Permutations / C programming homework MM - Do before mid-module quiz

The ungraded homework assigned below is never turned in, but should be completed before the mid-module quiz opens.

The graded homework assigned below is due 24 hours before the mid-module quiz opens. No late graded homework is accepted.

The mid-module homework can be done individual or collaboratively. Read the collaboration policy to know what this means.

Ungraded homework

The point of ungraded homework is to develop your abilities and prepare you for the quiz. Solutions will be provided, but they should be consulted only when you need a hint and/or afterward to compare and contrast your solution with mine.

C Programming problems

1. At https://codestepbystep.com do the following problems in the C/bitwise section: ascii_to_hex, binary_to_hex1, bitmask0, bitmask1, bitmask2, bitmask3, bitset1, bitwise1, hex_to_binary1.
2. Study this program: cbc_encrypt.c. For each function call in the program, find its documentation and read it to get a preliminary understanding of what it does. Probably the easiest way to find the documentation is to do an internet search for each function name with the word *man* (eg, "man fopen" and "man EVP_MD_CTX_destroy"). Create a document and for each function write one or two sentences that explain in your own words what each function does.
3. Study this program: cbc_decrypt.c. and find how it is different from cbc_encrypt.c. It has only very minor differences. Briefly describe them.
4. Log onto a computer that has OpenSSL installed and save the two files cbc_encrypt.c and cbc_decrypt.c. Compile each using the commands.

   ```
   gcc -Wall -lcrypto -o cbc_encrypt cbc_encrypt.c
   gcc -Wall -lcrypto -o cbc_decrypt cbc_decrypt.c
   ```

   The `-Wall` commands turn on extra warnings. (It's good practice to eliminate all such warnings when you program.) The `-lcrypto` tells GCC to link your program with

OpenSSL's crypto library. And `-o` tells GCC to name the output something other than `a.out`.

Now experiment by encrypting and then decrypting a file to see that it works. Both `cbc_encrypt` and `cbc_decrypt` take two command line arguments: source and destination file names. So, `cbc_encrypt foo bar` will encrypt the file `foo` and put the result into file `bar` and `cbc_decrypt bar foo2` will reverse the process putting the result into file `foo2`. Look at both output files: the encrypted one should look like random gibberish and the decrypted one should match the original. Be careful with your destination file names; if the file already exists it will be overwritten.

Permutation problems

1. Let Z_n be shorthand for the set containing the n smallest non-negative integers (eg, Z_3 = {0,1,2}). Is f: $Z_5 \rightarrow Z_5$ defined as f(x) = 2x mod 5 an invertible function? If so, what is its inverse (given either as a set of ordered pairs or as a formula). If not, explain in one short sentence. *Note: since this signature is of the form A → A, if it is invertible then it can also be called a permutation or permutation function.*
2. (a) How many functions exist with signature f: $Z_4 \rightarrow Z_5$? (b) Given that a and b are positive integers, how many functions exist with signature f: $Z_a \rightarrow Z_b$? (c) How many permutation functions exist with signature f: $Z_4 \rightarrow Z_5$? (d) How many permutation functions exist with signature f: $Z_4 \rightarrow Z_4$? (e) Given that a and b are positive integers, how many permutation functions exist with signature f: $Z_a \rightarrow Z_b$?
3. Let `rand(n)` be a library function that evaluates to a random integer in Z_n each time it is called (like Java's `Random.nextInt(n)`). Write a method called `createRandomFunction` (right here in your homework) in C, Java, or pseudocode that takes a positive integer n as a parameter and returns an array with n elements each uniformly distributed in Z_n. Essentially I'm asking you to write a method that specifies a random function $Z_n \rightarrow Z_n$ using the table filling method (ie, `a = createRandomFunction(10)` fills `a` with random values and then `a[0]` would tell you what 0 maps to, `a[1]` tells you what 1 maps to, etc.).
4. Do Problem 4 again, but this time name the method `createRandomPermutation` and make the array a permutation (ie, 0 through n-1 each appear exactly once). For full credit, make your method run in O(n) time.
5. Let a and b be integers greater than 2, and let f: $Z_a \rightarrow Z_b$ be a random function. (a) What is Pr[f(0)=0]? Explain. (b) What is Pr[f(1)=1 | f(0)=0]? Explain. (c) What is Pr[f(1)=0 | f(0)=0]? Explain. Let a be an integer greater than 2, and let f: $Z_a \rightarrow Z_a$ be a random permutation function. (d) What is Pr[f(0)=0]? Explain. (e) What is Pr[f(1)=1 | f(0)=0]? Explain. (f) What is Pr[f(1)=0 | f(0)=0]? Explain.

6. A `string` is a concatenation of symbols from an alphabet. For example a string of bits is a concatenation of elements from the alphabet {0,1} and a string of bytes is a concatenation of elements from the alphabet {x | x is an 8-bit string}. A String in Java is the concatenation of elements from the type `char`. (a) Let A be the set of all bytes (ie, all 8-bit strings). How many elements are in A? Explain. (b) Let B be the set of all 8-byte strings (ie, all 64-bit strings). How many elements are in B? Explain.

Ungraded homework solutions

Study these only after completing the homework or after struggling with it for a while.

C programming solutions
Permutation solutions

Graded homework

On Canvas complete the Old MM Quiz that is open now. It closes 24 hours before your mid-module quiz opens.

Each old quiz is worth 1% of your overall grade. It is untimed and you may take it as many times as you want. You may do it alone or in collaboration. It is intended as a warm-up for the actual quiz.