athenahealth – Programming Round' 12

Duration: 1 hour

Instructions to be followed:

- Question No. 1 is mandatory and must be attempted first.
- Save all the programs inside a folder named "<YourFullName>_<Course>_<DeptName>_<CollegeName>" on your desktop.
- Ex: Ramesh_BTech_IT_CEG
    - Programs should be named like Program1.c/Program1.cpp, Program2.c/Program2.cpp etc.

---

1. Given two numbers N and K (K <= N) determine how many integers 1 to N have the pattern 'K'.

   Input: A single with N and K separated by a single space. 1 <= N <= 10000.
   Output: The number of integers that contain pattern K.

   Sample Input:
   902 90

   Sample Output:
   12

   Explanation: The pattern '90' occurs in 90, 190, 290, 390, 490, 590, 690, 790, 890, 900, 901 and 902.

2. You are going to write a letter to your friend. You do not want anybody else to be able to read it. You decide to use a simple substitution cipher to encode the message. Each letter in the message is replaced with its corresponding letter in a substitution alphabet. A substitution alphabet is a permutation of all the letters in the original alphabet, i.e. 'a'-'z'. In this problem, the alphabet will consist of only lowercase letters ('a' - 'z'). For example, if your message is "hello" and your cipher maps 'h' to 'q', 'e' to 'w', 'l' to 'e' and 'o' to 'r', the encoded message will be "qweer". Given the original message, determine a cipher that will produce the encoded string that comes earliest alphabetically. Return this encoded string.

   Input: A single line with the original message. A message will consist of one or more words. Each word will be separated by a single space. All characters will be in lower-case.
   Output: On a single line, the encoded message that comes first alphabetically.

   Sample Input:
   hello world

Sample Output:
abccd edfcg

Explanation: Of all the possible ciphers that you could use to encode your message, the one that gives the alphabetically first encoding is: 'd' -> 'g'; 'e' -> 'b'; 'h' -> 'a'; 'l' -> 'c'; 'o' -> d' and 'r' -> 'f'.

3. We designed a storage device to hold three types of data: a character that needs 1-byte; an integer that needs 2-bytes and a double that needs 4-bytes. We have a set of N elements (that includes one, two or all three of the data types) to be stored in the device. There is only some space left in the device (we've already used up some space!). We know that the available space will exactly hold the N elements. Given the starting (A) and ending (B) addresses of the available space, your task is to determine the number of ways we can store those N elements.

   Input: A single line with 3 positive integers separated by spaces. The first integer represents N. Then, addresses A and B follow; A < B. A and B will be positive integers represented in decimal system only (we made the addresses human-readable!). Also, each memory location represented by an address can hold a byte of data.
   Output: The number of ways we can store N items in the space available.

   Sample Input:
   10 4 29

   Sample Output:
   4530

   Explanation: The available space is 26 bytes as (29 - 4 + 1 = 26). We have to store 10 elements. There are only 3 possible cases we can have the 10 elements fit in 26 bytes.

   - 0 characters, 7 integers and 3 doubles, making a total of 10 elements and a total space of (0 * 1 + 7 * 2 + 3 * 4) = 26. The number of ways you can arrange these 7 integers and 3 doubles is 120.

   - 2 characters, 4 integers and 4 doubles, making a total of 10 elements and a total space of (2 * 1 + 4 * 2 + 4 * 4) = 26. The number of ways you can arrange these 2 characters, 4 integers and 4 doubles is 3150.

   - 4 characters, 1 integers and 5 doubles, making a total of 10 elements and a total space of (4 * 1 + 1 * 2 + 5 * 4) = 26. The number of ways you can arrange these 4 characters, 1 integer and 5 doubles is 1260.

   There are no other possible set of 10 elements that would fit in 26 bytes.