ECE 15, Fall 2019

Midterm

Sequence number	
Last name	
First + middle name(s)	
PID	

Grade

/ 20

Instructions:

- Do not look at the questions or start writing until it is announced you can do so.
- You cannot use any electronic devices.
- This exam is open book, open notes (if they are on paper).
- Write only your final answers in the space provided.

Academic Integrity Statement:

Integrity of scholarship is essential for an academic community. The University expects that both faculty and students will honor this principle and in so doing protect the validity of University intellectual work. For students, this means that all academic work will be done by the individual to whom it is assigned, without unauthorized aid of any kind.

By signing below, I certify that I have completed this exam in accordance to the academic integrity statement.

Signature: ____

Date: ____________________

Question 1 (6 pts)

Complete the question listed in the box to the right of the program. The program does not contain any errors (run-time or compilation). Assume this is a 64-bit system, with integers 4 bytes and doubles 8 bytes. Use ?? for garbage values.

```
#include <stdio.h>

int x;

int main() {
    int *a;
    int b, c = 9;

    a = &b;
    b = 2;
    *a = c / *a;
    a = &c;

    printf("%d \n", --x);
    printf("%d %d ",(*a)++, b);
    printf(“%d \n”, c);
    printf("%1.1f \n", (int)c + 0.5);
}
```

What gets printed?

Question 2 (6 pts)

You must implement a function, called **`modify()`**. This function must have a void return type. To explain what it does, consider the program on the next page.

In each iteration of the for-loop, the variable `letter` contains a new lower-case letter of the alphabet. The function `modify` should change `letter` to contain the letter in the alphabet that comes 1 spot after it (so ‘e’ becomes ‘f’, etc.). The letter ‘z’ is changed to ‘a’. You can choose what parameters the function `modify` takes; it must have a void return type.

Your task:

1. Add a single line of code in `main` in which you call your `modify` function. Write this line in the blue-dashed-line box. You cannot make any other modifications to `main`.

2. Write your `modify` function on the next page below the code box. Assume it will be placed where we added the comment in the code.

As an illustration, on the right, you see the correct outputs for the test data in the starter code. These are just example inputs. Your code should work for any lower-case letters.

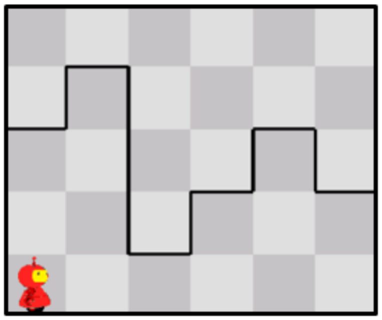
```
letter: e
letter: b
letter: a
letter: c
letter: z
letter: h
letter: a
letter: b
```

```
#include <stdio.h>
// Your modify function will go here

int main() {

    // This is test data
    char inputs[8] = {‘d’, ‘a’, ‘z’, ‘b’, ‘y’, ‘g’, ‘z’, ‘a’};
    char letter;
    int i;
    for (i=0;i<8;i++) {
        letter = inputs[i];
        // Call your modify function



        printf("letter: %c \n", letter);
    }
}
```

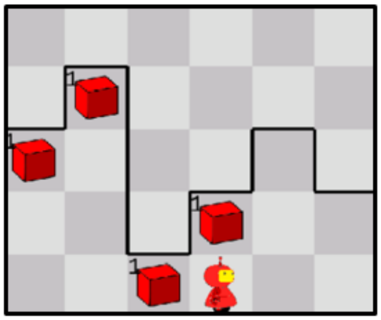
Question 3 (8 pts)

You must program Karel to “fix his ceiling”. We give you the basic skeleton of the code below.

You cannot modify the main function. Your task is to write the fix_ceiling function. You are encouraged to use helper functions.



Start state



End state

Karel starts at the SW corner of the map facing E. In each column, he needs to go up to the top wall and put an item there (i.e., fill a hole in the ceiling). Karel should continue this until he has reached his limit of holes to fill. This total number is given by the value we assigned to `max_holes_to_fill`. We used 4 in the example, but your program should be general enough to take any number. He needs to end at the S end of the last column where he put an item (the direction he is facing is not important).

You may assume Karel never hits the E wall (he will never be asked to fill more holes than there are columns), so no need to check for that. Also, he starts with an infinite number of items in his bag. The bottom row is always open. The repeat() functionality is not available.

Your task: Implement the `fix_ceiling` function on the next page. This function has Karel put an item at the top of his current column. Make sure that the function behaves such that the overall functionality is correctly executed, when called from `main` as shown in the code. Think about how you can make sure he only fixes the correct number of holes and then stops.

```
#include <karel.h>

void turn_around() {
   turn_left();turn_left();
}

// Write the fix_ceiling function

int main() {
   karel_setup("settings/settings01.json");

   int max_holes_to_fill = 4;
   while(1) {
      fix_ceiling(max_holes_to_fill);
      move();
   }
   turn_off();
}
```

---- END ----