1. Write a program that defines a function count_lower_upper() that accepts a string and calculates the number of uppercase and lowercase alphabets in it. It should return these values as a dictionary. Call this function for some sample string.

INPUT

```
def count_lower_upper(string):
    result = {"lowercase": 0, "uppercase": 0}
    for char in string:
        if char.islower():
            result["lowercase"] += 1
        elif char.isupper():
            result["uppercase"] += 1
    return result

sample_string = "Hello World"
print(count_lower_upper(sample_string))
```

OUTPUT

```
{'lowercase': 8, 'uppercase': 2}
```

2. Write a program that defines a function compute() that calculates the value of n + nn + nnn + nnnn, where n is digit received by the function. test the function for digits 4 to 7.3. Write a program that defines a function create_array() to create and return a 3D array whose dimensions are passed to the function. Also initialize each element of this aray to a value passed to the function. e.g. create_array(3,4,5,n) where first three arguments are 3D array dimensions and 4th value is for initialing each value of the 3D array.

INPUT

```
def compute(n):
    result = n + int(str(n)*2) + int(str(n)*3) + int(str(n)*4)
    return result

for num in range(4, 8):
    print(f"{num}: {compute(num)}")
```

OUTPUT

```
4: 4936
5: 6170
6: 7404
7: 8638
```

3.Write a program that defines a function create_array() to create and return a 3D array whose dimensions are passed to the function. Also initialize each element of this aray to a value passed to the function. e.g. create_array(3,4,5,n) where first three arguments are 3D array dimensions and 4th value is for initialing each value of the 3D array.

INPUT

```
def create_array(x, y, z, value):
    return [[[value for _ in range(z)] for _ in range(y)] for _ in range(x)]

array = create_array(3, 4, 5, 10)
print(array)
```

OUTPUT

```
[[[10, 10, 10, 10, 10], [10, 10, 10, 10, 10], ...]]
```

4. Write a program that defines a function sum_avg() to accept marks of five subjects and calculates total and average. It should return directly both values.

INPUT

```
def sum_avg(marks):
    total = sum(marks)
    avg = total / len(marks)
    return total, avg

marks = [80, 85, 90, 75, 88]
total, avg = sum_avg(marks)
print("Total:", total, "Average:", avg)
```

OUTPUT

```
Total: 418 Average: 83.6
```

5. Pangram is a sentence that uses every letter of the alphabet. Write a program to check whether a given string is pangram or not, through a user-defined function ispangram().

Test the function with “The quick brown fox jumps over the lazy dog” or “Crazy Fredrick bought many very exquisite opal jewels”. Hint: use set() to convert the string into a set of characters present in the string and use <= to check whether alphaset is a subset of the given string.

```
INPUT
def ispangram(string):
    alphabet_set = set("abcdefghijklmnopqrstuvwxyz")
    string_set = set(string.lower().replace(" ", ""))
    return alphabet_set <= string_set

test_string = "The quick brown fox jumps over the lazy dog"
print(ispangram(test_string))
OUTPUT
True
```

6. Write a function to create and return a list containing tuples of the form (x,x2,x3) for all x between 1 and given ending value (both inclusive).

```
INPUT
def create_tuples(end):
    return [(x, x**2, x**3) for x in range(1, end+1)]

print(create_tuples(5))
OUTPUT
[(1, 1, 1), (2, 4, 8), (3, 9, 27), (4, 16, 64), (5, 25, 125)]
```

7. A palindrome is a word or phrase that reads the same in both directions. Write a program that defines a function ispalindrome() which checks whether a given string is a palindrome or not. Ignore spaces and case mismatch while checking for palindrome.

```
INPUT
def ispalindrome(string):
    stripped = string.replace(" ", "").lower()
    return stripped == stripped[::-1]

print(ispalindrome("A man a plan a canal Panama"))
OUTPUT
True
```

8. Write a program that defines a function convert() that receives a string containing a sequence of whitespace separated words and returns a string after removing all duplicate words and sorting them alphanumerically. Hint: use set(), list () , sorted(), join().

```
INPUT
def convert(string):
    words = string.split()
    return " ".join(sorted(set(words)))

sample = "banana apple apple orange banana grape"
print(convert(sample))
OUTPUT
apple banana grape orange
```

9. Write a program that defines a function count_alpha_digits() that accepts a string and calculates the number of alphabets and digits in it. It should return these values as a dictionary.

```
INPUT
def count_alpha_digits(string):
    result = {"alphabets": 0, "digits": 0}
    for char in string:
        if char.isalpha():
            result["alphabets"] += 1
        elif char.isdigit():
            result["digits"] += 1
    return result

sample = "Hello123"
print(count_alpha_digits(sample))
OUTPUT
```

```
{'alphabets': 5, 'digits': 3}

10. Write a program that defines a function called frequency() which computes the
frequency of words present in a string passed to it. The frequencies should be returned
in sorted order of words in the string.
INPUT
def frequency(string):
    words = string.split()
    freq = {word: words.count(word) for word in set(words)}
    return dict(sorted(freq.items()))

sample = "apple banana apple orange banana banana"
print(frequency(sample))
OUTPUT
{'apple': 2, 'banana': 3, 'orange': 1}


11. Write a function create_list() that creates and returns a list which is an
intersection of two lists passed to it.
INPUT
def create_list(list1, list2):
    return list(set(list1) & set(list2))

list1 = [1, 2, 3, 4, 5]
list2 = [3, 4, 5, 6, 7]
print(create_list(list1, list2))
OUTPUT
[3, 4, 5]
```