

TOTAL: _____/30

Question 1. [9 MARKS]

In the box beside each set of `bash` statements, show the output when the statements are executed. If running these statements would result in an error, print ERROR and give a brief explanation.

Part (a) [1 MARK]

```
v=mon
v="$v day"
echo $v
```

Part (b) [1 MARK]

```
y=thurs
y='$y day'
echo $y
```

Part (c) [1 MARK]

```
z="fri day"
for piece in "$z"
  do
    echo "$piece *"
  done
```

Part (d) [1 MARK]

```
z="fri day"
for piece in $z
  do
    echo "$piece *"
  done
```

Part (e) [2 MARKS]

Write a few lines of shell that would display the phrase `The current month is XXX` with the current month inserted instead of `XXX` Reminder: the default output from the command `date` looks as follows:

```
Wed Feb 20 16:36:00 EST 2013
```

Part (f) [3 MARKS]

Some of the files in the current working directory contain the phrase `FIX ME` in places that need correction. When we run the command `grep "FIX ME" *` the output contains one line for every line in the files containing `FIX ME`. The first few lines of output are shown here.

```
buxfer.c:FIX ME
commands.txt:FIX ME
commands.txt:line that needs changes // FIX ME
```

Write a single shell expression (using pipes) the counts the number of **different** files that need fixing (i.e. the number of files that contain this phrase.)

Use this space for rough work or extra space for any answer. Clearly label anything you want marked.

Question 2. [5 MARKS]

Consider the following C code fragment.

```
char s[11] = "0123456789";
char * t = "source";
char * p = s + 3;
strncpy(s+2, t, 9);
printf("%s\n",s);
```

Part (a) [2 MARKS]

What is the output? If the code results in an error before anything is printed, write ERROR and give a brief explanation.

Part (b) [1 MARK] What is the type of `&t` ?

Part (c) [1 MARK] What is the type of `*p` ?

Part (d) [1 MARK] What is the **value** of `p[2]` after this fragment has executed?

Question 3. [4 MARKS]

Suppose the current working directory contains the following files:

```
cheese  makefile
```

The box to the right contains the contents of `makefile`.

In the table below are three commands that are run sequentially (one after the other) from the shell. Complete the table by filling in the output that be printed to standard out and the names of files that are changed or created.

```
dinner: pizza salad

pizza: cheese topping.peppers
        echo making pizza
        cat cheese topping.peppers  > pizza

cheese:
        echo goey cheese > cheese

topping.%:
        echo yummy $@ > $@

salad:
        echo salad is healthy
```

Command	Output printed to standard output	Names of Files Created or Changed
make salad		
make pizza		
make dinner		

Question 4. [12 MARKS]

Part (a) [7 MARKS] Consider the C program below which produces the output:

```
first token is Craig and second token is Michelle
first token is Reid and second token is Karen
```

Complete function `split_on_comma` as described in the function comment. Your program should generate this output without making any changes to `main`. Your function must **not** change parameter `s`. You may want to use function `char * strchr(const char *s, int c)` which returns a pointer to the first occurence of `c` in `s`. You do not need to check the return values of your system calls.

```
/* Return an array of two strings. The first is the substring of s from the beginning
 * up to but not including the comma. The second is the substring of s after the comma.
 * Precondition: string s contains exactly one comma. */
char ** split_on_comma(char * s) {
```

```
int main() {
    char s1[15] = "Craig,Michelle";
    char ** r = split_on_comma(s1);
    printf("first token is %s and second token is %s\n",r[0],r[1]);

    char * s2 = "Reid,Karen";
    r = split_on_comma(s2);
    printf("first token is %s and second token is %s\n",r[0],r[1]);
    return 0;
}
```

Part (b) [1 MARK]

We explicitly stated that your function could not change parameter `s`. State when and why the program would fail if you didn't follow this instruction.

- Which call to `split_on_comma` would fail? (pick one)

  ☐ Only the first call (on `s1`) would fail.
  ☐ Only the second call (on `s2`) would fail.
  ☐ Both calls would fail.

- Why would it fail?

Part (c) [1 MARK]

As written, the program has a memory leak in the `main` function. Explain where this is.

Part (d) [3 MARKS]

Complete a function `free_all` that could be added to the program to fix the memory leak. Write the function here, but add the call(s) to `free_all` into the main function on the previous page.

C function prototypes:

```
int fclose(FILE *stream)
char *fgets(char *s, int n, FILE *stream)
FILE *fopen(const char *file, const char *mode)
size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);
void free(void *)
int fseek(FILE *stream, long offset, int whence)
size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE *stream);
char *index(const char *s, int c)
void *malloc(size_t size);
void perror(const char *s)
int sprintf(char *s, const char *format, ...)
char *strchr(const char *s, int c)
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)
char *strrchr(const char *s, int c)
char *strstr(const char *haystack, const char *needle)
```

Useful Unix programs for shell programs: `cat, cut, wc, grep, sort, sort -n (for numerical sorting), head, tail, echo, set, uniq`

Makefile variables: `$@ target, $^ list of prerequisites, $< first prerequisite.`

Print your name in this box.