Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра Информатики
Дисциплина «Избранные главы информатики»

ОТЧЕТ

к лабораторной работе №3
на тему:

«СТАНДАРТНЫЕ ТИПЫ ДАННЫХ, КОЛЛЕКЦИИ, ФУНКЦИИ, МОДУЛИ»

БГУИР 6-05-0612-02

Выполнил студент группы 353504
ГОРДАШУК Владислав Сергеевич

(дата, подпись студента)

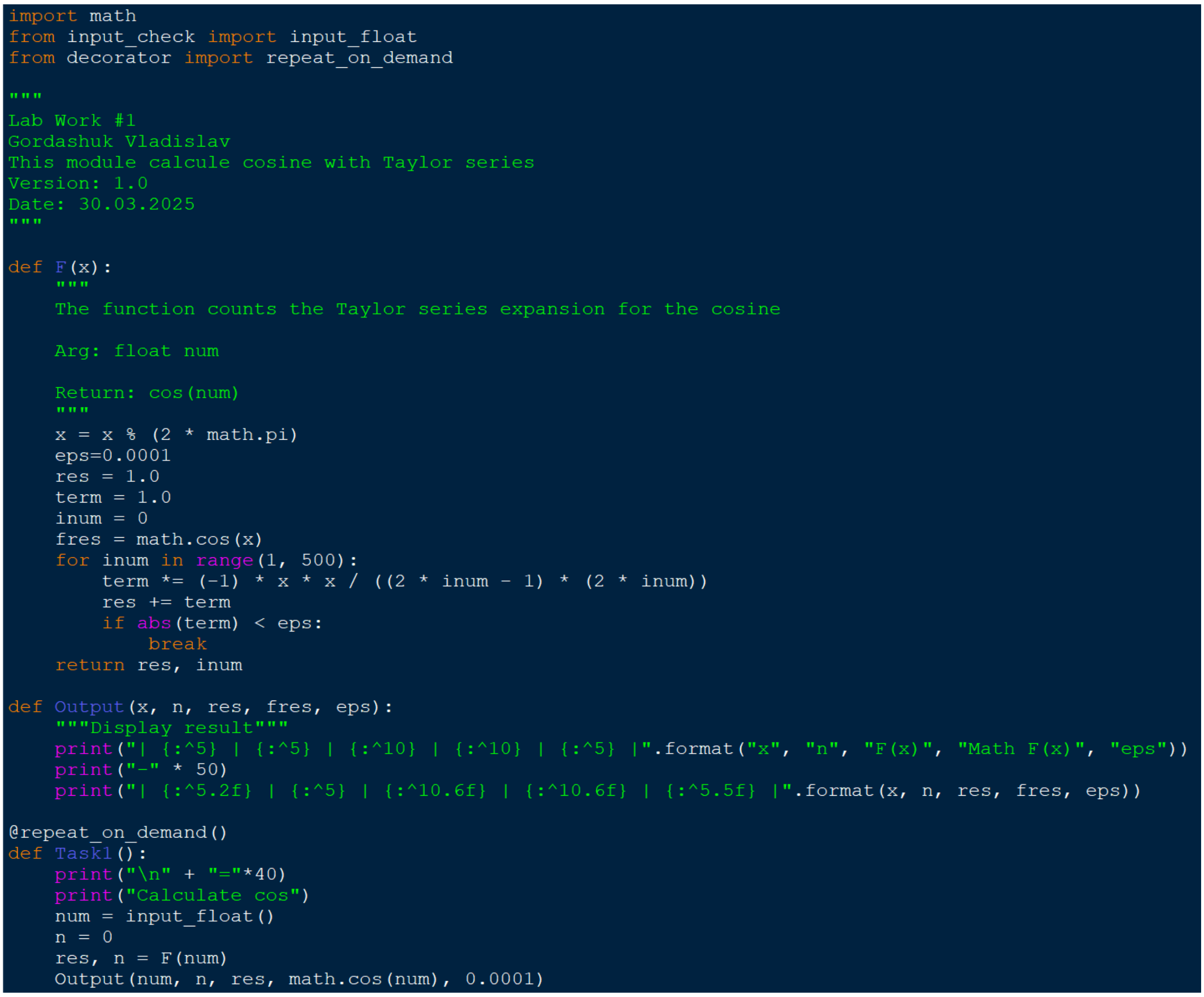
Проверил
Жвакина Анна Васильевна

(дата, подпись преподавателя)

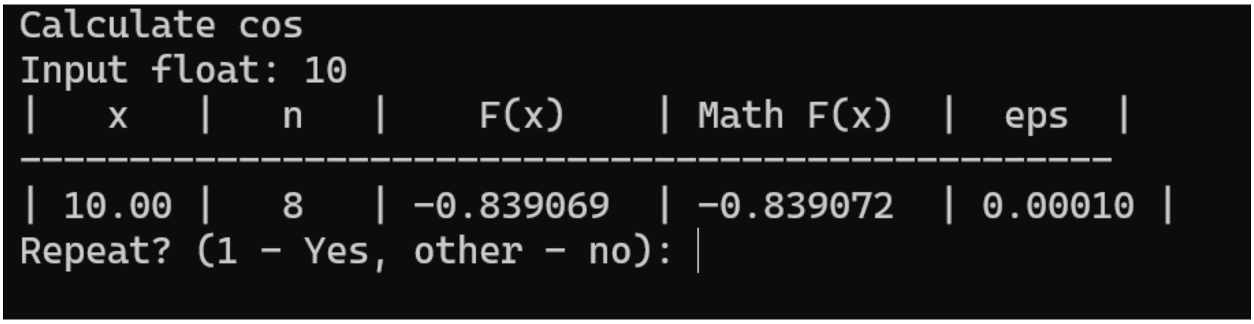
Минск 2024

Задание 1. В соответствии с заданием своего варианта составить программу для вычисления значения функции с помощью разложения функции в степенной ряд. Задать точность вычислений eps.

