

COP 3223 Quiz #4 11/14/2025

Last Name: _____, First Name: _____

Circle Lab Time: 8:30 am

9:30 am

11:30 am

12:30 pm

1:30 pm

2:30 pm

3:30 pm

4:30 pm

1) (10 pts) Write a function, **leftcycle**, that takes in a pointer to an array of strings, **words**, as well as the length of that array, **numwords**, and cyclically shifts the strings in the array to the left by one place. This means that each string in the array moves to the index below its previous location and the string in index 0 goes to the end of the array. For example, if the array looked like this before the function call:

coffee	dog	cat	homework	sponge	soda
--------	-----	-----	----------	--------	------

then after the function call the array would look like this:

dog	cat	homework	sponge	soda	coffee
-----	-----	----------	--------	------	--------

You may assume that the length of each string in the input array is **less than** 20 characters.

```
void leftcycle(char words[][20], int numwords) {
```

```
}
```

2) (8 pts) Complete the program below so that it reads its input from standard input, character by character, and for each non-digit character, outputs it, but for each digit, your program will output the character that is that digit subtracted from 9. For example, if the input has the character '3', then your program should output the character '6'. Please use the function putchar only to output.

```
#include <stdio.h>
#include <ctype.h>

int main() {

    int c;
    while ((c = getchar()) != EOF) {

        }
        return 0;
    }
```

3) (17 pts) We can model a mountainous area as a two-dimensional array of integers, where the integer stored in `elevation[x][y]` represents the elevation in the location with coordinates (x, y) on the Cartesian plane. A valley is a cell with 8 adjacent cells that each have a **strictly greater** height. Write a function that returns the number of cells in the input array, **elevation**, that are valley cells. The second input parameter to the function, **n**, represents that the meaningful data is being stored in the first n rows and n columns (square grid) of the input array. (You may assume $n \leq \text{MAX}$.) For full credit, please use the DX and DY arrays provided. They will greatly shorten your code.

```
#define MAX 100
```

```
const int DX[] = {-1,-1,-1, 0,0, 1,1,1};
```

```
const int DY[] = {-1, 0, 1,-1,1,-1,0,1};
```

```
const int NUMDIR = 8;
```

```
int numValleyCells(int elevation[][MAX], int n) {
```

```
}
```

4) (14 pts) Consider managing money with a money struct (shown below). Please complete the three functions below. The totalCents function should return the equivalent number of cents of the money struct pointed to by mine. The compare function should return a negative integer if the money object pointed to by pt1 is less than the money object pointed to by pt2, 0 if they are equal, and a positive integer otherwise. Finally, the deductPayment function should determine if the money struct pointed to by pay is less than or equal to the money struct pointed to by mine, if so, then pay should be subtracted from mine and mine should be adjusted so its new dollars and cents components are accurate and cents are in between 0 and 99, followed by returning 1. If the money struct pointed to by pay is greater than the one pointed to by mine, make no change to mine and return 0.

```
struct money {
    int dollars;
    int cents;
};

int totalCents(const struct money* mine) {

}

int compare(const struct money* pt1, const struct money* pt2) {

}

int deductPayment(struct money* mine, const struct money* pay) {

}
```

5) (1 pt) What color is the container for Big Red Chewing Gum? _____

C Language Reference

stdio.h function

```
// Reads the next character from standard input.  
int getchar(void ;
```

```
// Writes a character to the standard output.  
int putchar(int character);
```

string.h functions

```
// Returns a negative integer if the string pointed to by str1  
// comes before the string pointed to by str2 lexicographically,  
// 0 if both strings are equal and a positive integer if the  
// string pointed to by str1 comes after the string pointed to  
// by str2 lexicographically.  
int strcmp(const char* str1, const char* str2);
```

```
// Returns the length of the string pointed to by str.  
int strlen(const char* str);
```

```
// Copies the contents of the string pointed to by src into  
// the string pointed to by dest and returns a pointer to the  
// memory address where the string was copied.  
char* strcpy(char* dest, const char* src);
```

```
// Appends the contents of the string pointed to by src  
// to the string pointed to by dest and returns a pointer to  
// the memory address of the beginning of the concatenated  
// string.  
char* strcat(char* dest, const char* src);
```

ctype.h macros

```
// Returns 1 if c is a digit, and 0 otherwise.  
int isdigit(int c)
```