COP-4338 System Programming

Programming Assignment 2: Pointers

FIU Knight Foundation School of Comp. & Info Sciences

In this assignment, you will write a program that performs encryption/decryption using a very simple transposition technique inspired by the Rail Fence Cipher. Let's see an example to illustrate how it works:

Assume that we are going to encrypt the following plain-text message before sending it out to provide some level of security:

```
The war starts at seven AM.
Bring water and canned food.
Don't attack from West.
Play music when you arrive.
```

The first step is to separate the words in each sentence by white space character and place all the words in a table like this:

0	1	2	3	4	5
the	war	starts	at	seven	am
bring	water	and	canned	food	
don't	attack	from	west		
play	music	when	you	arrive	

As you see, every column of this table is labeled with a non-negative integer. Then, we use a given permutation of the column labels like $(3, 5, 2, 1, 0, 4)$ to obtain the encrypted text (called the cipher-text). The given permutation is called *the encryption key.* To obtain the cipher-text, we need to iterate through columns of this table in the order specified by the key. For each column, we list the words in the column from top to bottom. Since the number of words in each row may be different, you may find some missing cells in a column. In such cases, we use the filler word "null" (e.g. the words in column 4 are: "seven, food, null, arrive"). Therefore, the cipher-text in this example will be:

```
at canned west you am null null null starts and from when war
water attack music the bring don't play seven food null arrive
```

The recipient of this message can decrypt the above cipher-text easily as long as they know what the key is (in this example, $(3, 5, 2, 1, 0, 4)$). Here are the steps for decryption and obtaining plain-text from cipher-text:

1. Count the number words n in the cipher-text (24 words in this example).
2. Divide the number of words ($n = 24$) by the length of given key ($k = 6$): $n/k = 4$.
3. Draw a table with n/k rows and k columns
4. Place words in the columns of drawn table using the order given by the key.
5. Extract each sentence from each row of the table after removing the filler (null) words.

1 Program Input Commands

There are three valid commands for your program:

- encrypt $(a_0, a_1, \ldots, a_{k-1})$ ↩ PLAINTEXT Ctrl D: encrypts the given plain-text using the given key which is a permutation of the first k non-negative integers. The plain-text starts with a new line character and ends with an EOF character which is entered via keyboard using Ctrl+D. This command prints the encrypted text on the screen. For the sake of simplicity, assume that:
  - Plain-text is made or one or more sentences (at-most 5000 sentences).
  - Every sentence ends with a dot (.).
  - Two consecutive sentences are separated by a single new line character.
  - Each sentence is made of at-most 100 words separated by white-space.
  - Each word in a sentence is made of only lowercase alphabetical letters or an apostrophe. Length of each word is at-most 24 characters.
  - No punctuation marks are used in the plain-text except apostrophe and dot.
- decrypt $(a_0, a_1, \ldots, a_{k-1})$ ↩ CIPHERTEXT Ctrl D: decrypts the given cipher-text using the given key which is a permutation of the first k non-negative integers. The cipher-text starts with a new line character and ends with an EOF character which is entered via keyboard using Ctrl+D. This command prints the decrypted text on the screen. For the sake of simplicity, assume that the cipher-text contains no new line character.
- quit: ends the program.

2 Submissions

You need to submit a *.zip* file compressing the followings:

- all of .c and .h files of the program,
- Makefile which organizes the compilation process of your program.