$$\cos x = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots$$



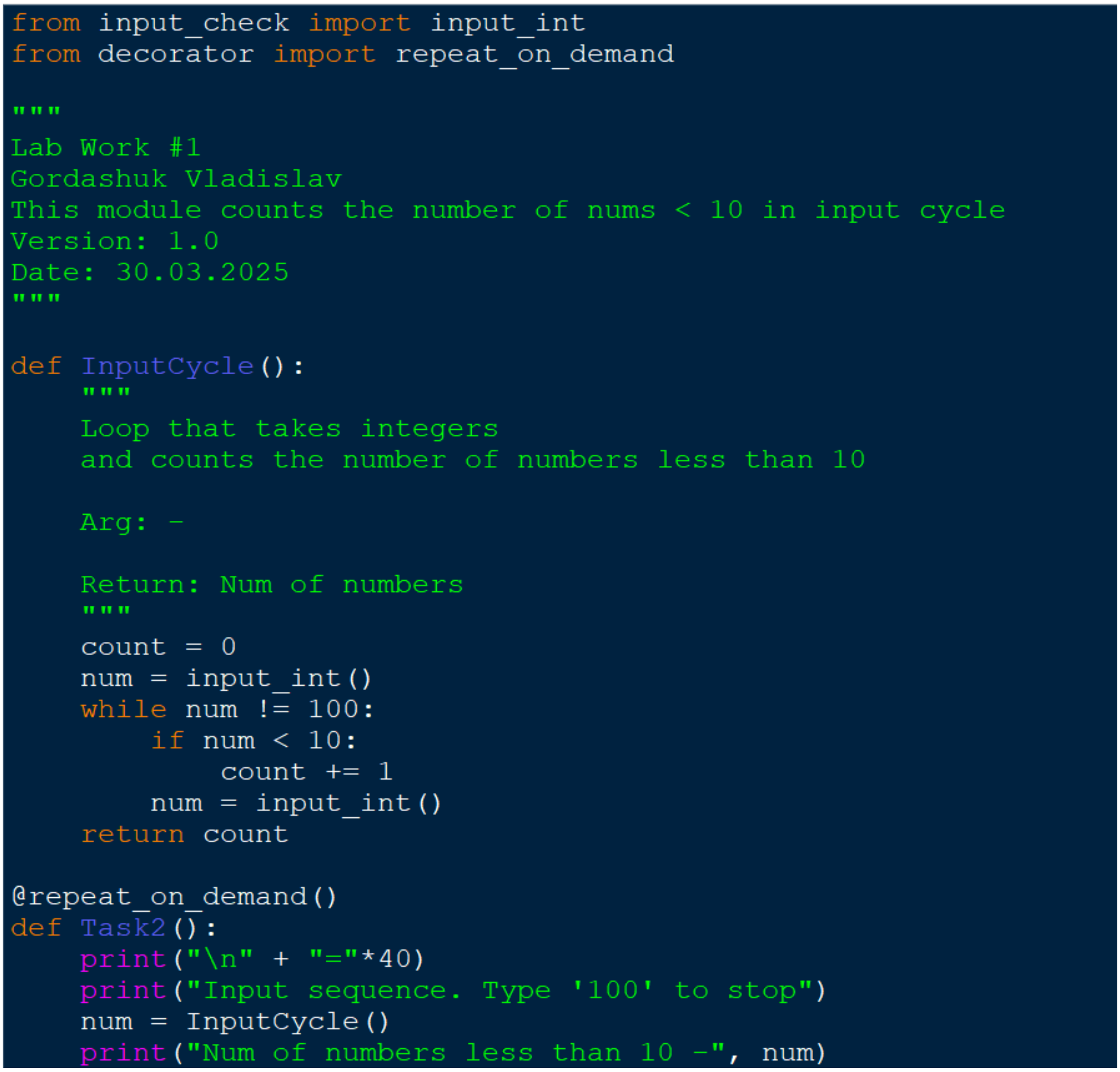
```
import math
from input_check import input_float
from decorator import repeat_on_demand

"""
Lab Work #1
Gordashuk Vladislav
This module calcule cosine with Taylor series
Version: 1.0
Date: 30.03.2025
"""

def F(x):
    """
    The function counts the Taylor series expansion for the cosine

    Arg: float num

    Return: cos(num)
    """
    x = x % (2 * math.pi)
    eps=0.0001
    res = 1.0
    term = 1.0
    inum = 0
    fres = math.cos(x)
    for inum in range(1, 500):
        term *= (-1) * x * x / ((2 * inum - 1) * (2 * inum))
        res += term
        if abs(term) < eps:
            break
    return res, inum

def Output(x, n, res, fres, eps):
    """Display result"""
    print("| {:^5} | {:^5} | {:^10} | {:^10} | {:^5} |".format("x", "n", "F(x)", "Math F(x)", "eps"))
    print("-" * 50)
    print("| {:^5.2f} | {:^5} | {:^10.6f} | {:^10.6f} | {:^5.5f} |".format(x, n, res, fres, eps))

@repeat_on_demand()
def Task1():
    print("\n" + "="*40)
    print("Calculate cos")
    num = input_float()
    n = 0
    res, n = F(num)
    Output(num, n, res, math.cos(num), 0.0001)
```



