CSE 1310: Introduction to Computers & Programming
University of Texas at Arlington
Fall 2020
Dr. Alex Dillhoff

Assignment 7

This assignment will focus on **pseudorandom number generation**.

1. Create a program that generates n values in range $[low, high]$.

   - Create a function `generate_vals` which takes as input an array of double values and three integers representing the size of the array, low value of range, and high value of range, respectively. The function should use `rand()` to generate the requested number of values.
   - Prompt the user to enter the size of their array as show below. It cannot exceed a maximum size of 128. If it does, set the size equal to 128 and continue with the program.
   - Prompt the user to enter a low and high value as show below. These values will be used to limit the range that each generated number can be.
   - Print the generated values such that it matches the format below.
   - Save your code as `prob1.c`.

   Example Run

   ```
   Enter size: 8
   Enter low value: 9
   Enter high value: 18
   16.05, 16.19, 17.20, 10.78, 12.02, 15.91, 11.50, 13.99
   ```

2. Create a randomized version of the Caesar Cipher which encodes and decodes a message given an initial seed. The Caesar Cipher is a rudimentary encoding scheme in which each character is rotated to the right by n places. To decode an encoded message using this scheme, each character in the message is rotated to the left by n places.

   Given a seed for randomization, your encoding function should rotate each character in the message by a random value. Assuming the user has the same seed on their system, they can decode the message using your decoding function.

   - Create an encoding function that takes a string and an integer value as input. The integer will be used to set the seed of the pseudorandom number generator using `srand()`. The encoding function should then generate a random value for each character in the message. That value will be used to alter the input character.

- Create a decoding function that takes a string and an integer value as input. Again, the integer is used as the seed using `srand()`. This function will reverse the encoding done in the previous step.
- Your program should prompt the user for an input message and seed similar to the example run below.
- Only consider ASCII characters with values between 32 and 127 (inclusive) on the ASCII table.
- Save your code as `prob2.c`.

Hints

- When encoding a character, shift it to the left by 32 so that the first usable character has index 0. This will frame your problem as encoding an index into a character array of size (127 - 32).
- Don't forget to shift back when decoding.

Example Run

```
Enter seed value (int): 100
Enter string to encode: test
Encoded message: 3>J}
Decoded message: test
```

3. Create a program stores a list of cities read from a file using redirection. Your program should randomly pick one of the cities and print it to the user.

- Create a function `read_cities` that reads continuous lines of input until the string `"END"` is read. Each line of input that is not `"END"` should be saved in a 2D array that is passed to the function from `main()`. This function will return the number of lines read (excluding the `"END"` line).
- Make sure you strip the newline character from each line of input.
- Generate a random value based on the number of cities read and use it to pick a city from the 2D array of cities.
- Your input and output should look similar to the example run below.
- Use the file `cities.txt` located on Canvas.
- Save your code as `prob3.c`.

Example Run

```
$ ./a.out < cities.txt
Los Angeles, CA
```

4. Create a program that rolls virtual dice. It should support the following dice:

- d4
- d6
- d8

- d10
- d12
- d20

- Use a single function to implement the dice roll. This function will take an integer as input and return an integer reflecting the roll of the dice.
- In `main()`, keep the user in a loop until they are done using the program.
- Your program should show a menu of options similar to the example run which displays a list of options.
- Save your code as `prob4.c`.

Example Run

```
1. Roll d4
2. Roll d6
3. Roll d8
4. Roll d10
5. Roll d12
6. Roll d20
Press Q to quit.
> 6
You rolled a 20!
> q
Thanks for playing!
```

Create a `zip` file using the name template `LASTNAME_ID_A7.zip` which includes ALL of your code files. Submit the `zip` file through Canvas.