CS 300 Exam 1 Review

**VOCABULARY Expect to write some code, read some code, explain concepts Read all your slides (even ones we skipped), Read the example code. Read your project code.**

**Linux:**

What is the main job of the Linux kernel?
What is the main job of the shell?

If you are logged into your Linux VM, which directory do you expect to be in after typing `cd ~`

Why do we use the `tar` command?

Who is allowed to read the following file:

```
chadd@linux-9f91:~/cs00f18> ls -al README.md
-rw-r----- 1 chadd users 0 Jul 18 20:25 README.md
```

Was the -l option necessary to answer the question above?

Was the -a option necessary to answer the question above?

What is the difference between scp and wget? Both will transfer files.

Given the following:

```
chadd@linux-9f91:~/cs300f18/FutureLab/bin> ls -al F_Lab
-rwxr-xr-x 1 chadd users 22112 Sep 12 20:13 F_Lab
chadd@linux-9f91:~/cs300f18/FutureLab/bin> ./F_Lab
```

Why is the ./ necessary before F_Lab, or why is it not really necessary?

gcc -c will produce what type of file as output?

**Basic C:**

What does it mean when Chadd says a pointer variable “has access to two values”? What are those two values? How do you access each?

Who’s job is it to take care of #define and #include?

Why does scanf() require a pointer as an argument?

Where (stack or heap) are local variables stored?

How do you create variables in the heap?

## Makefiles:

What is a target? What is a dependency?

How does make decide whether or not to execute a command listed in the Makefile?

What should the tarball target depend on? Why?

Why is it useful to build the .o files in a separate rule and then combine them into an executable in another rule?

## Dynamic Memory/Pointers:

Declare an integer and a pointer to an integer. Set the integer to 3, put the address of the integer in the pointer and then use the pointer to display, to the screen, the value of the integer. Also display the address of the integer. Display the address of the pointer.

Why is the type void * useful? Why can it be tricky to use?

How is pass by reference achieved in C? Specify what is provided/expected in the function and in the function call.

```
int *pInt;
int value;
void *pPtr;
```

What data type is pInt?
Write code to point pInt at a single, dynamically allocated integer. Set that integer to 31;
Write code to free the int.
Write code to point pInt at a dynamically allocated array of 10 integers.
Use pointer notation to set each int in the array above to 7. Use array notation to print each int in the array above.
Write code to free the above array.
Write code to point pInt at value. Is value static memory or dynamic memory?
Draw a picture of memory to show the values and addresses of pInt and value. Be sure to denote the stack and the heap.
Use pInt to set the value to 300. Use pInt to printf() value to the screen.
Can you call free using pInt now? Why or why not?
What does the parameter to malloc() mean?

Does the following C code work? If so, what is the output, if not why not?

```
value = 101;
pPtr = &value;
printf(“%d”, *pPtr);
```

**ADTs:**

What does the abstract in ADT mean? What, exactly, is abstracted away? Why do we want to abstract this away?

Imagine you want to provide something like the string.h API from C++ (or any modern language) in C. How might you represent the string datatype in C (*What would the definition of* ***`struct string { }`*** *look like*)? What are two operations that your users would want? What are the function prototypes for those two operations? Implement those two functions.

**Stack ADT:**

What does LIFO mean?

What are the operations for a stack?

Using the stack from stk.h, draw the array and the value of top after each operation completes. The stack starts out empty.

createStack(); push(1); push(2); pop(); push(3); pop(); peek(); push(4)

Imagine a linked list rather than an array is used to implement the stack above. Draw the linked list after each operation above.

►Given a string, ABCDEFG, use a stack to help you print that string in *gibberish order*: ACEGFDB
other examples: 1234567 → 1357642 ZYXWVU → ZXVUWY AB → AB 123→ 132
Write a function that takes a string and uses your stack to achieve the above.
GKOCOUDLGiven:

```
typedef struct ListElement* ListElementPtr;
typedef struct List* ListPtr;

typedef struct ListElement {
   int data;
   ListElementPtr psNext;
} ListElement;

typedef struct List {
   ListElementPtr psHead;
} List;
```

Don’t use any functions from list.h, write all the code you need here.

1. Assume you have declared a List called sReviewList and that list contains the integers 150, 250, and 300 in that order. Draw the list.

2. Write code to display the list in reverse order (300, 250, 150).

3. Write code to insert the integer 260 between 250 and 300.