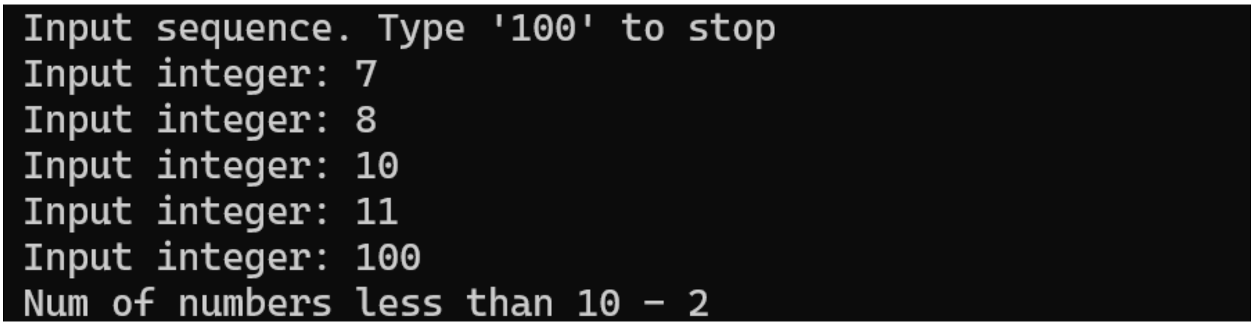
```
Calculate cos
Input float: 10
|   x   |   n   |    F(x)    | Math F(x)  |  eps  |
--------------------------------------------------
| 10.00 |   8   | -0.839069  | -0.839072  | 0.00010 |
Repeat? (1 - Yes, other - no):
```

Задание 2. В соответствии с заданием своего варианта составить программу для нахождения суммы последовательности чисел.

Организовать цикл, который принимает целые числа с клавиатуры и подсчитывает количество чисел, меньших числа 10. Окончание цикла – ввод числа 100.



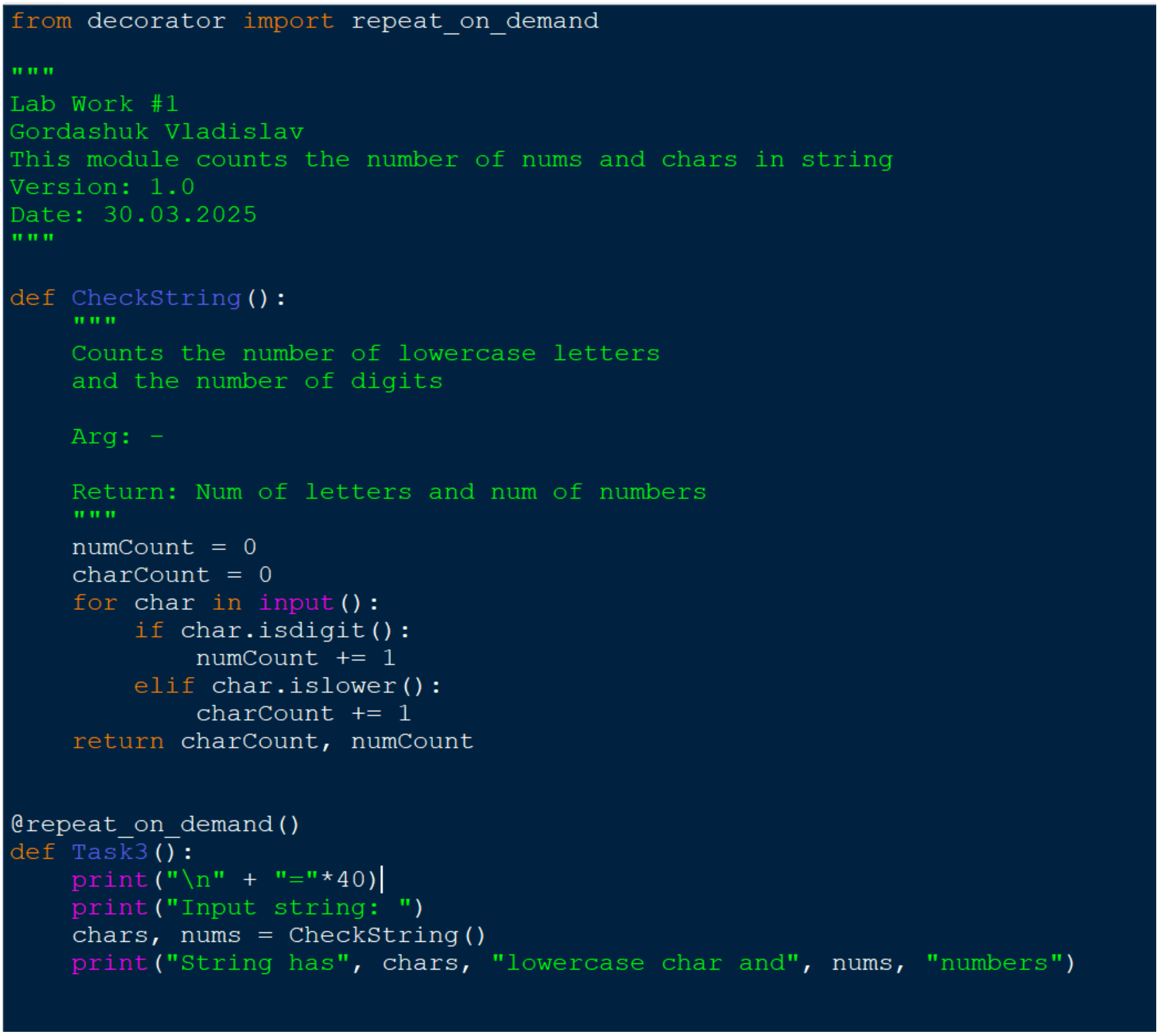
```python
from input_check import input_int
from decorator import repeat_on_demand

"""
Lab Work #1
Gordashuk Vladislav
This module counts the number of nums < 10 in input cycle
Version: 1.0
Date: 30.03.2025
"""

def InputCycle():
    """
    Loop that takes integers
    and counts the number of numbers less than 10

    Arg: -

    Return: Num of numbers
    """
    count = 0
    num = input_int()
    while num != 100:
        if num < 10:
            count += 1
        num = input_int()
    return count

@repeat_on_demand()
def Task2():
    print("\n" + "="*40)
    print("Input sequence. Type '100' to stop")
    num = InputCycle()
    print("Num of numbers less than 10 -", num)
```



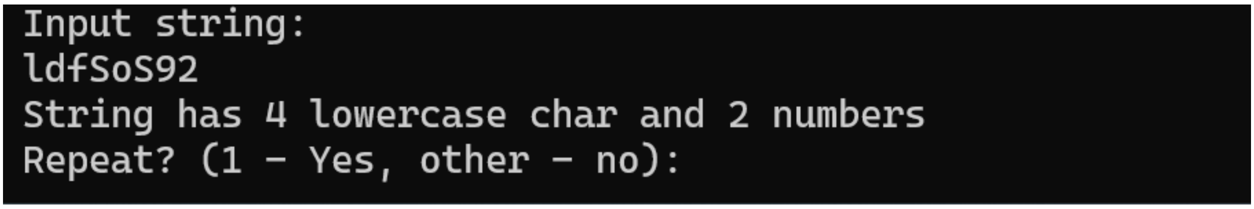
```
Input sequence. Type '100' to stop
Input integer: 7
Input integer: 8
Input integer: 10
Input integer: 11
Input integer: 100
Num of numbers less than 10 - 2
```

Задание 3. Не использовать регулярные выражения. В соответствии с заданием своего варианта составить программу для анализа текста, вводимого с клавиатуры.

В строке, вводимой с клавиатуры, подсчитать количество букв нижнего регистра и цифр.



```
from decorator import repeat_on_demand

"""
Lab Work #1
Gordashuk Vladislav
This module counts the number of nums and chars in string
Version: 1.0
Date: 30.03.2025
"""

def CheckString():
    """
    Counts the number of lowercase letters
    and the number of digits

    Arg: -

    Return: Num of letters and num of numbers
    """
    numCount = 0
    charCount = 0
    for char in input():
        if char.isdigit():
            numCount += 1
        elif char.islower():
            charCount += 1
    return charCount, numCount


@repeat_on_demand()
def Task3():
    print("\n" + "="*40)
    print("Input string: ")
    chars, nums = CheckString()
    print("String has", chars, "lowercase char and", nums, "numbers")
```



```
Input string:
ldfSoS92
String has 4 lowercase char and 2 numbers
Repeat? (1 - Yes, other - no):
```

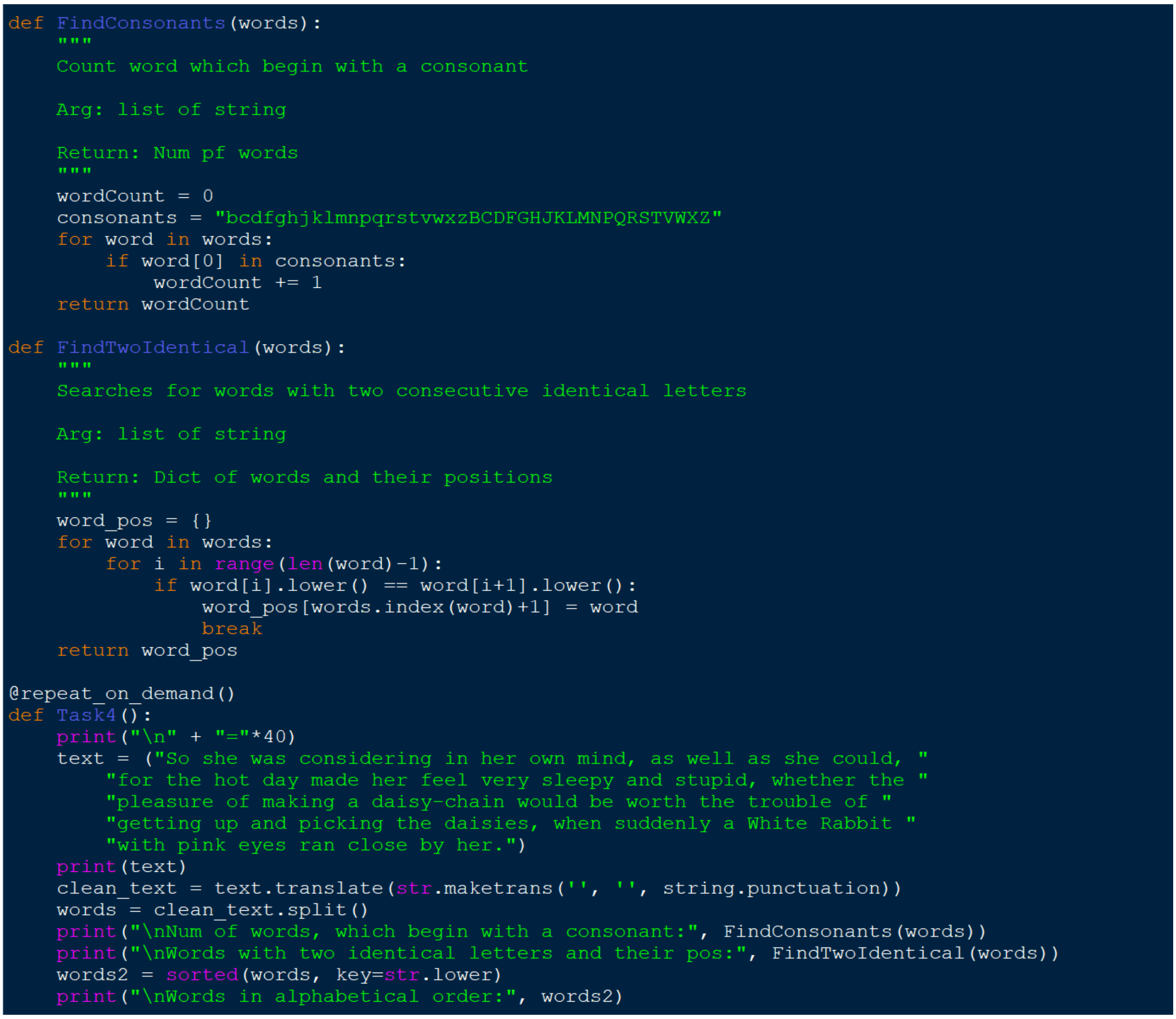
Задание 4. Не использовать регулярные выражения. Дана строка текста, в которой слова разделены пробелами и запятыми. В соответствии с заданием своего варианта составьте программу для анализа строки, инициализированной в коде программы:

«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.»

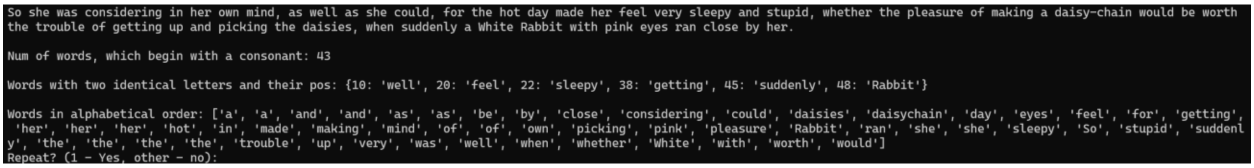
а) определить число слов, которые начинаются с согласной;

б) найти слова, содержащие две одинаковые буквы подряд и их порядковые номера;

в) вывести слова в алфавитном порядке



```
def FindConsonants(words):
    """
    Count word which begin with a consonant

    Arg: list of string

    Return: Num pf words
    """
    wordCount = 0
    consonants = "bcdfghjklmnpqrstvwxzBCDFGHJKLMNPQRSTVWXZ"
    for word in words:
        if word[0] in consonants:
            wordCount += 1
    return wordCount

def FindTwoIdentical(words):
    """
    Searches for words with two consecutive identical letters

    Arg: list of string

    Return: Dict of words and their positions
    """
    word_pos = {}
    for word in words:
        for i in range(len(word)-1):
            if word[i].lower() == word[i+1].lower():
                word_pos[words.index(word)+1] = word
                break
    return word_pos

@repeat_on_demand()
def Task4():
    print("\n" + "="*40)
    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(text)
    clean_text = text.translate(str.maketrans('', '', string.punctuation))
    words = clean_text.split()
    print("\nNum of words, which begin with a consonant:", FindConsonants(words))
    print("\nWords with two identical letters and their pos:", FindTwoIdentical(words))
    words2 = sorted(words, key=str.lower)
    print("\nWords in alphabetical order:", words2)
```



```
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.

Num of words, which begin with a consonant: 43

Words with two identical letters and their pos: {10: 'well', 20: 'feel', 22: 'sleepy', 38: 'getting', 45: 'suddenly', 48: 'Rabbit'}

Words in alphabetical order: ['a', 'a', 'and', 'and', 'as', 'as', 'be', 'by', 'close', 'considering', 'could', 'daisies', 'daisychain', 'day', 'eyes', 'feel', 'for', 'getting',
 'her', 'her', 'her', 'hot', 'in', 'made', 'making', 'mind', 'of', 'of', 'own', 'picking', 'pink', 'pleasure', 'Rabbit', 'ran', 'she', 'she', 'sleepy', 'So', 'stupid', 'suddenl
y', 'the', 'the', 'the', 'the', 'trouble', 'up', 'very', 'was', 'well', 'when', 'whether', 'White', 'with', 'worth', 'would']
Repeat? (1 - Yes, other - no):
```

Задание 5. В соответствии с заданием своего варианта составить программу для обработки вещественных списков. Программа должна содержать следующие базовые функции:

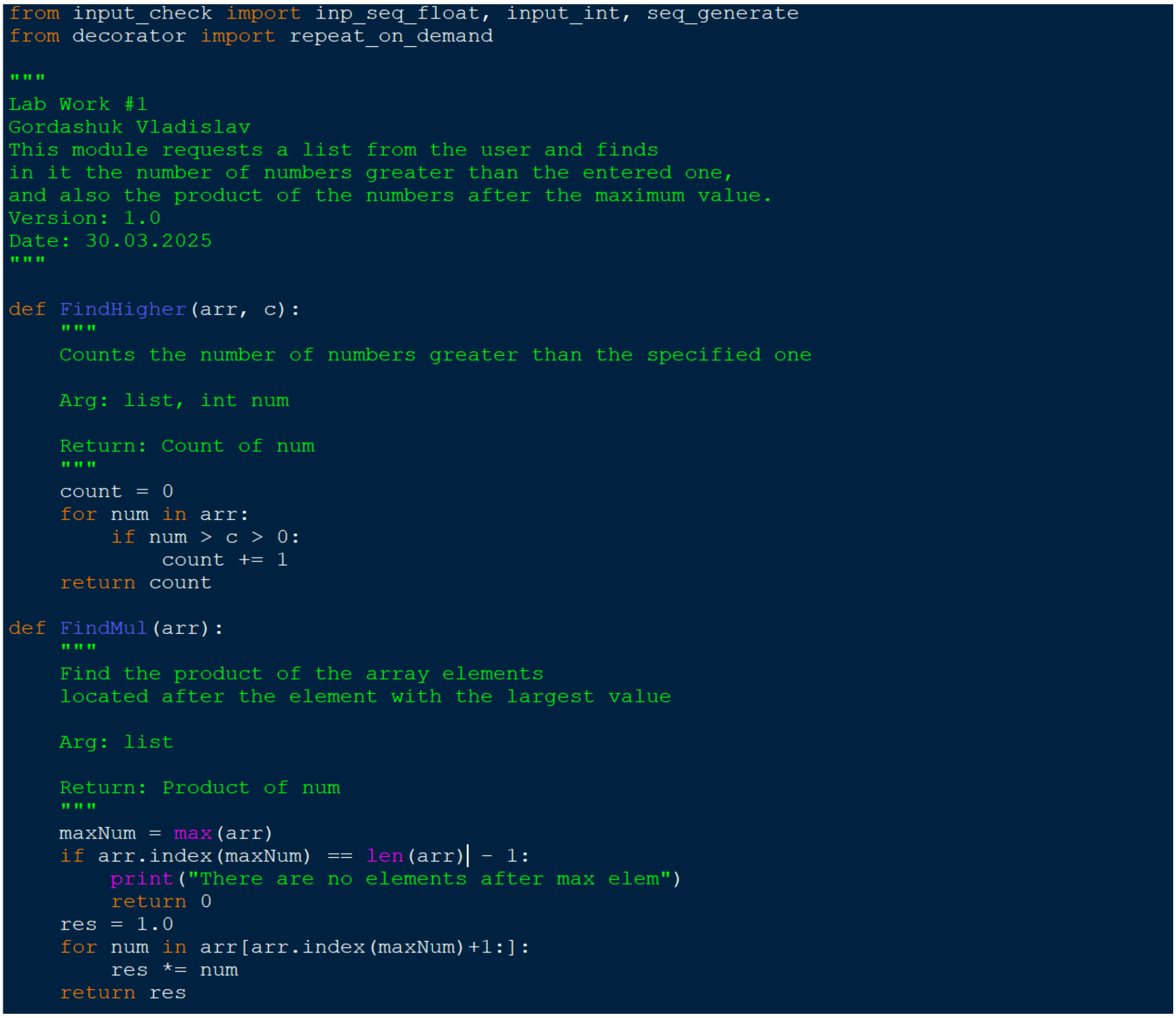
1) ввод элементов списка пользователем;

2) проверка корректности вводимых данных;

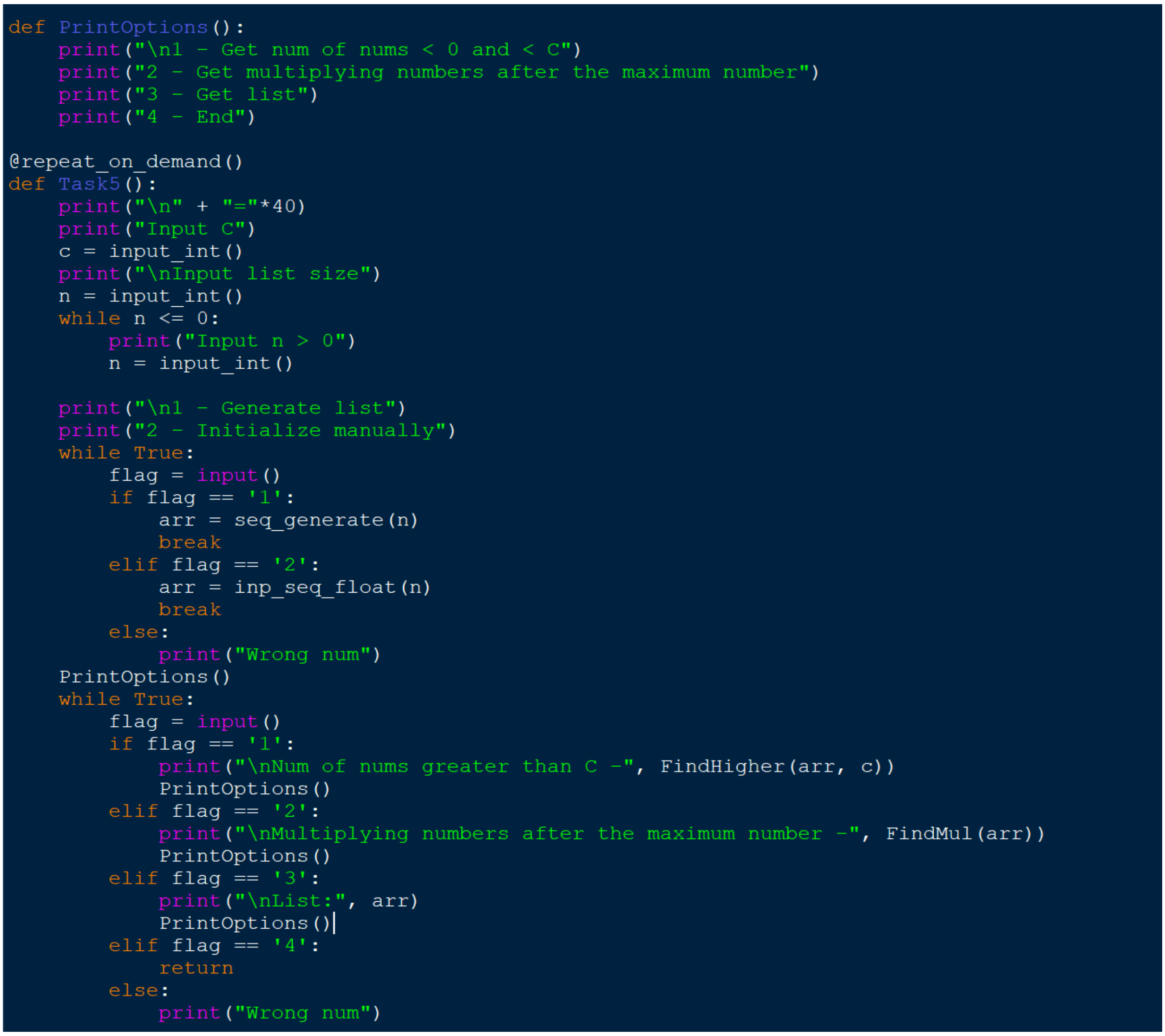
3) реализация основного задания с выводом результатов;

4) вывод списка на экран.

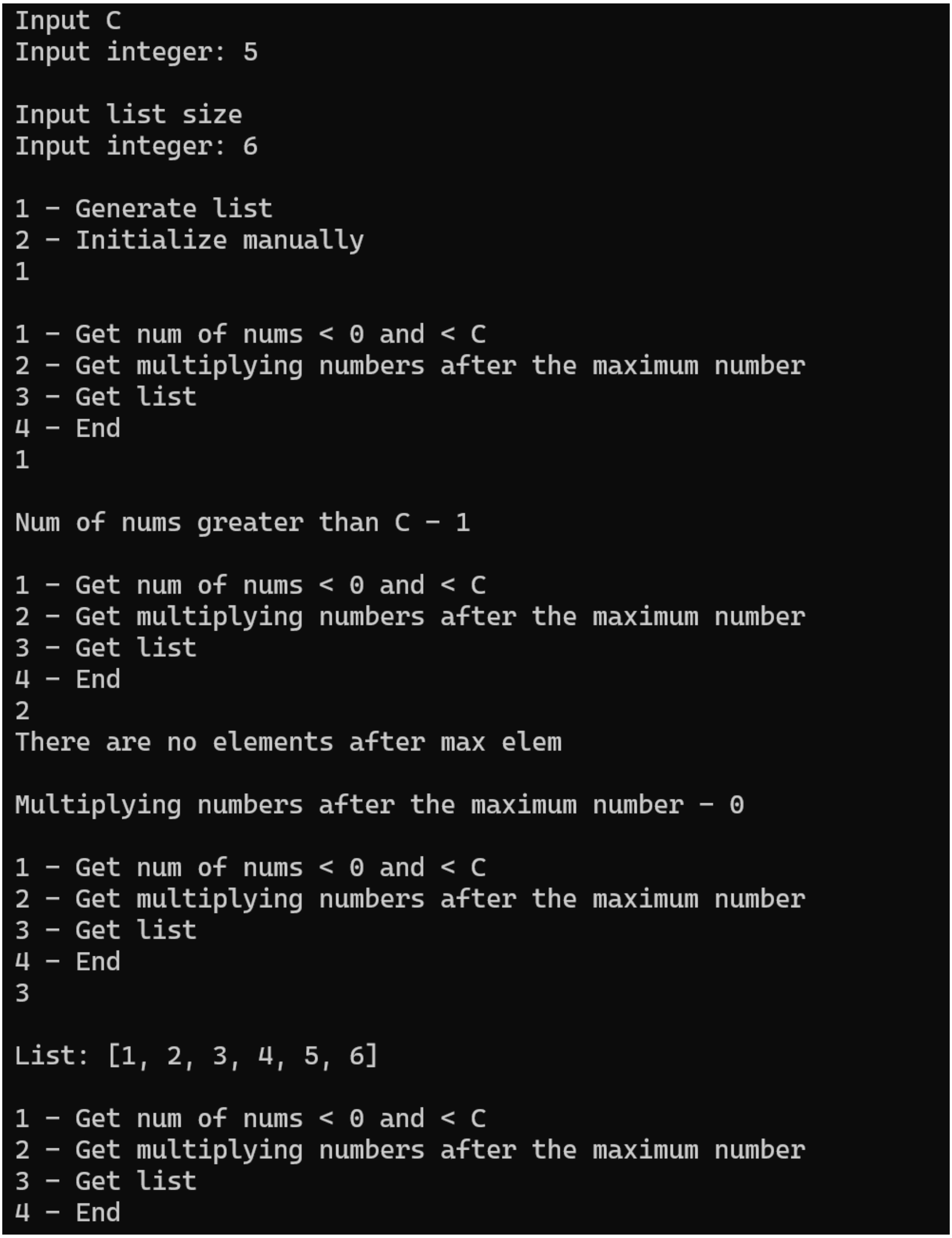
Найти количество положительных элементов списка, больших числа C (параметр C вводится с клавиатуры пользователем) и произведение элементов списка, расположенных после максимального по модулю элемента.



```
from input_check import inp_seq_float, input_int, seq_generate
from decorator import repeat_on_demand

"""
Lab Work #1
Gordashuk Vladislav
This module requests a list from the user and finds
in it the number of numbers greater than the entered one,
and also the product of the numbers after the maximum value.
Version: 1.0
Date: 30.03.2025
"""

def FindHigher(arr, c):
    """
    Counts the number of numbers greater than the specified one

    Arg: list, int num

    Return: Count of num
    """
    count = 0
    for num in arr:
        if num > c > 0:
            count += 1
    return count

def FindMul(arr):
    """
    Find the product of the array elements
    located after the element with the largest value

    Arg: list

    Return: Product of num
    """
    maxNum = max(arr)
    if arr.index(maxNum) == len(arr) - 1:
        print("There are no elements after max elem")
        return 0
    res = 1.0
    for num in arr[arr.index(maxNum)+1:]:
        res *= num
    return res
```

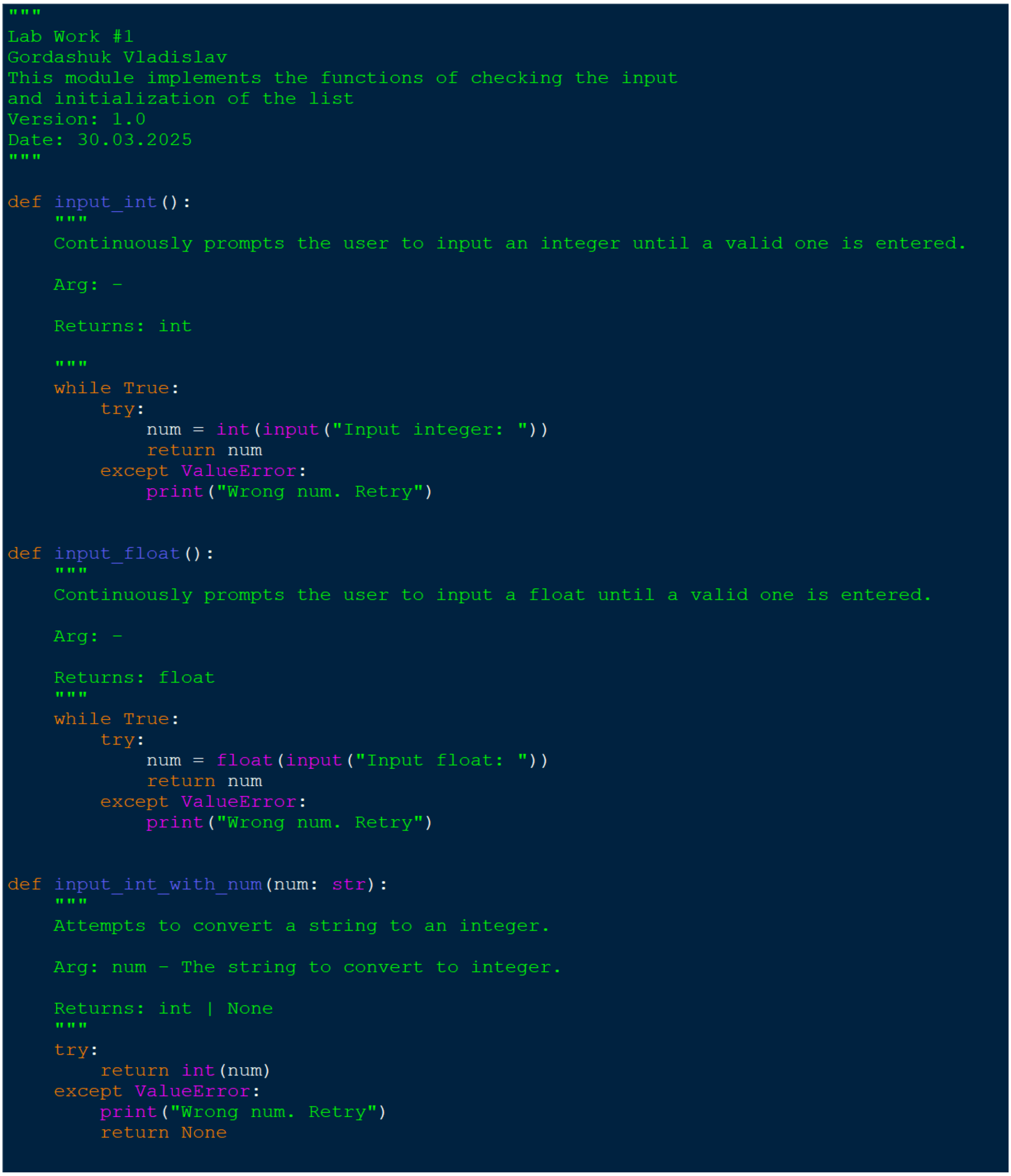


```
def PrintOptions():
    print("\n1 - Get num of nums < 0 and < C")
    print("2 - Get multiplying numbers after the maximum number")
    print("3 - Get list")
    print("4 - End")

@repeat_on_demand()
def Task5():
    print("\n" + "="*40)
    print("Input C")
    c = input_int()
    print("\nInput list size")
    n = input_int()
    while n <= 0:
        print("Input n > 0")
        n = input_int()

    print("\n1 - