```python
# 1.1 Count the number of times a word occurs in a paragraph
def count_word_frequency(paragraph, target):
    words = paragraph.split() # splitting the paragraph
    word_count = 0 # initialising a variable
    for word in words: # checking each target word in the splitted paragraph
        cleaned_word = word.strip(".,!?\"'()").lower()
        if cleaned_word == target.lower(): #Comparing  the word with the
target word and update the counter
            word_count += 1  # incrementing the word count

    return word_count #returning the total count

paragraph = """ A payroll management system is software that enables employers
to give salaries to their employees. It is a medium of showing their
commitment towards the faculty members and fulfilling their obligations along
with keeping financial records in order.
    A payroll management system is an online solution that helps the
institutes to pay salaries to the faculty members working there. It is a way
how they show their commitment to the faculty in form of benefits, appraisals,
and paid leaves. It enables them to fulfill their obligations as stated by the
government and keep financial records in order.
    """ # Paragraph in here
target_word = input("Enter the word to find its frequency in the paragraph:
")  #taking input to find the target word
frequency = count_word_frequency(paragraph, target_word) # Call the function
to find the frequency of the target word
print(f"The word '{target_word}' appeared --{frequency}-- time(s) in the given
paragraph.")



# 1.2 Lines, words, and characters of a paragraph
print("\n--- Lines, words, and characters of a paragraph ---")
length=len(paragraph)
print(f"Number of characters: {length}") # displays the number of characters
in a paragraph
para_words=paragraph.split()
plength=len(para_words)
print(f"Number of words: {plength}") #displays number of words in the
paragraph
lines = paragraph.split('\n')
lines_length=len(lines)
print(f"Number of lines: {lines_length}") # displays number of lines divided
by '\n'


# 1.3 arrange words in an alphabetical order
```

```python
def sorted_word(word):
    sorted_word = ''.join(sorted(word))
    return sorted_word
print("\n\n\n ----Arrage a word alphabetically---- \n")
word = input("Enter a word for arranging it alphabetically: ")
sorted_word = sorted_word(word)
print(f"The word '{word}' arranged alphabetically is: {sorted_word}")
```

OUTPUT-1:

```
Enter the word to find its frequency in the paragraph: payroll
The word 'payroll' appeared --2-- time(s) in the given paragraph.

--- Lines, words, and characters of a paragraph ---
Number of characters: 608
Number of words: 97
Number of lines: 3



 ----Arrage a word alphabetically----

Enter a word for arranging it alphabetically: Management
The word 'Management' arranged alphabetically is: Maaeegmnnt
```

OUTPUT-2:

```
Enter the word to find its frequency in the paragraph: employee
The word 'employee' appeared --0-- time(s) in the given paragraph.

--- Lines, words, and characters of a paragraph ---
Number of characters: 608
Number of words: 97
Number of lines: 3



 ----Arrage a word alphabetically----

Enter a word for arranging it alphabetically: payroll
The word 'payroll' arranged alphabetically is: allopry
```

```python
#2. Encryption of a string with a encryption key
def encrypt_string(string, encryption_key):
    encrypted_string = ""
    for char in string:
        if char.isalpha():
            # Check if the character is uppercase or lowercase
            is_uppercase = char.isupper()
            # Convert the character to its corresponding ASCII value
            ascii_value = ord(char)
            # Add 'n' to the ASCII value to encrypt the character
            encrypted_ascii = ascii_value + encryption_key
            # Adjust the ASCII value to keep it within the alphabet range
            if is_uppercase:
                encrypted_ascii = (encrypted_ascii - 65) % 26 + 65
            else:
                encrypted_ascii = (encrypted_ascii - 97) % 26 + 97
            # Convert the encrypted ASCII value back to a character
            encrypted_char = chr(encrypted_ascii)
            encrypted_string += encrypted_char
        else:
            # If the character is not an alphabet, keep it as it is
            encrypted_string += char

    return encrypted_string


string = input("Enter a string to encrypt: ")
encryption_key = int(input("Enter the value of 'n' (encryption key values):
"))

encrypted_domain = encrypt_string(string, encryption_key)
print(f"Encrypted domain name: {encrypted_domain}")
```

OUTPUT-1:

```
Enter a string to encrypt: bat
Enter the value of 'n' (encryption key values): 3
Encrypted domain name: edw
```

OUTPUT-2:

```
Enter a string to encrypt: employee
Enter the value of 'n' (encryption key values): 5
Encrypted domain name: jruqtdjj
```

OUTPUT-3:

```
Enter a string to encrypt: Password
Enter the value of 'n' (encryption key values): 687
Encrypted domain name: Alddhzco
```

```python
#3.Arranging List of tuples
data = [
('main st.', 4, 4000),
('elm st.', 1, 1200),
('pine st.', 2, 1600)
]

# Sort in ascending order by first numeric value (index 1)
sorted_by_first_numeric = sorted(data, key=lambda x: x[1])
print("Sorted in ascending order by first numeric value:")
print(sorted_by_first_numeric)

# Sort in descending order by second numeric value (index 2)
sorted_by_second_numeric = sorted(data, key=lambda x: x[2], reverse=True)
print("\nSorted in descending order by second numeric value:")
print(sorted_by_second_numeric)

# Sort in alphabetical order of string value (index 0)
sorted_alphabetically = sorted(data, key=lambda x: x[0])
print("\nSorted in alphabetical order of string value:")
print(sorted_alphabetically)
```

OUTPUT-1:

```
Sorted in ascending order by first numeric value:
[('elm st.', 1, 1200), ('pine st.', 2, 1600), ('main st.', 4, 4000)]

Sorted in descending order by second numeric value:
[('main st.', 4, 4000), ('pine st.', 2, 1600), ('elm st.', 1, 1200)]

Sorted in alphabetical order of string value:
[('elm st.', 1, 1200), ('main st.', 4, 4000), ('pine st.', 2, 1600)]
```

```python
# 4. function to calculate the wages pay function
def pay(hourly_wage, hours_worked):
    regular_hours = min(hours_worked, 40)
    overtime_hours = max(0, hours_worked - 40)

    regular_pay = regular_hours * hourly_wage
    overtime_pay = overtime_hours * (hourly_wage * 1.5)

    total_pay = regular_pay + overtime_pay
    return total_pay

wage=int(input("Enter the wage per hour: "))
hours=int(input("Enter the number of hours worked: "))
print(pay(wage,hours))
```

OUTPUT-1:

```
Enter the wage per hour: 10
Enter the number of hours worked: 40
400.0
```

OUTPUT-2:

```
Enter the wage per hour: 500
Enter the number of hours worked: 24
12000.0
```