CSC 209H1 F 2017 Midterm Test
Duration — 50 minutes
Aids allowed: none

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

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

Section: L0101 (1:10-2:00pm)
Instructor: Reid

---

*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 5 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, 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.
No error checking is required unless you are specifically requested to do it for an individual question.
You do not need to provide the include statements for your programs.
If you use any space for rough work, indicate clearly what you want marked.

1: _____/ 4
2: _____/ 4
3: _____/ 4
4: _____/ 6
5: _____/ 5

TOTAL: _____/23

Question 1. [4 MARKS]

Assume you have a terminal open, and the current working directory contains a C program file called `calculate.c` and a text file called `grades.txt`.

Part (a) [2 MARKS]

Write a shell command that compiles `calculate.c` into an executable called `doit`. Include the flag to display all warning messages.

Part (b) [2 MARKS]

Given the following makefile rule, explain what would cause the action of the rule to be executed when we run `make testout`. Assume that the files `testout`, `doit`, and `grades.txt` all exist in the current working directory.

```
testout : doit grades.txt
    doit grades.text > testout
```

Question 2. [4 MARKS]

For each code fragment below, if the code will not compile or will generate a warning when compiled with the `-Wall` flag, check `COMPILE ERROR` and explain why. If the code will compile, but is not guaranteed to run without an error, check `RUN-TIME ERROR` and explain why. Otherwise, check `NO ERROR` and show what is printed. The first one is done for you.

Code Fragment	ERROR	Output or explanation for error

`int y = 2;`
`int x = y;`
`printf("%d %d", x, y);`
	☑ NO ERROR
	☐ COMPILE ERROR
	☐ RUN-TIME ERROR
		2 2

`char word[4];`
`char *thing = "box";`
`word = thing;`
`printf("%s", word);`
	☐ NO ERROR
	☐ COMPILE ERROR
	☐ RUN-TIME ERROR	

`char **w = malloc(2 * sizeof(char *));`
`w[0] = "hello";`
`*(w + 1) = w[0] + 1;`
`printf("%s, %s", w[0], w[1]);`
	☐ NO ERROR
	☐ COMPILE ERROR
	☐ RUN-TIME ERROR	

`char name[] = "You";`
`char *person = name;`
`printf("%s\n", person);`
`free(person);`
	☐ NO ERROR
	☐ COMPILE ERROR
	☐ RUN-TIME ERROR	

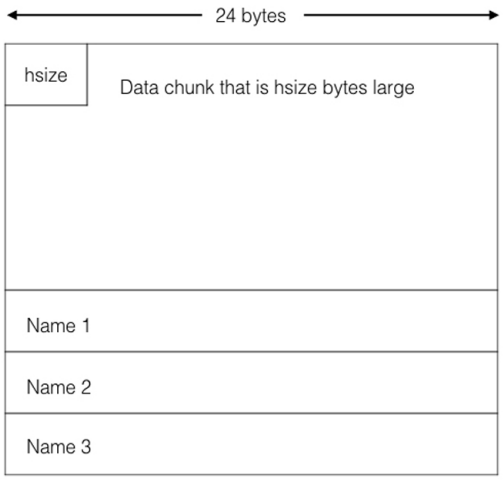
`char *names[2];`
`names[0] = malloc(5 * sizeof(char));`
`strncpy(names[0], "Jane", 5);`
`names[1] = "John";`
`for (int i = 0; i < 5; i++) {`
`  names[0][i] = 'A';`
`  names[1][i] = 'B';`
`}`
`printf("%s\n%s", names[0], names[1]);`
	☐ NO ERROR
	☐ COMPILE ERROR
	☐ RUN-TIME ERROR	

Question 3. [4 MARKS]

Write the `terminate_string()` function below according to its description. Notice that variable `c` is not necessarily a string before the function is called.

```
/* From the index of the first null terminator in character array c up to and
 * including index idx, replace each character with a null terminator.
 *
 * If c does not contain a null terminator within the characters from indexes
 * 0 to idx inclusive, replaces the charcter at index idx with a null terminator.
 *
 * Return the index of the first null terminator in c after the
 * replacement(s).
 *
 * Assume that idx >= 0.
 */
int terminate_string(char *c, int idx) {









}
```

Question 4. [6 MARKS]



You are given a binary file with the following format:

- The first four bytes of the file are an integer that contains the size of the data chunk that follows, not including the first 4 bytes.
- After the data chunk is a sequence of strings representing names. Each string is stored in a 24 byte character array.

The layout of the file is shown in the diagram on the right.

Complete the program below that takes a file name of the above format as a command line argument, and prints to standard output the second string in the sequence.

```
int main(int argc, char **argv) {
```

Question 5. [5 MARKS]

Write the `concat` function below according to its description.

```
typedef struct node {
    char *item;
    struct node *next;
} Node;

/*
 * Given the head of a linked list where each node stores a string,
 * return a dynamically-allocated string obtained by concatenating
 * each string in the linked list, in order they appear in the list.
 *
 * You may assume the linked list is non-empty, and contains only
 * valid strings.
 *
 * Do this in two passes through the linked list:
 *   1. First, calculate the total amount of space needed to store
 *      the output string.
 *   2. Allocate space for the string, build the final result, and return.
 */
char *concat(Node *head) {









}
```

[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.]

C function prototypes:

```
int fclose(FILE *stream)
FILE *fopen(const char *file, const char *mode)
int fprintf(FILE *stream, const char *format, ...);
size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);
void free(void *ptr)
int fscanf(FILE *restrict stream, const char *restrict format, ...);
int fseek(FILE *stream, long offset, int whence)
size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE *stream);
void *malloc(size_t size);
void perror(const char *s)
int scanf(const char *restrict 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 *strstr(const char *haystack, const char *needle)
```

Excerpt from fseek man page:

```
If whence is set to SEEK_SET, SEEK_CUR, or SEEK_END, the offset  is  relative
to the  start of the file, the current position indicator, or end-of-file,
respectively.
```

Excerpt from fopen man page:

```
The argument mode points to a string beginning with one  of  the  following
sequences (possibly followed by additional characters, as described below):

   r      Open  text file for reading.  The stream is positioned at the begin
          ning of the file.
   r+     Open for reading and writing.   The  stream  is  positioned  at  the
          beginning of the file.
   w      Truncate  file  to zero length or create text file for writing.  The
          stream is positioned at the beginning of the file.
   w+     Open for reading and writing.  The file is created if  it  does  not
          exist,  otherwise  it is truncated.  The stream is positioned at the
           beginning of the file.
```

Print your name in this box.

END OF EXAMINATION