ADDITIONAL LAB QUESTIONS (SOLUTIONS WILL NEVER BE SHARED!!!)

1.
Palindrome: "A palindrome is a word, phrase, number, or other sequence of characters which reads the same backward or forward, such as 'madam' or 'kayak'" (excerpt from Wikipedia).

Write a program that reads 8 words from Palindrome.txt and checks whether the word you read is a palindrome (please read its description above) or not. Print the index numbers of all the palindrome words in your array. You are required to read words character by character into a character array.

- Write **readFile** function, which gets a file pointer and a two-dim array, to read each word (bunch of characters) line by line and store them into a two-dim character array.
- Write **findSize** function takes a one-dim array and returns its actual size.
- Write **isPalindrome** function takes a one-dim character array to check whether the content is a polindrome or not. If the word (character squence) is a palindrome, then it returns 1. Otherwise, it returns 0.

Palindrome.txt

```
anna
araba
civic
kayak
merkez
level
madam
nazan
```

```
Example run:
INDEXES:
0
Not palindrome
2
3
Not palindrome
5
6
7
```

Project_name: Lab1_AQ1
File_name: AQ1.cpp

2.

- Write a function **dot_product** that gets two one-dim arrays and finds their dot product which is calculated by using following formula: $\Sigma A[i]*C[i]$ where A and C are the one dim arrays.
- Write a function **disp-one** that displays the elements of a one-dimensional array.
- Write a function **disp-two** that displays the elements of a two-dimensional array with 4 columns.

Write a program that reads a 4x4 matrix and a 4 element vector from a text file, and and finds their product as in the following example:

$$
\begin{bmatrix} 5 & 11 & 9 & 16 \\ 2 & 4 & 3 & 1 \\ 9 & 12 & 8 & 7 \\ 6 & 1 & 4 & 10 \end{bmatrix}
*
\begin{bmatrix} 3 \\ 6 \\ 12 \\ 4 \end{bmatrix}
=
\begin{bmatrix} 5*3 + 11*6 + 9*12 + 16*4 \\ 2*3 + 4*6 + 3*12 + 1*4 \\ 9*3 + 12*6 + 8*12 + 7*4 \\ 6*3 + 1*6 + 4*12 + 10*4 \end{bmatrix}
$$

Notice that each row of the result matrix is the dot product of one row of the matrix with the vector. Thus, you need to copy each row of the matrix to a one dim-array before using the **dot_product** function. The output should be displayed as in the example run, using the functions **disp_one** and **disp_two(..).**

Example Run:
```
Matrix is
5   11    9   16
2    4    3    1
9   12    8    7
6    1    4   10

Vector is
3 6 12 4

The product is
253
70
223
112
```