CS241: Week 2 Review Session
January 22 / 25
Introduction to C

# 1  Coding environment

<u>Task 1: Write a Bash script</u>

Run the terminal commands on the slides. What is the output you get?

$$\begin{array}{|c|}\hline \\ \\ \\ \\ \\ \\ \hline \end{array}$$

<u>Task 2: Write a Bash script</u>

Replace the parameters given on the slide to use the find utility. Also, use the man page to figure out how to use grep to search lines within a file. Where is the file found? How many calls are there?

$$\begin{array}{|c|}\hline \\ \\ \\ \\ \hline \end{array}$$

# 2  Pointers in C

Fill in the corresponding dereferenced values of the pointers.

Round 1: _____

Round 2: _____

Round 3: _____

Round 4: _____

Round 5: _____

Round 6: _____

# 3   Input and Output in C

## 3.1   printf and scanf

What string formats are used to represent the following data types / special characters in printf? [3 pt]

Type "man 3 printf" into your terminal.

| Data type / character | String format |
|---|---|
| String | |
| Integer | |
| Character | |
| Pointer | |
| Float | |
| size_t (a type of unsigned integer) | |

Fill in the string formats for the following printf statements. [2]

1. printf("(_____,_____)", intVar, stringVar);
2. printf("(_____,_____)", characterVar, &characterVar);

What should we put into the parameters for scanf? [3 pt]
Use "man 3 scanf" in your terminal. Draw out the variables if you're not sure!

1. char input1;
   char input2;
   int received;

   _____ = scanf("%c %c", _____, _____);

2. char* textBuffer = calloc(200,1);
   int received;

   _____ = scanf("%s", _____);

3. char textBuffer[256];
   int received;

   _____ = scanf("%s", _____);

## 3.2   getline

Use "man 3 getline" for more information

Answer select true or false for the following descriptions of getline. [3 pts]
Below is an example usage of getline.

```
while((read = getline(&line, &len, stdin)) != 1) {
    printf("%s", line);
}
```

1. getline requires the address of your character buffer          T / F

2. line is a character pointer                                    T / F

3. line must point to an allocated block of memory                T / F

4. getline may allocate / reallocate memory for the buffer        T / F

5. getline requires the address of your buffer length variable    T / F

6. getline returns the number of characters read                  T / F

## 3.3 Other input / output functions

Why is it a security flaw to use gets? [1 pt]
```
char *gets(char* s);
```

_____

_____


What version of scanf can you use to read from a file? [0.5pt]    _____

What about reading from a character buffer? [0.5pt]    _____

How many parameters does getchar take in? [0.5 pt]    _____

What version of printf can you use to print to a string? [0.5 pt]    _____


# 4 Memory allocation in C

What kind of memory leaks do you get for each valgrind file? [0.5pts each]

valgrind1.c

valgrind2.c

valgrind3.c

Look at memory1.c and memory2.c

<u>Why would we use calloc over malloc?</u> [0.5 pts]

Answer the following questions about realloc [0.5 pts each]

1.  The pointer variable must be passed into realloc                    T / F

2.  realloc always returns a different address                          T / F

3.  If realloc returns a different address, you do not need to          T / F
    free the original pointer.

4.  Any new memory returned by realloc will be initialized              T / F