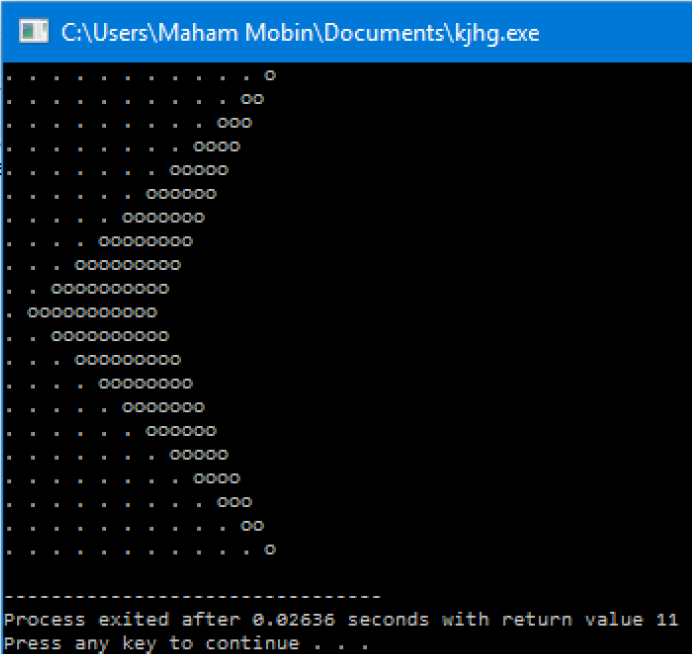
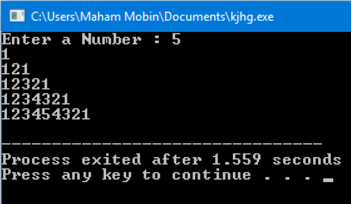
**LAB TASK 9:-**

**Task 1:-**Write a function that takes N as argument and print shapes. Shape A) should have 2N+1 rows as shown below here N=10.

A)



B)



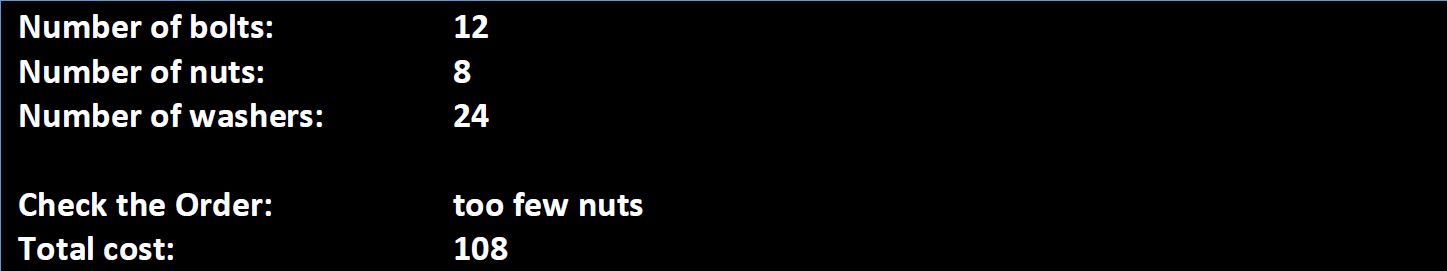
**Task 2:-**

Bob's Discount Bolts charges the following prices:

* **5 cents per bolt**
* **3 cents per nut**
* **1 cent per washer**

Write a program that asks the user for the number of bolts, nuts, and washers in their purchase and then calculates and prints out the total. As an added feature, the program checks the order. A correct order must have at least as many nuts as bolts and at least twice as many washers as blots, otherwise the order has an error. For an error the program writes out "Check the Order: too

few nuts" or "Check the Order: too few washers" as appropriate. Both error messages are written if the order has both errors. If there are no errors the program writes out "Order is OK."



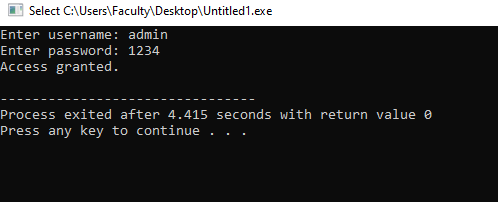
**Task 3:-**

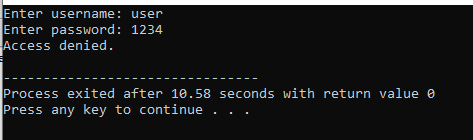
Write a program to verify if a user-entered username and password match a pre-stored username and password.

**Instructions:**

1. Define a pre-set username (admin) and password (1234) in the program.
2. Ask the user to enter their username and password.
3. Use strcmp to check if the entered username and password match the pre-stored values.
4. If both match, print "Access granted". If either does not match, print "Access denied".

**Sample output:**

****

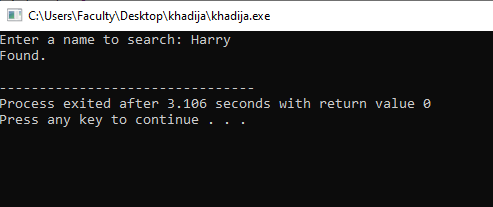
****

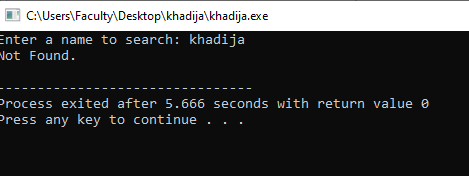
**Task 4:-**

Write a C program that initializes a 2D character array with a list of student names. Then, take a name as input from the user and check if it exists in the array. Display "Found" if the name is there, otherwise display "Not Found".

**Instructions:**

1. Initialize a 2D character array with at least five student names (e.g., "Alice", "Harry", "Charlie", "Diana", "Eve").
2. Prompt the user to enter a name.
3. Use a loop to search for the name in the array.
4. If the name is found, print "Found". If not, print "Not Found".





**Task 5:**

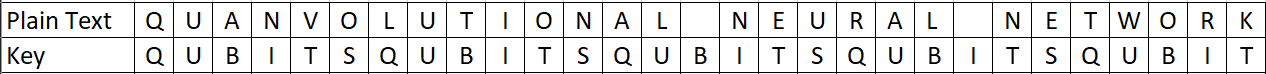
Substitution ciphers replace an alphabet with another based on a fixed or a variable rule. One of the most famous substitution ciphers is the Vigenère cipher.

Vigenère cipher requires a Key to encrypt the text. A key can be any word or phrase that is repeated until the whole plain text is covered.

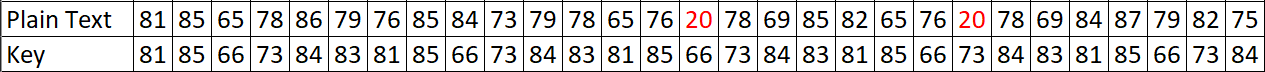
Let's suppose our Plain text is:   
“Quanvolutional Neural Networks”

And the Key is:  
“Qubit”

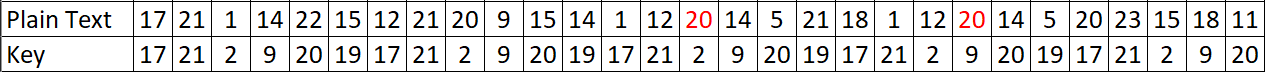
We will first Map the Key onto our Plain text like this:

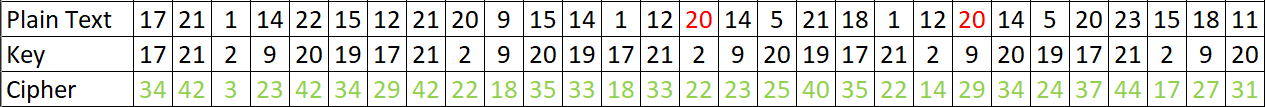


We will then figure out the underlying decimal value of all the characters:  
(space = 20)

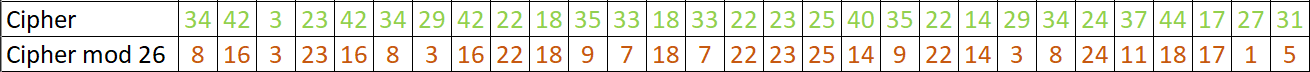


Subtract 64 from each value:  
(ignore subtraction for space value)

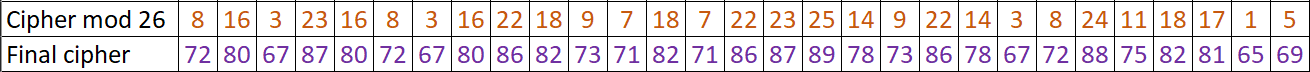


Add the values vertically:  


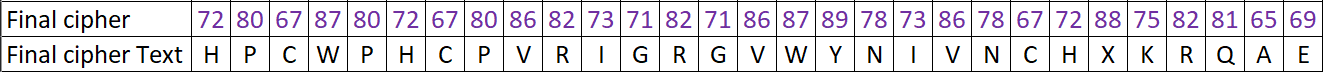
Take their modulo by 26 (i.e. (32 mod 26) = 6 )



Add 64 :



Convert the values back to their Character Representation:



Finally, Our Ciphertext would be : “HPCWPHCPVRIGRGVWYNIVNCHXKRQAE”

You are required to prompt the user for plaintext and Key. Apply the Vigenère cipher and show the resultant cipher text.