In [65]:
```python
# 1. Write a program that asks the user to enter a string. The program should then print the following:
#(a) The total number of characters in the string
#(b) The string repeated 10 times
#(c) The first character of the string (remember that string indices start at 0) (d) The first three
# characters of the string
#(e) The last three characters of the string
#(f) The string backwards
#(g) The seventh character of the string if the string is long enough and a message otherwise
#(h) The string with its first and last characters removed
#(i) The string in all caps
#(j) The string with every a replaced with an e



user=input("Enter a string: ")

print("Total number of characters in the string {}:".format(len(user)))

print("String repeated 10 times {}:".format(user*10))

print("total number of characters in the string{}:".format(user[0]))

print("the First three characters of the string{}:".format(user[:3]))

print("the Last three characters of the string{}:".format(user[3:]))

print("String backwards {}:".format(user[::-1]))

if len(user)>=7:
    print("the Seventh character of the string{}:".format(user[6]))
else:
    print("The string is not enough long:")
print("String with first and last characters removed{}:".format(user[1:-1]))

print("String in all caps{}:".format(user.upper()))

print("String with every a replaced with an e {}:".format(user.replace('a','E')))
```

```
Enter a string: keerthi
Total number of characters in the string 7:
String repeated 10 times keerthikeerthikeerthikeerthikeerthikeerthikeerthikeerthikeerthikeerthi:
total number of characters in the stringk:
the First three characters of the stringkee:
the Last three characters of the stringrthi:
String backwards ihtreek:
the Seventh character of the stringi:
String with first and last characters removedeerth:
String in all capsKEERTHI:
String with every a replaced with an e keerthi:
```

In [47]:
```python
# 2. A simple way to estimate the number of words in a string is to count the number of spaces in the string.write a program
#  that asks the user for a string and returns an estimate of how many words are in the string.
# Tip: You need to count the number of words using spaces

def estimate_word_count():
    user_input = input("Enter a string: ")
    word_count = user_input.count(' ') + 1  # Assuming words are separated by spaces
    print(f"Estimated word count: {word_count}")

# Run the program
estimate_word_count()
```

```
Enter a string: "welcome"
Estimated word count: 1
```

In [14]:
```python
# 3.Write a program that asks the user to enter a word and prints out whether that word contains any vowels

def contains_vowels(word):
    vowels = "aeiouAEIOU"
    for char in word:
        if char in vowels:
            return True
    return False

def main():
    user_input = input("Enter a word: ")
    if contains_vowels(user_input):
        print("The word contains vowels.")
    else:
        print("The word does not contain any vowels.")

# Run the program
main()
```

```
Enter a word: "hyderabad"
The word contains vowels.
```

In [16]:
```python
# 4. Improvise above code by providing unique vowels

def contains_unique_vowels(word):
    vowels = set("aeiouAEIOU")
    word_set = set(word)
    common_vowels = vowels.intersection(word_set)
    return bool(common_vowels)

def main():
    user_input = input("Enter a word: ")
    if contains_unique_vowels(user_input):
        print("The word contains unique vowels.")
    else:
        print("The word does not contain any unique vowels.")

# Run the program
main()
```

```
Enter a word: "fly"
The word does not contain any unique vowels.
```

In [18]:
```python
# 5. Write a program that asks the user to enter a string. The program should create a new string called new_string from the
#  user’s string such that the second character is changed to an asterisk and three exclamation points are attached to the end
# of the string. Finally, print new_string.
# Typical output is shown below:
# Enter your string: Qbert
# Output: Q*ert!!!

def modify_string(input_string):
    if len(input_string) >= 2:
        new_string = input_string[:1] + '*' + input_string[2:] + '!!!'
        return new_string
    else:
        return "Please enter a string with at least two characters."

def main():
    user_input = input("Enter your string: ")
    result_string = modify_string(user_input)
    print("Output:", result_string)

# Run the program
main()
```

```
Enter your string: Qbert
Output: Q*ert!!!
```

In [20]:
```python
# 6. Write a program that asks the user to enter a word and determines whether the word is a palindrome or not.A palindrome is
# a word that reads the same backwards as forwards

def is_palindrome(word):
    reversed_word = word[::-1]
    return word == reversed_word

def main():
    user_input = input("Enter a word: ")
    if is_palindrome(user_input):
        print("The word is a palindrome.")
    else:
        print("The word is not a palindrome.")

# Run the program
main()
```

```
Enter a word: "level"
The word is a palindrome.
```

