CSE1133 Introduction to Programming

2022-2023 Fall Semester Homework-2

November 22, 2022

Deadline: December 2, 2022 (23:50)

Announcement

- **Your implementations should be efficient as possible. Write your own algorithms!**
- **You can prepare your homework yourself (single person).**
- **Your source code must have comment line which explain your algorithms.**
- **Your class name must be "HW + your student ID", i.e. "HW220315000.java"**
- **Upload just single .java file, not all project, word document, or zip file etc.**
- **Write your name, surname, and student ID at the beginning of code as comment lines.**
- **Write all your work in a single main method.**
- **There are two different assignments in Microsoft Teams for formal education/ night education. Upload your work only to related assignment.**

Question 1: Classical Cryptography (30 pts)

One of the earliest known uses of cryptography was by Julius Caesar. He made messages secret by shifting each letter three letters forward in the alphabet (sending the last three letters of the alphabet to the first three). For instance, using this scheme the letter B is sent to E and the letter X is sent to A. This is an example of **encryption**, that is, the process of making a message secret.

To express Caesar's encryption process mathematically, first replace each letter by an element of $\mathbf{Z}_{26}$, that is, an integer from 0 to 25 equal to one less than its position in the alphabet. For example, replace A by 0, K by 10, and Z by 25. Caesar's encryption method can be represented by the function f that assigns to the nonnegative integer p, $p \leq 25$, the integer $f(p)$ in the set $\{0, 1, 2, \ldots, 25\}$ with

$$f(p) = (p + 3) \textbf{ mod } 26.$$

In the encrypted version of the message, the letter represented by p is replaced with the letter represented by $(p + 3)$ **mod** 26.

To recover the original message from a secret message encrypted by the Caesar cipher, the function f^{-1}, the inverse of f, is used. Note that the function f^{-1} sends an integer p from $\mathbf{Z}_{26}$, to $f^{-1}(p) = (p - 3)$ **mod** 26. In other words, to find the original message, each letter is shifted back three letters in the alphabet, with the first three letters sent to the last three letters of the alphabet. The process of determining the original message from the encrypted message is called **decryption**.

For example:

The message is "MEET YOU IN THE PARK"

First replace the letters in the message with numbers. This produces

12 4 4 19   24 14 20   8 13   19 7 4   15 0 17 10.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Now replace each of these numbers p by $f(p) = (p + 3) mod\ 26$. This gives

15 7 7 22   1 17 23   11 16   22 10 7   18 3 20 13.

Translating this back to letters produces **the encrypted message "PHHW BRX LQ WKH SDUN".**

the decrypted message is "MEET YOU IN THE PARK".

Write a Java program that reads a message from the keyboard. If the message contains non-letters, ask the user for another message until a valid message is entered. The program should compute <u>the encrypted version</u> of the input message using the algorithm above.

<u>Question 2: (30 pts)</u>

Write a java program that shows the following outputs.

Output-1	**Output-2**	**Output-3**
1	12345	1
23	2345	1 9
345	345	1 9 25
4567	45	1 9 25 49
56789	5	1 9 25 49 81
		1 9 25 49
		1 9 25
		1 9
		1

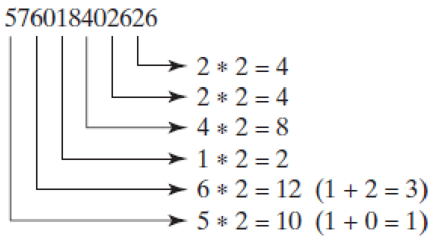
<u>Question-3: Examine a valid account number (40 pts)</u>

Write a program that prompts the user to enter an account number. Display whether the number is valid or invalid according to the following situations.

Account numbers follow certain patterns.

1) An account number must have 16 characters. It must start with two capital letters but the first letter must always be lower than the second. (for example: AB, CH, AZ, etc.)
2) Two digits after the letters must be prime numbers.
3) For the next 12 digits:

a) Double every second digit from right to left. If doubling of a digit results in a two-digit number, add up the two digits to get a single-digit number.



b) Now add all single-digit numbers from Step a.

4 + 4 + 8 + 2 + 3 + 1 = 22

c) Add all digits in the odd places from right to left in the card number.

6 + 6 + 0 + 8 + 0 + 7 = 27

d) Sum the results from Step b and Step c.

22 + 27 = 49

e) If the result from Step d is divisible by 10, the account number is valid; otherwise, it is invalid.

For example:

The account number AC11576018412626 is a valid.