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

Student Number: ____________

Last Name: ______________________ **First Name:** ______________________

Instructor: Reid
Section: L0201 (12:10-1:00pm)

---

*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 6 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:	______/10
# 2:	______/ 3
# 3:	______/ 1
# 4:	______/ 2
# 5:	______/ 4
# 6:	______/ 6
TOTAL:	______/26

Question 1. [10 marks]

Part (a) [7 marks]

```c
struct award {
    char *name;
    char **nominees;
};

// Complete the function below that will allocate just enough memory on the heap
// for nominee to store name, and will set nominee to name

void set_nominee(char **nominee, char *name) {






}

int main() {

    char line[1024];
    // Declare a struct award variable named bp


    // Initialize the award's name to the read-only string "Best Picture"



    // Initialize the award's nominees to refer to heap-allocated space for
    // 4 nominees




    printf("Enter a film name:\n");
    fgets(line, 1024, stdin); // assume line is long enough and fgets succeeds

    // Use set_nominee to initialize nominee 0 to the string stored in line


    // Use set_nominee to initialize nominee 1 to the read-only string "Moonlight"



    return 0;
}
```

Part (b) [3 marks]

Check the box beside each of the free statements below that correctly free memory used in the above program. For each line that is not checked, briefly explan why the statement is incorrect.

- ☐ `free(bp.name);`
- ☐ `free(bp.nominees[0]);`
- ☐ `free(bp.nominees[1]);`
- ☐ `free(bp.nominees[2]);`
- ☐ `free(bp.nominees);`
- ☐ `free(bp);`

Question 2. [3 MARKS]

Given the following Makefile

```
labs.gf : lab1.gf lab2.gf
    gather lab1.gf lab2.gf > labs.gf

lab1.gf : lab1.csv classlist
    mkgrades lab1.csv < classlist

lab2.gf : lab2.csv classlist
    mkgrades lab2.csv < classlist

clean :
    rm *.gf
```

The contents of the current working directory:

```
    lab1.csv    lab2.csv    classlist
```

Specfiy which files are created, deleted or modified when the following commands are run in sequence.

Part (a) [1 MARK] `make lab2.gf`

Created	Modified	Deleted

Part (b) [1 MARK] `make`

Created	Modified	Deleted

Part (c) [1 MARK] Suppose `lab1.csv` is modified, and then `make` is run again with no arguments. Which files are created modified or deleted?

Created	Modified	Deleted
	lab1.gf lab1.csv labs.gf	

Question 3. [1 MARK]

When I run

```
gcc -Wall -g -o prog prog.c
```

I get the error message

```
/u/reid/tmp/prog.c:5: undefined reference to ‘foo’
```

Exaplain what I need to do to address this problem.

Question 4. [2 MARKS]

I spent a long time working on my `print_ftree` program on cdf and it works perfectly, but when I try it on my Linux machine at home, once it a while it gives me a segmentation fault.

Part (a) [1 MARK] Is there a bug in my code? Explain your answer.

Part (b) [1 MARK] When does the segmentation fault message appear? Check the appropriate answer or answers.

- ☐ Compile time
- [✓ Run time]
- ☐ When runnning `make print_ftree`

Question 5. [4 MARKS]

For each of the code fragments below, there is missing code. At the very least, the line (or lines) that declare and possibly initialize the variable `x` are missing. If the code will not compile no matter what you put for the missing code, 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 give the correct declaration for x. You don't need to show any other missing code. The first one is done for you.

Code Fragment	ERROR	Declaration for `x` or explanation

`int y;`
`// missing code`
`x = y;`
	☑ `NO ERROR`
	☐ `COMPILE ERROR`
	☐ `RUN-TIME ERROR`
		`int x;`

`char **n = malloc(3*sizeof(char *));`
`// missing code`
`strcpy(n[0], x);`
	☐ `NO ERROR`
	☐ `COMPILE ERROR`
	☐ `RUN-TIME ERROR`	

`char course[7];`
`x = "csc209";`
`// missing code`
`course = x;`
	☐ `NO ERROR`
	☐ `COMPILE ERROR`
	☐ `RUN-TIME ERROR`	

`char *film = "La La Land";`
`// missing code`
`char *x = *film + 6;`
	☐ `NO ERROR`
	☐ `COMPILE ERROR`
	☐ `RUN-TIME ERROR`	

`// struct definition used`
`// for the last two snippets`
`struct Node {`
`    char *fname;`
`    struct Node *next`
`}`		

`x = malloc(sizeof(struct Node));`
`// missing code`
`x.fname =  "file1.txt";`
	☐ `NO ERROR`
	☐ `COMPILE ERROR`
	☐ `RUN-TIME ERROR`	

`struct Node *root;`
`root = malloc(sizeof(Node));`
`// missing code`
`x = &root->next;`
	☐ `NO ERROR`
	☐ `COMPILE ERROR`
	☐ `RUN-TIME ERROR`	

Question 6. [6 marks]

Complete the function below, so that the following examples will work correctly and the minimum amount of memory is used. The only string library functions you may use are `strlen` and `strncpy`. (You are not required to use these functions.)

```
    char *path1 = "/";
    char *result1 = dirname(path1);

    char *path2 = "/usr/include";
    char *result2 = dirname(path2);

    char path3[9] = "file.txt";
    char *result3 = dirname(path3);

    char path4[20] = "a2/test/test1.in";
    char *result4 = dirname(path4);

    printf("%s, %s, %s, %s\n",
         result1, result2, result3, result4);
```

Prints:

```
/, /usr, ., a2/test
```

```
/* Returns the component(s) of path up to but not including the final '/'
 * path will not be NULL
 * If there is no '/' then dirname returns the string "."
 */
char *dirname(char *path) {
```

C function prototypes:

```
int fclose(FILE *stream)
char *fgets(char *s, int n, 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)
DIR *opendir(const char *name)
void perror(const char *s)
int printf(const char *format, ...)
struct dirent *readdir(DIR *dir)
int scanf(const char *restrict format, ...)
int lstat(const char *file_name, struct stat *buf)
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)
```

Excerpt from fgets man page:

```
    fgets() reads in at most one less than size characters from stream  and
    stores  them  into  the buffer pointed to by s.  Reading stops after an
    EOF or a newline.  If a newline is read, it is stored into the  buffer.
    A  terminating  null  byte ('\0') is stored after the last character in
    the buffer.
```

Excerpt from scanf/fscanf man page:

```
RETURN VALUES
     scanf and fscanf return the number of input items assigned.  This can be
     fewer than provided for, or even zero, in the event of a matching fail-
     ure. The value EOF is returned if an input failure occurs before any
     conversion such as an end- of-file occurs.
```

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

Print your name in this box.

End of Examination