In [26]:
```python
# 7. At a certain school, student email addresses end with @student.college.edu, while professor email addresses end with @prof.
# .college.edu. Write a program that first asks the user how many email addresses they will be entering,and then has has the
#  user enter those addresses. After all the email addresses are entered,the program should print out a message indicating
#  either that all the addresses are student addresses or that there were some professor addresses entered.



def check_email_addresses():
    num_addresses = int(input("How many email addresses will you be entering? "))

    student_domain = "@student.college.edu"
    professor_domain = "@prof.college.edu"

    student_count = 0
    professor_count = 0

    for _ in range(num_addresses):
        email = input("Enter an email address: ")
        if email.endswith(student_domain):
            student_count += 1
        elif email.endswith(professor_domain):
            professor_count += 1

    if professor_count == 0:
        print("All addresses are student addresses.")
    else:
        print(f"There were some professor addresses entered ({professor_count} professor(s)).")

# Run the program
check_email_addresses()
```

```
How many email addresses will you be entering? 1
Enter an email address: endswith(student_domain)
All addresses are student addresses.
```

In [27]:
```python
# 8. Write a program that asks the user to enter a string, then prints out each letter of the string
# doubled and on a separate line. For instance,
# if the user entered HEY,
# the output would be
# HH
# EE
# YY

def double_and_print_letters(user_input):
    for char in user_input:
        print(char * 2)

def main():
    user_input = input("Enter a string: ")
    double_and_print_letters(user_input)

# Run the program
main()
```

```
Enter a string: "thank you"
""
tt
hh
aa
nn
kk

yy
oo
uu
""
```

In [29]:
```python
# 9.Write a program that asks the user to enter a word that contains the letter a. The program should
# then print the following two lines: On the first line should be the part of the string up to and
# including the the first a, and on the second line should be the rest of the string.
# Sample output is shown below:
# Enter a word: buffalo
# buffa
# lo

def split_and_print(word):
    index_of_a = word.find('a')
    if index_of_a != -1:
        first_part = word[:index_of_a + 1]
        second_part = word[index_of_a + 1:]
        print(first_part)
        print(second_part)
    else:
        print("The word doesn't contain the letter 'a'.")

# Run the program
user_word = input("Enter a word: ")
split_and_print(user_word)
```

```
Enter a word: "congratulations"
"congra
tulations"
```

In [30]:
```python
# 10. Write a program that asks the user to enter a word and then capitalizes every other letter of that
# word.
# So if the user enters rhinoceros,
# the program should print rHiNoCeRoS.

def capitalize_every_other(word):
    result = ''
    for i, char in enumerate(word):
        if i % 2 == 1:  # Capitalize every other letter (indexing starts from 0)
            result += char.upper()
        else:
            result += char
    return result

# Run the program
user_word = input("Enter a word: ")
capitalized_word = capitalize_every_other(user_word)
print(capitalized_word)
```

```
Enter a word: "happy"
"HaPpY"
```

In [32]:
```python
# 11. Write a program that asks the user to enter two strings of the same length. The program should then check to see if the
# strings are of the same length. If they are not, the program should print an appropriate message and exit.if they are of the
# same length, the program should alternate the characters of the two strings.for example,
# if the user enters abcde and ABCDE
# the program should print out AaBbCcDdEe.

def alternate_strings(str1, str2):
    if len(str1) != len(str2):
        print("Both strings should be of the same length.")
        return

    result = ''
    for char1, char2 in zip(str1, str2):
        result += char1 + char2

    print(result)

# Run the program
user_str1 = input("Enter the first string: ")
user_str2 = input("Enter the second string: ")
alternate_strings(user_str1, user_str2)
```

```
Enter the first string: ZYX
Enter the second string: ABC
ZAYBXC
```

In [34]:
```python
# 12. Write a program that asks the user to enter their name in lowercase and then capitalizes the first letter of each word of
# their name.

list1=['geetha reddy','bhargavi reddy']
output=[]
for i in list1:
    output.append(i.capitalize())
output

output1=[i.capitalize() for i in list1]
output1
```

Out[34]:
```
['Geetha reddy', 'Bhargavi reddy']
```

In [37]:
```python
# 13. The goal of this exercise is to see if you can mimic the behavior of the in operator and the count and index methods using
# only variables, for loops, and if statements.
# (a) Without using the in operator, write a program that asks the user for a string and a letter and prints out whether or not
# the letter appears in the string.
# (b) Without using the count method, write a program that asks the user for a string and a letter and counts how many
# occurrences there are of the letter in the string.
# (c) Without using the index method, write a program that asks the user for a string and a letter and prints out the index of
# the first occurrence of the letter in the string. If the letter is not in the string,the program should say so.


# a

def letter_in_string():

    user_string = input("Enter a string: ")
    user_letter = input("Enter a letter: ")

    letter_found = False
    for char in user_string:
        if char == user_letter:
            letter_found = True
            break

    if letter_found:
        print(f"The letter '{user_letter}' appears in the string.")
    else:
        print(f"The letter '{user_letter}' does not appear in the string.")

letter_in_string()

# b

def count_occurrences():

    user_string = input("Enter a string: ")
    user_letter = input("Enter a letter: ")

    count = 0
    for char in user_string:
        if char == user_letter:
            count += 1

    print(f"The letter '{user_letter}' appears {count} times in the string.")

count_occurrences()

# c


def find_index():

    user_string = input("Enter a string: ")
    user_letter = input("Enter a letter: ")

    index = -1
    for i, char in enumerate(user_string):
        if char == user_letter:
            index = i
            break

    if index != -1:
        print(f"The index of the first occurrence of '{user_letter}' is {index}.")
    else:
        print(f"The letter '{user_letter}' is not in the string.")


find_index()
```

```
Enter a string: "hello"
Enter a letter: a
The letter 'a' does not appear in the string.
Enter a string: "hai"
Enter a letter: h
The letter 'h' appears 1 times in the string.
Enter a string: "fine"
Enter a letter: o
The letter 'o' is not in the string.
```

In [40]:
```python
# 14. Finding a substring within a string for example,if we were presented a series of lines formatted as followa
# From stephen.marquard@uct.ac.za Sat Jan 5 09:14:16 2008
# and we wanted to pull out only the second half of the address (i.e., uct.ac.za)


line = "From stephen.marquard@uct.ac.za Sat Jan 5 09:14:16 2008"

at_symbol_index = line.find('@')

second_half = line[at_symbol_index + 1:]

print("Second half of the address:", second_half)
```

```
Second half of the address: uct.ac.za Sat Jan 5 09:14:16 2008
```

In [42]:
```python
# 15. Write a Python program to add 'ing' at the end of a given string (length should be at least 3). If the given string alread
# ends with 'ing' then add 'ly' instead.
# If the string length of the given string is less than 3, leave it unchanged.
# Go to the editor
# Sample String : 'abc'
# Expected Result : 'abcing'
# Sample String : 'string'
# Expected Result : 'stringly'


def modify_string(s):
    if len(s) < 3:
        result = s
    elif s[-3:] == 'ing':
        result = s + 'ly'
    else:
        result = s + 'ing'

    return result

sample_string1 = 'abc'
sample_string2 = 'string'

result1 = modify_string(sample_string1)
result2 = modify_string(sample_string2)

print("Sample String 1:", sample_string1)
print("Expected Result 1:", result1)

print("\nSample String 2:", sample_string2)
print("Expected Result 2:", result2)
```

```
Sample String 1: abc
Expected Result 1: abcing

Sample String 2: string
Expected Result 2: stringly
```

In [43]:
```python
# 16. Take the following Python code that stores a string:‘
# string = 'X-DSPAM-Confidence: 0.8475'
# Extract the portion of the string after the colon character and then use the float function to convert
# the extracted string into a floating point number.


string = 'X-DSPAM-Confidence: 0.8475'

parts = string.split(':')

confidence_str = parts[1].strip()

confidence_float = float(confidence_str)

print("Extracted Confidence as Float:", confidence_float)
```

```
Extracted Confidence as Float: 0.8475
```

In [ ]:

In [ ]:

In [ ]: