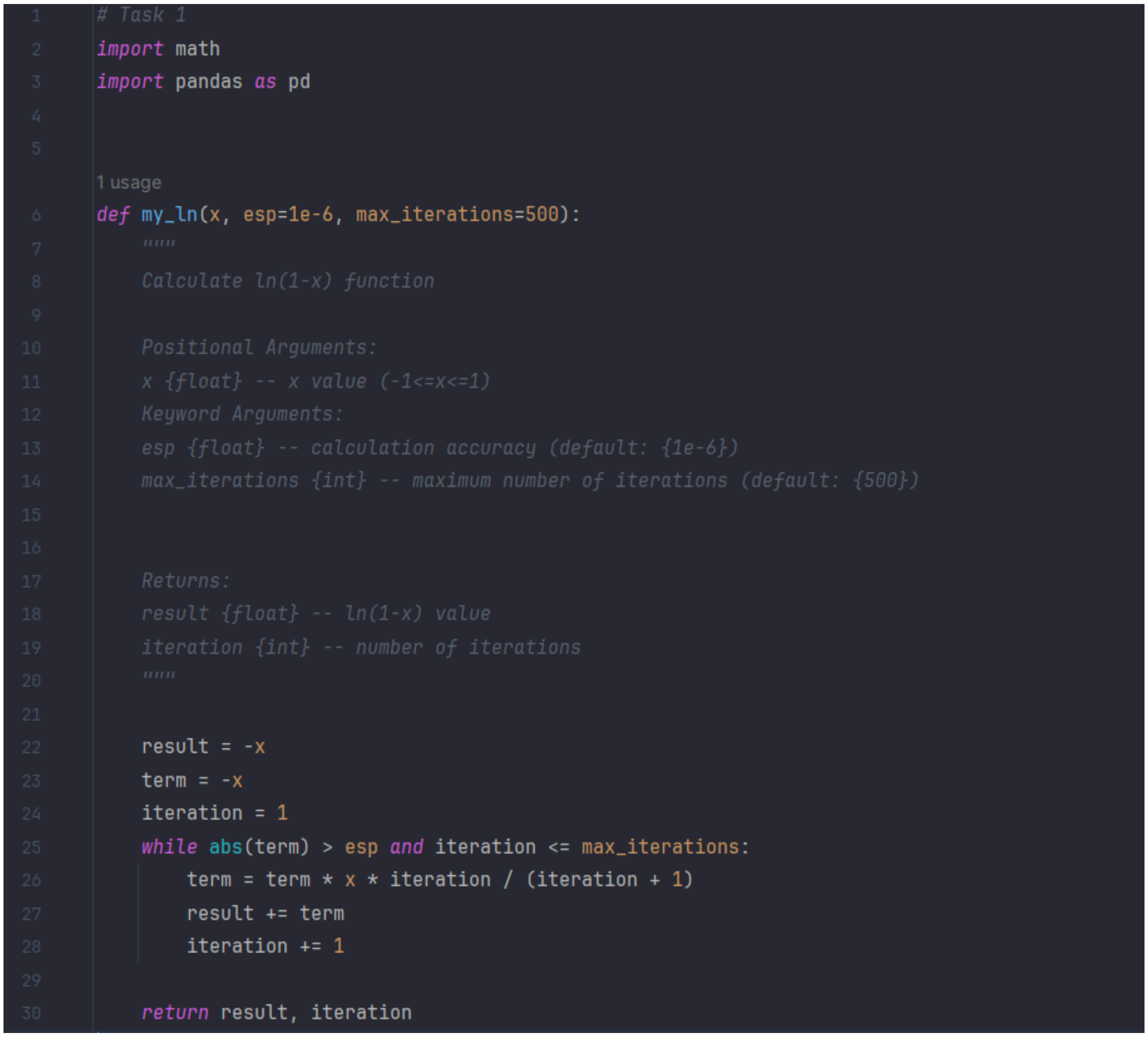
Лабораторная работа №3

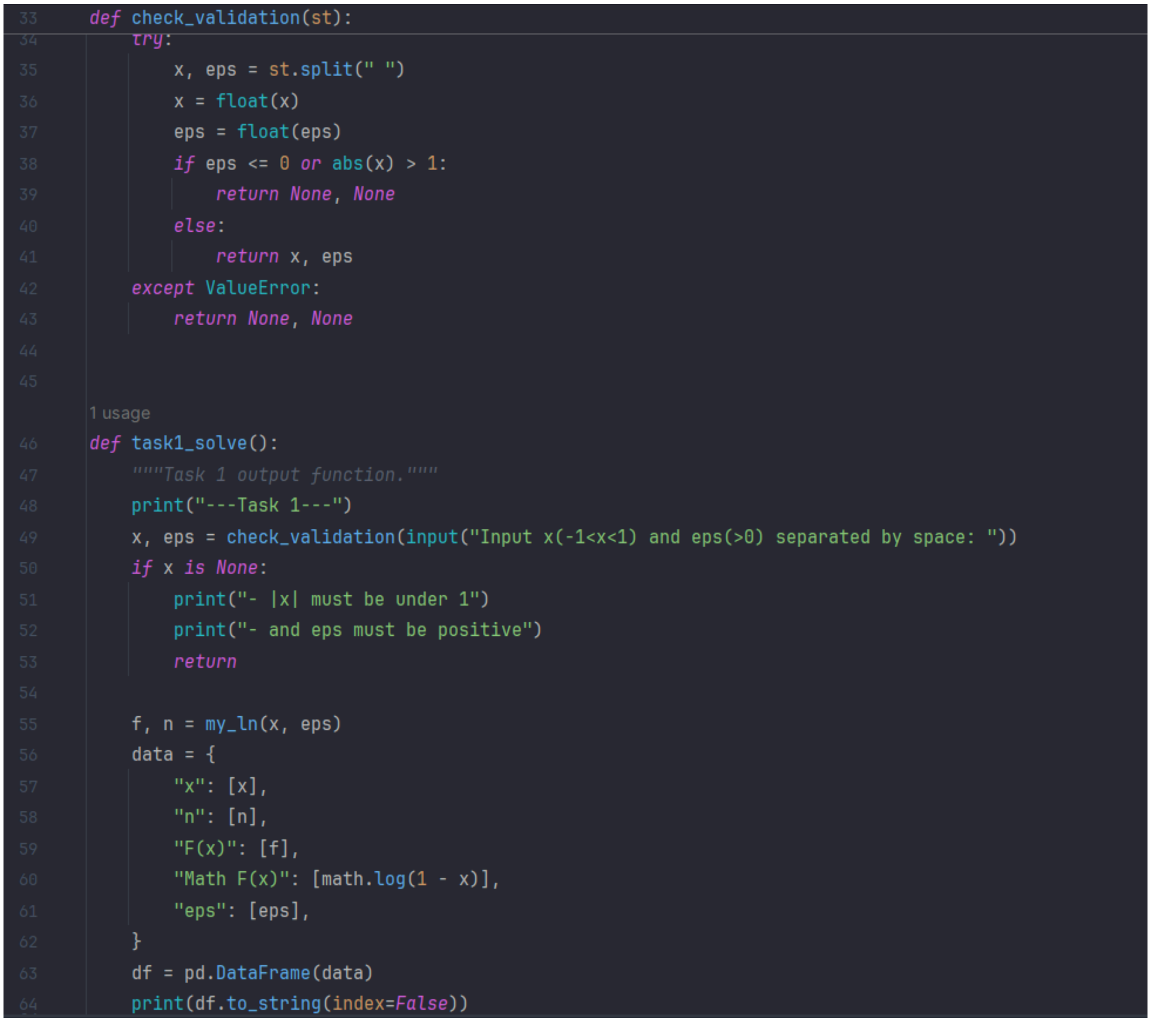
Тема: Стандартные типы данных, коллекции, функции, модули.

Цель: освоить базовый синтаксис языка Python, приобрести навыки работы со стандартными типами данных, коллекциями, функциями, модулями и закрепить их на примере разработки интерактивных приложений.
Выполнил: Барановский Г.В. Гр.253502

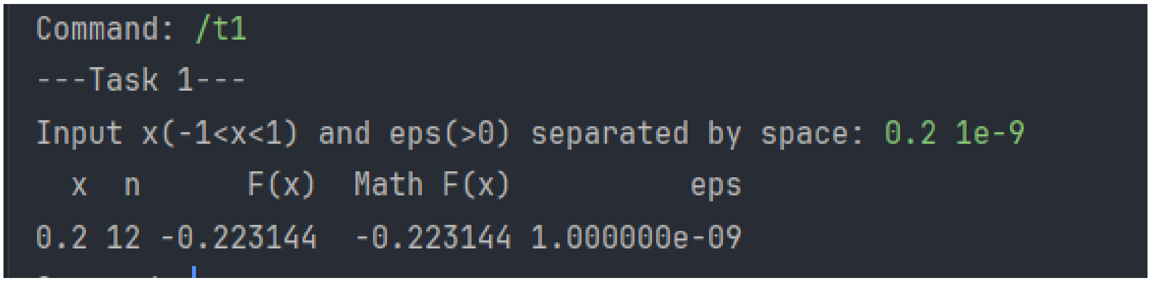
1. Задание 1:



```python
# Task 1
import math
import pandas as pd


def my_ln(x, esp=1e-6, max_iterations=500):
    """
    Calculate ln(1-x) function

    Positional Arguments:
    x {float} -- x value (-1<=x<=1)
    Keyword Arguments:
    esp {float} -- calculation accuracy (default: {1e-6})
    max_iterations {int} -- maximum number of iterations (default: {500})


    Returns:
    result {float} -- ln(1-x) value
    iteration {int} -- number of iterations
    """

    result = -x
    term = -x
    iteration = 1
    while abs(term) > esp and iteration <= max_iterations:
        term = term * x * iteration / (iteration + 1)
        result += term
        iteration += 1

    return result, iteration
```

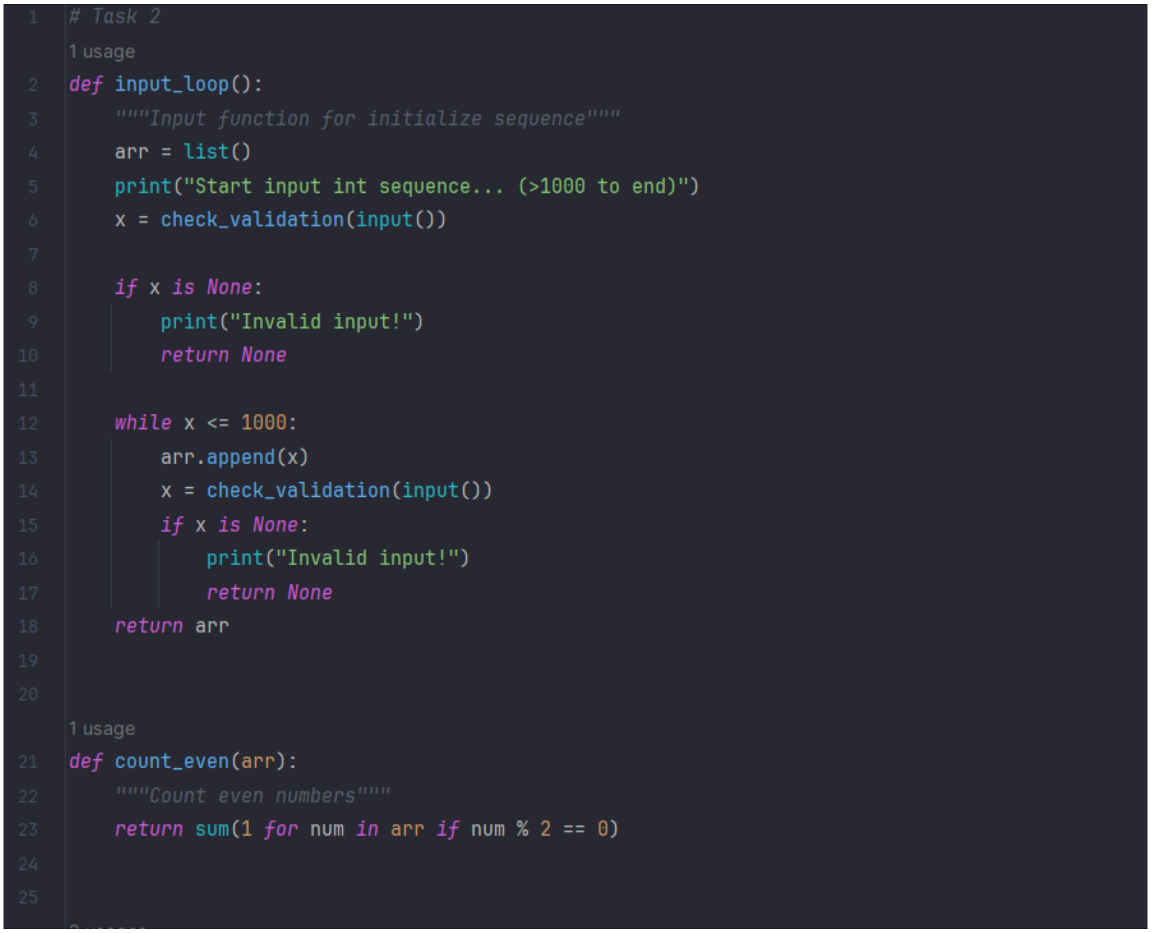


```
def check_validation(st):
    try:
        x, eps = st.split(" ")
        x = float(x)
        eps = float(eps)
        if eps <= 0 or abs(x) > 1:
            return None, None
        else:
            return x, eps
    except ValueError:
        return None, None


1 usage
def task1_solve():
    """Task 1 output function."""
    print("---Task 1---")
    x, eps = check_validation(input("Input x(-1<x<1) and eps(>0) separated by space: "))
    if x is None:
        print("- |x| must be under 1")
        print("- and eps must be positive")
        return

    f, n = my_ln(x, eps)
    data = {
        "x": [x],
        "n": [n],
        "F(x)": [f],
        "Math F(x)": [math.log(1 - x)],
        "eps": [eps],
    }
    df = pd.DataFrame(data)
    print(df.to_string(index=False))
```

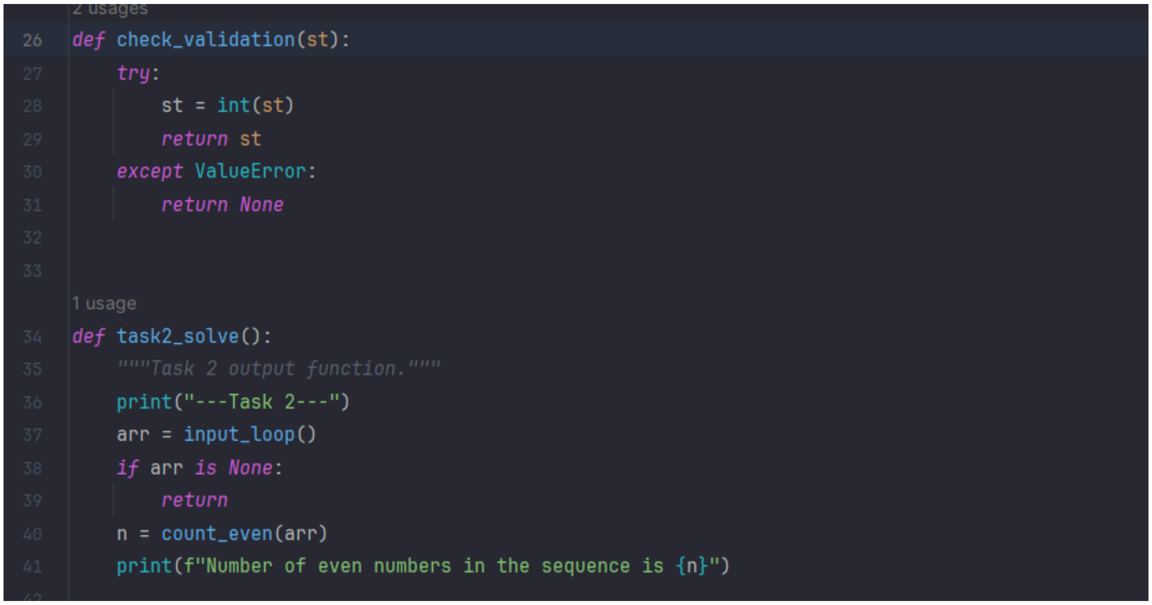


```
Command: /t1
---Task 1---
Input x(-1<x<1) and eps(>0) separated by space: 0.2 1e-9
  x  n      F(x)  Math F(x)          eps
0.2 12 -0.223144  -0.223144 1.000000e-09
```

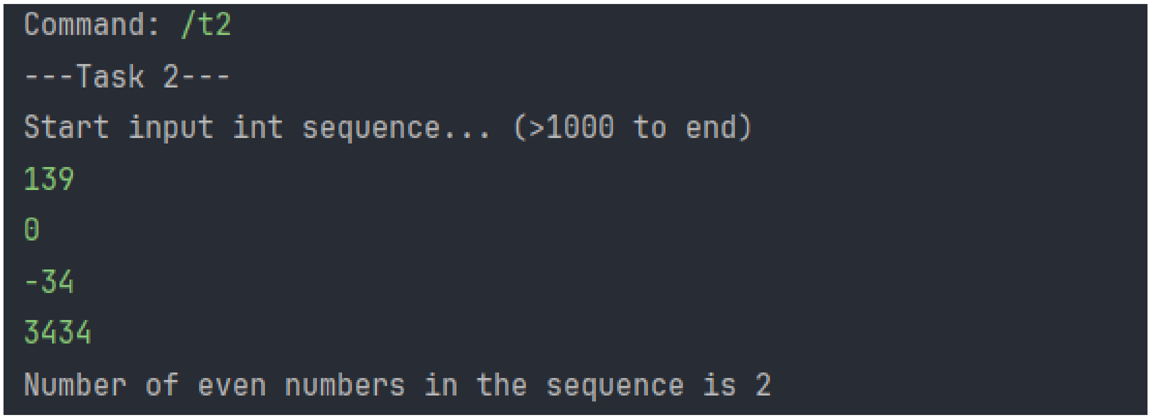
2. Задание 2:



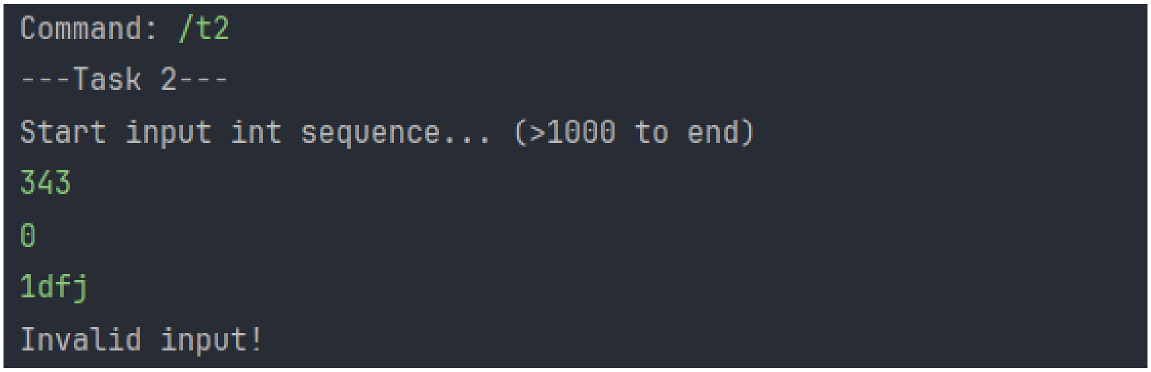
```python
# Task 2
def input_loop():
    """Input function for initialize sequence"""
    arr = list()
    print("Start input int sequence... (>1000 to end)")
    x = check_validation(input())

    if x is None:
        print("Invalid input!")
        return None

    while x <= 1000:
        arr.append(x)
        x = check_validation(input())
        if x is None:
            print("Invalid input!")
            return None
    return arr


def count_even(arr):
    """Count even numbers"""
    return sum(1 for num in arr if num % 2 == 0)


```



```python
def check_validation(st):
    try:
        st = int(st)
        return st
    except ValueError:
        return None


def task2_solve():
    """Task 2 output function."""
    print("---Task 2---")
    arr = input_loop()
    if arr is None:
        return
    n = count_even(arr)
    print(f"Number of even numbers in the sequence is {n}")
```

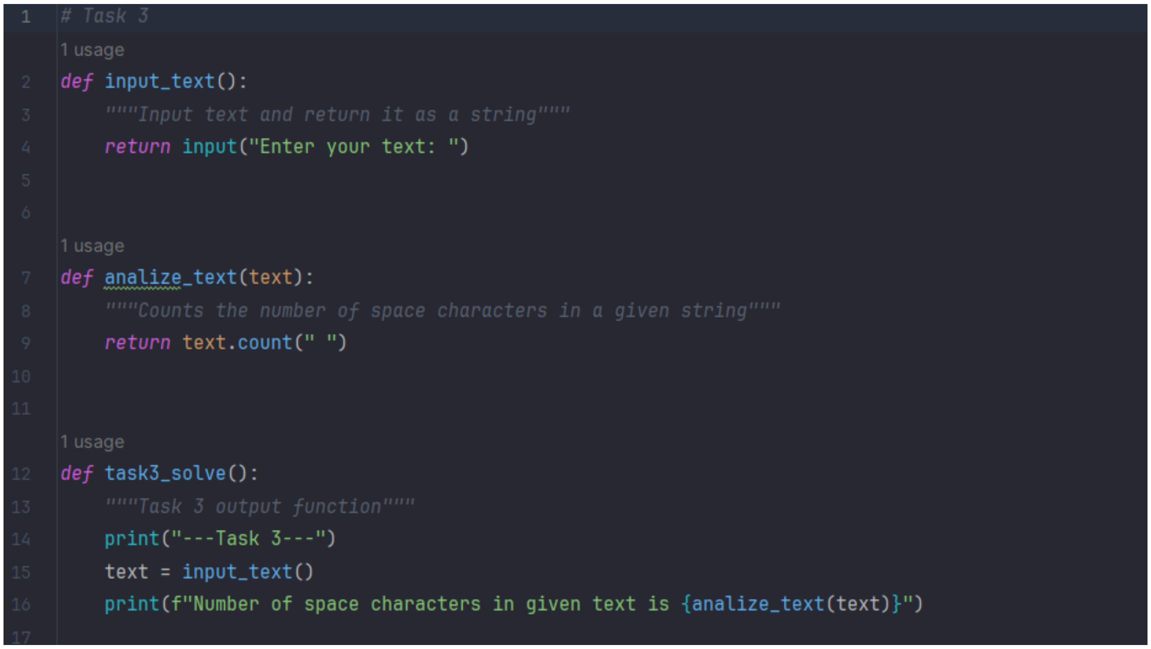


```
Command: /t2
---Task 2---
Start input int sequence... (>1000 to end)
139
0
-34
3434
Number of even numbers in the sequence is 2
```

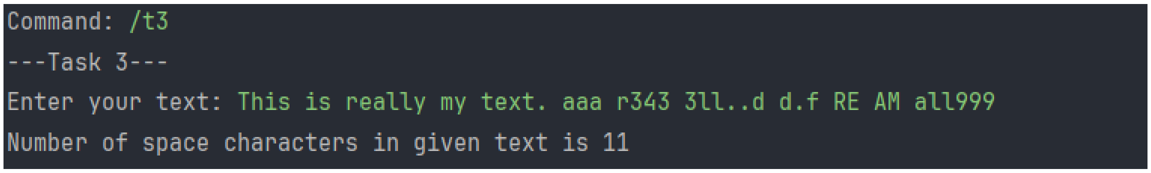


```
Command: /t2
---Task 2---
Start input int sequence... (>1000 to end)
343
0
1dfj
Invalid input!
```

3. Задание 3

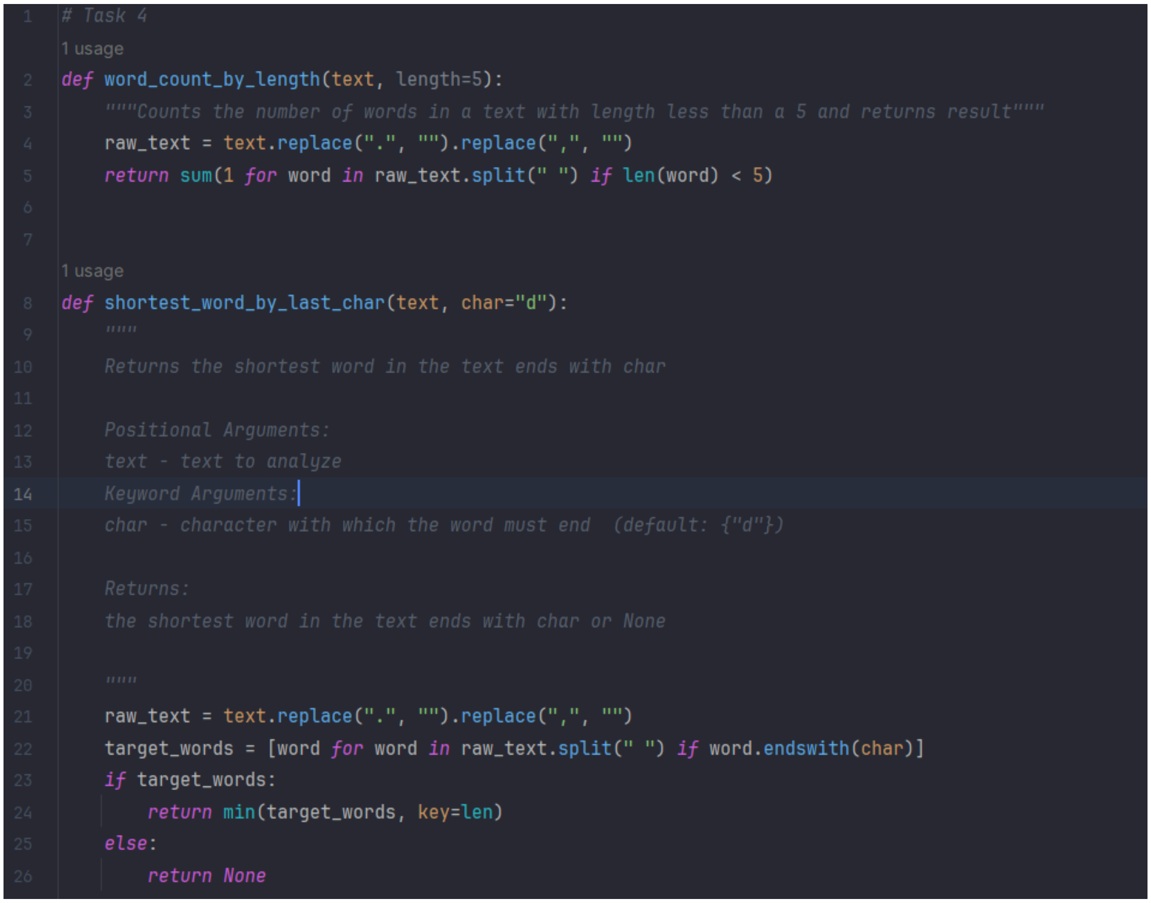


```python
# Task 3
1 usage
def input_text():
    """Input text and return it as a string"""
    return input("Enter your text: ")


1 usage
def analize_text(text):
    """Counts the number of space characters in a given string"""
    return text.count(" ")


1 usage
def task3_solve():
    """Task 3 output function"""
    print("---Task 3---")
    text = input_text()
    print(f"Number of space characters in given text is {analize_text(text)}")
```

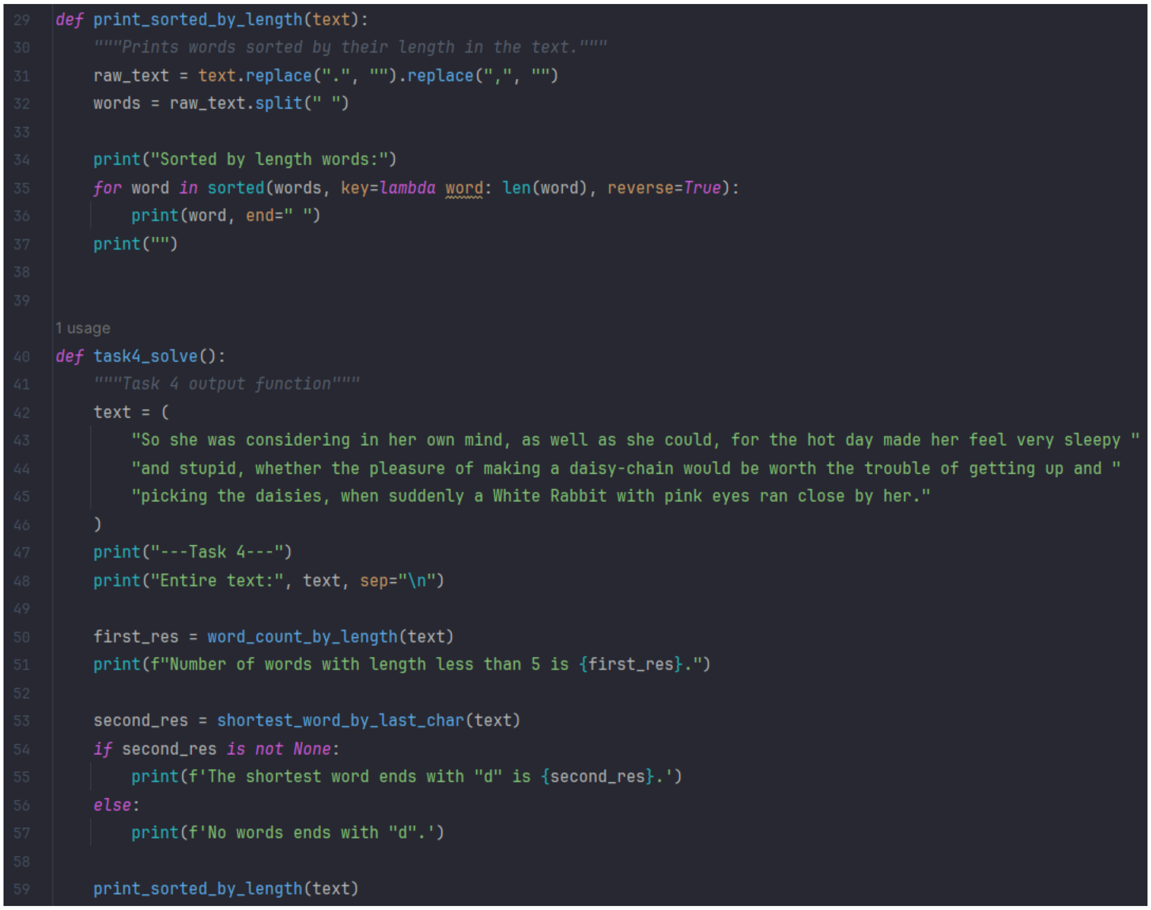


```
Command: /t3
---Task 3---
Enter your text: This is really my text. aaa r343 3ll..d d.f RE AM all999
Number of space characters in given text is 11
```

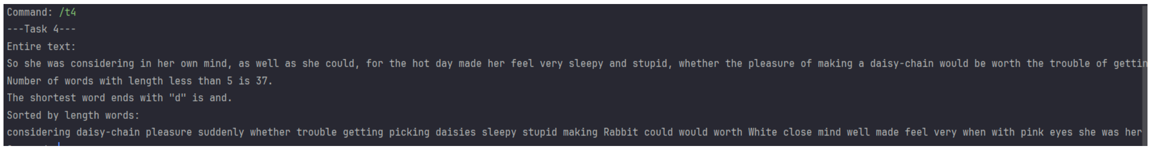
4. Задание 4:



```python
# Task 4
def word_count_by_length(text, length=5):
    """Counts the number of words in a text with length less than a 5 and returns result"""
    raw_text = text.replace(".", "").replace(",", "")
    return sum(1 for word in raw_text.split(" ") if len(word) < 5)


def shortest_word_by_last_char(text, char="d"):
    """
    Returns the shortest word in the text ends with char

    Positional Arguments:
    text - text to analyze
    Keyword Arguments:
    char - character with which the word must end  (default: {"d"})

    Returns:
    the shortest word in the text ends with char or None

    """
    raw_text = text.replace(".", "").replace(",", "")
    target_words = [word for word in raw_text.split(" ") if word.endswith(char)]
    if target_words:
        return min(target_words, key=len)
    else:
        return None
```

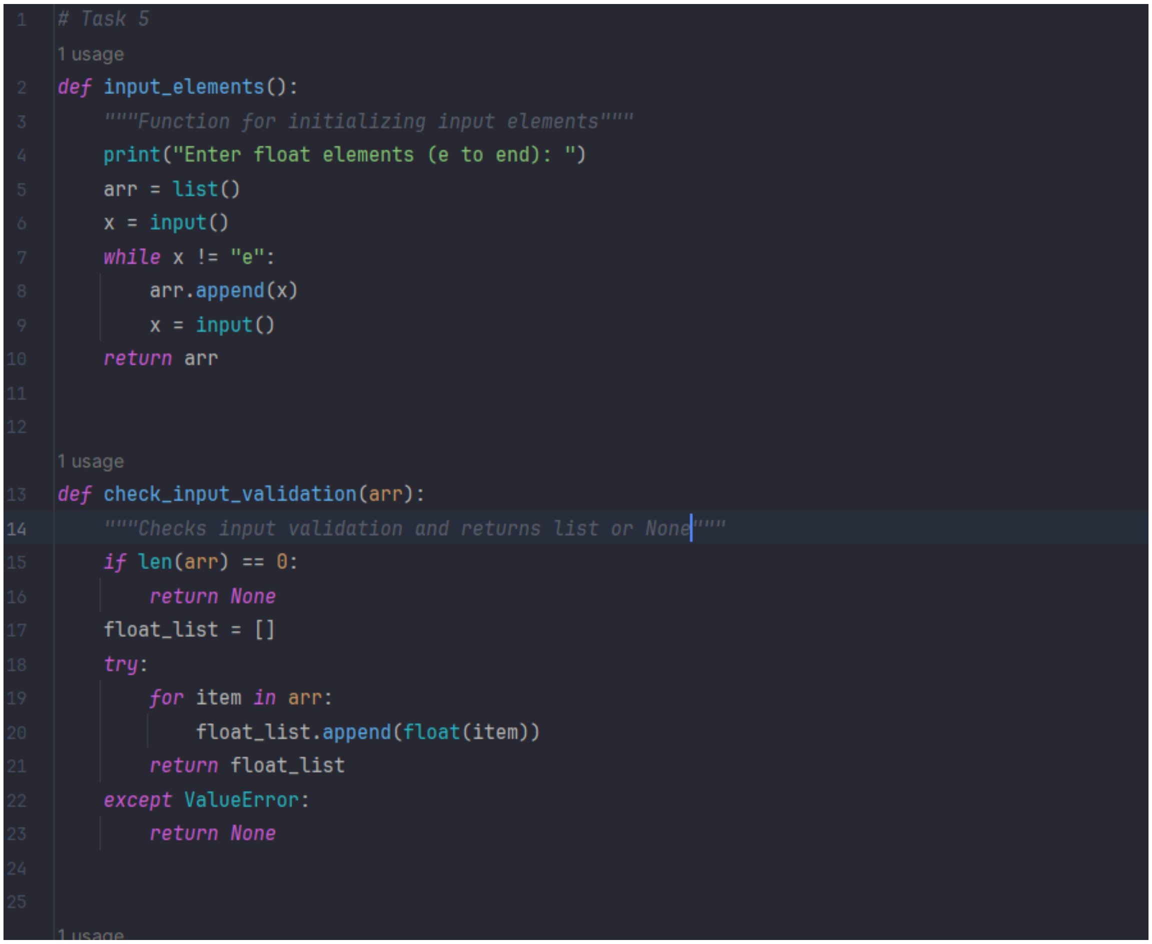


```python
def print_sorted_by_length(text):
    """Prints words sorted by their length in the text."""
    raw_text = text.replace(".", "").replace(",", "")
    words = raw_text.split(" ")

    print("Sorted by length words:")
    for word in sorted(words, key=lambda word: len(word), reverse=True):
        print(word, end=" ")
    print("")


1 usage
def task4_solve():
    """Task 4 output function"""
    text = (
        "So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy "
        "and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and "
        "picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her."
    )
    print("---Task 4---")
    print("Entire text:", text, sep="\n")

    first_res = word_count_by_length(text)
    print(f"Number of words with length less than 5 is {first_res}.")

    second_res = shortest_word_by_last_char(text)
    if second_res is not None:
        print(f'The shortest word ends with "d" is {second_res}.')
    else:
        print(f'No words ends with "d".')

    print_sorted_by_length(text)
```

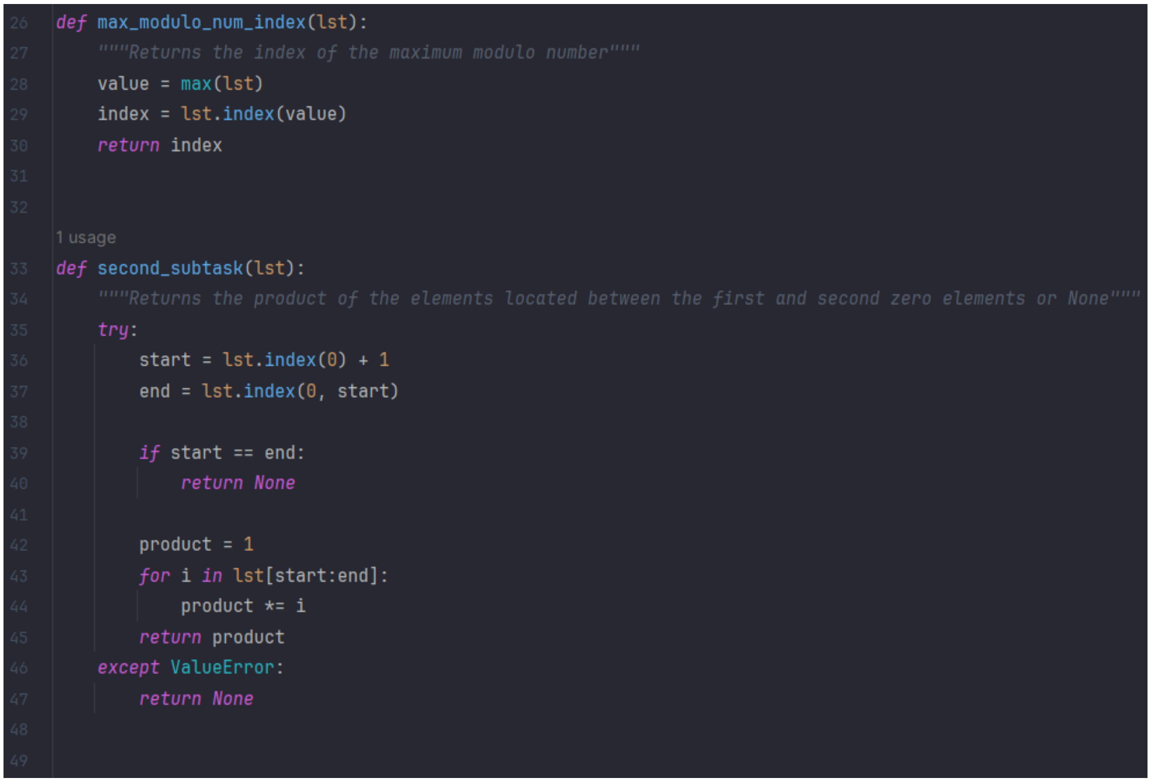


```
Command: /t4
---Task 4---
Entire text:
So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of gettin
Number of words with length less than 5 is 37.
The shortest word ends with "d" is and.
Sorted by length words:
considering daisy-chain pleasure suddenly whether trouble getting picking daisies sleepy stupid making Rabbit could would worth White close mind well made feel very when with pink eyes she was her
```

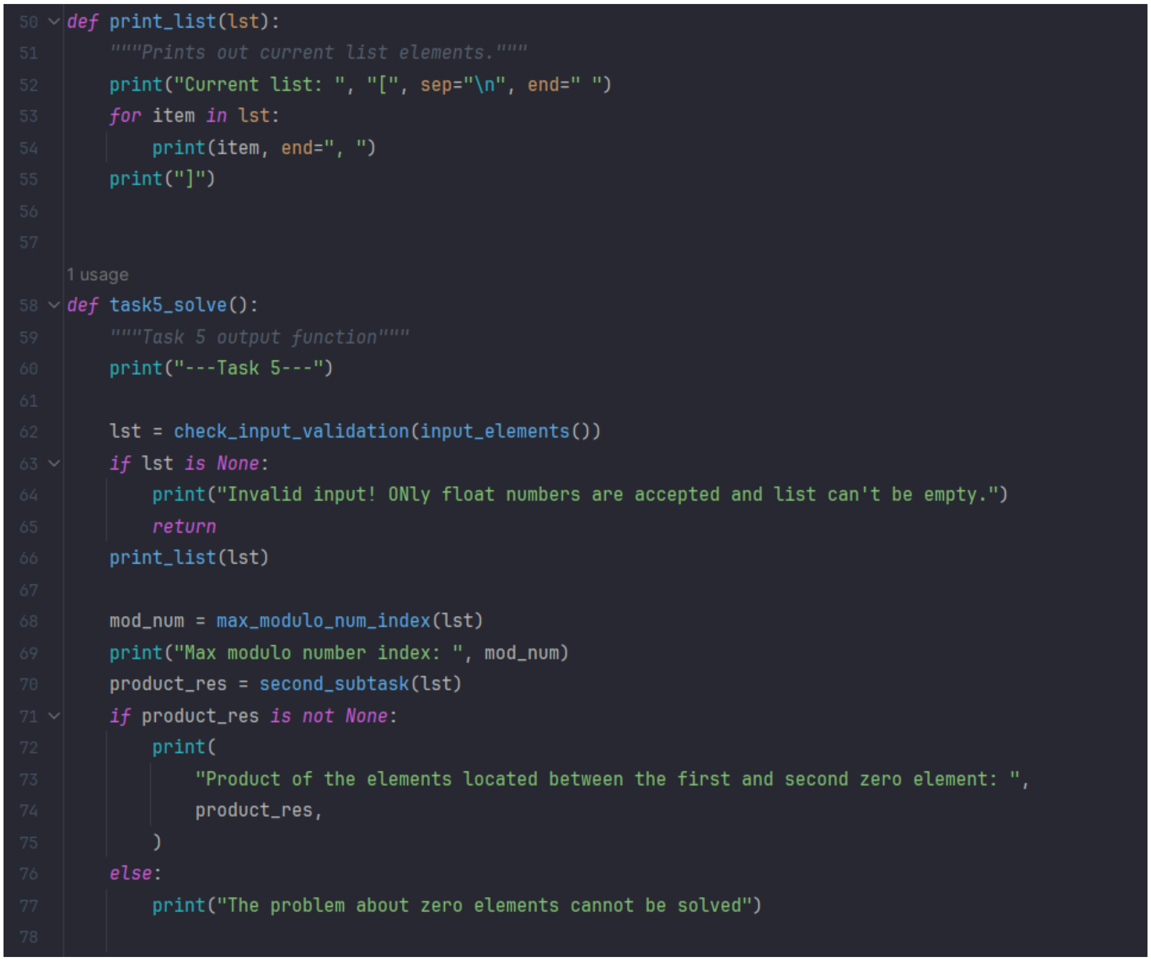
5. Задание 5:



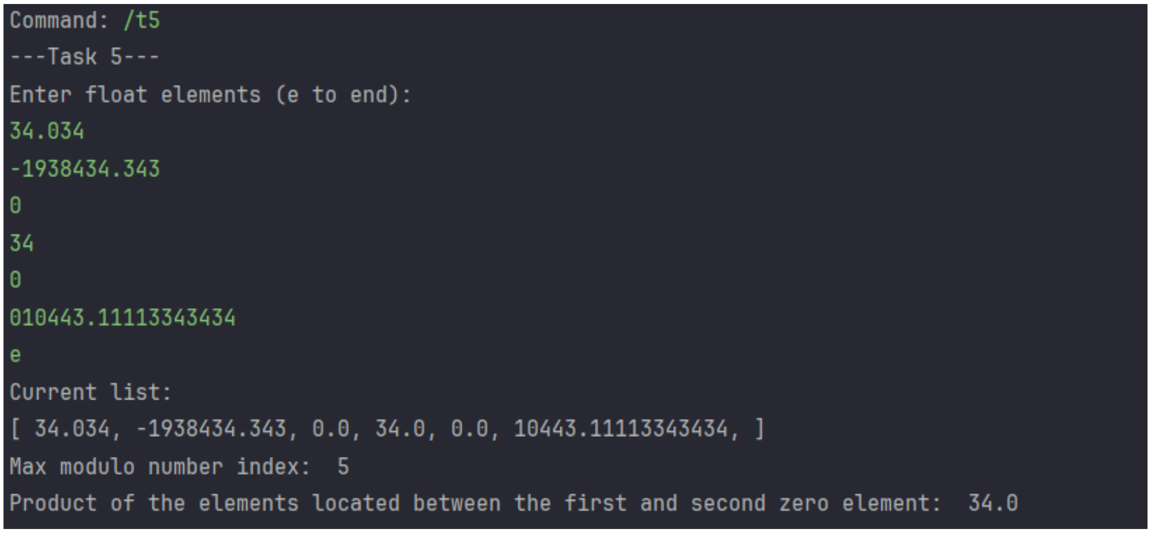
```python
# Task 5
1 usage
def input_elements():
    """Function for initializing input elements"""
    print("Enter float elements (e to end): ")
    arr = list()
    x = input()
    while x != "e":
        arr.append(x)
        x = input()
    return arr


1 usage
def check_input_validation(arr):
    """Checks input validation and returns list or None"""
    if len(arr) == 0:
        return None
    float_list = []
    try:
        for item in arr:
            float_list.append(float(item))
        return float_list
    except ValueError:
        return None


1 usage
```



```python
def max_modulo_num_index(lst):
    """Returns the index of the maximum modulo number"""
    value = max(lst)
    index = lst.index(value)
    return index


def second_subtask(lst):
    """Returns the product of the elements located between the first and second zero elements or None"""
    try:
        start = lst.index(0) + 1
        end = lst.index(0, start)

        if start == end:
            return None

        product = 1
        for i in lst[start:end]:
            product *= i
        return product
    except ValueError:
        return None


```

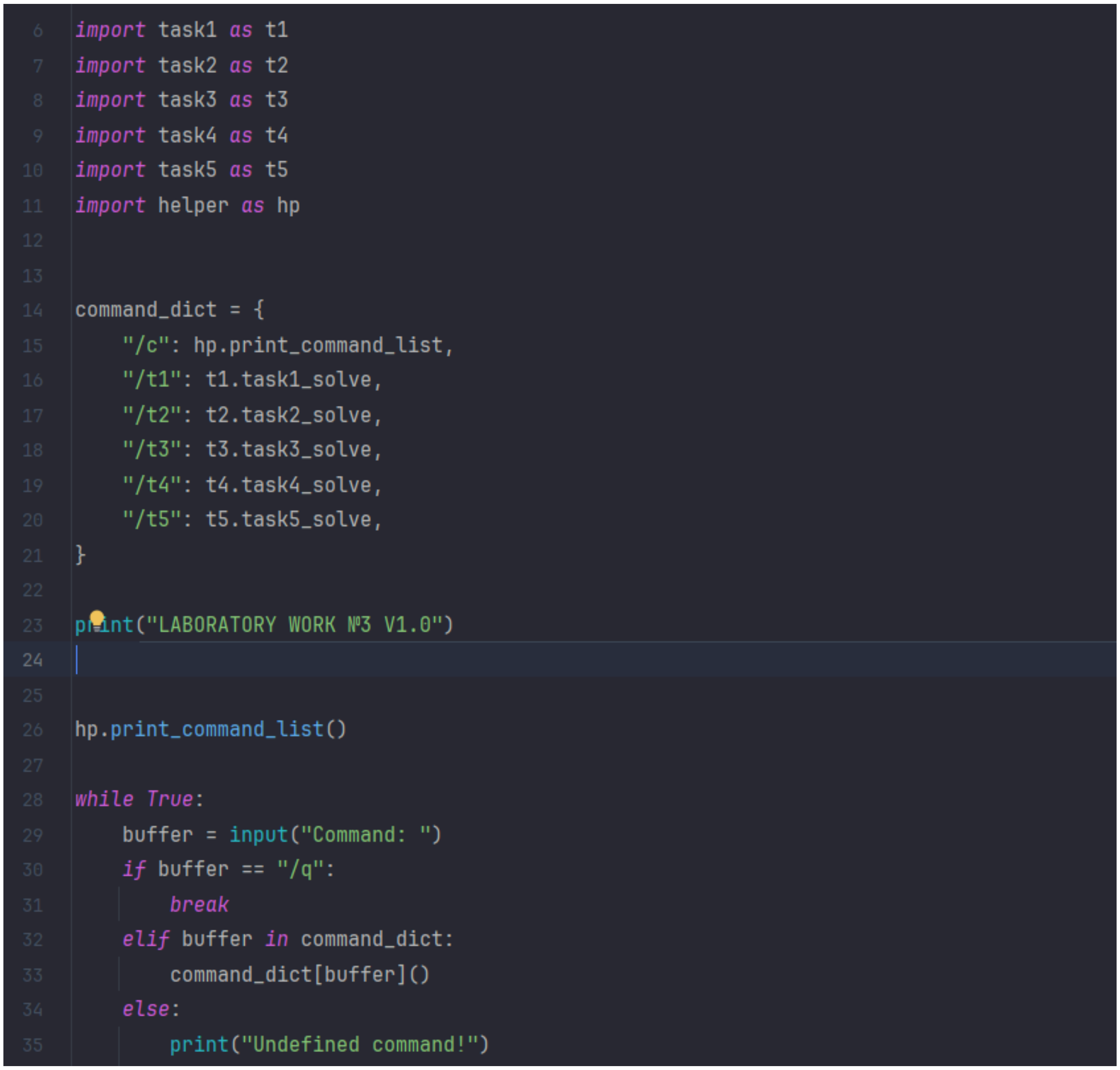


```
def print_list(lst):
    """Prints out current list elements."""
    print("Current list: ", "[", sep="\n", end=" ")
    for item in lst:
        print(item, end=", ")
    print("]")


def task5_solve():
    """Task 5 output function"""
    print("---Task 5---")

    lst = check_input_validation(input_elements())
    if lst is None:
        print("Invalid input! ONly float numbers are accepted and list can't be empty.")
        return
    print_list(lst)

    mod_num = max_modulo_num_index(lst)
    print("Max modulo number index: ", mod_num)
    product_res = second_subtask(lst)
    if product_res is not None:
        print(
            "Product of the elements located between the first and second zero element: ",
            product_res,
        )
    else:
        print("The problem about zero elements cannot be solved")

```

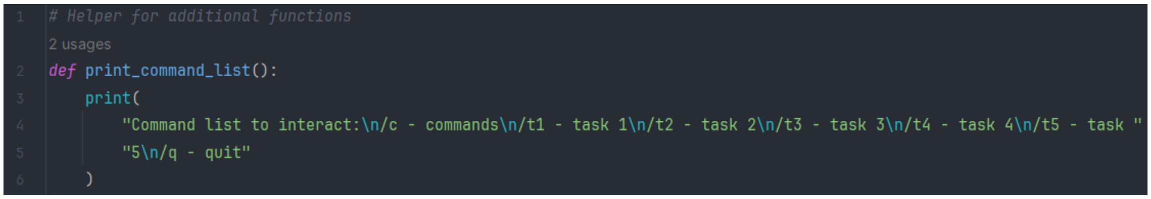


```
Command: /t5
---Task 5---
Enter float elements (e to end):
34.034
-1938434.343
0
34
0
010443.11113343434
e
Current list:
[ 34.034, -1938434.343, 0.0, 34.0, 0.0, 10443.11113343434, ]
Max modulo number index:  5
Product of the elements located between the first and second zero element:  34.0
```

6. Main program:



```python
import task1 as t1
import task2 as t2
import task3 as t3
import task4 as t4
import task5 as t5
import helper as hp


command_dict = {
    "/c": hp.print_command_list,
    "/t1": t1.task1_solve,
    "/t2": t2.task2_solve,
    "/t3": t3.task3_solve,
    "/t4": t4.task4_solve,
    "/t5": t5.task5_solve,
}

print("LABORATORY WORK №3 V1.0")


hp.print_command_list()

while True:
    buffer = input("Command: ")
    if buffer == "/q":
        break
    elif buffer in command_dict:
        command_dict[buffer]()
    else:
        print("Undefined command!")
```



```python
# Helper for additional functions
2 usages
def print_command_list():
    print(
        "Command list to interact:\n/c - commands\n/t1 - task 1\n/t2 - task 2\n/t3 - task 3\n/t4 - task 4\n/t5 - task "
        "5\n/q - quit"
    )
```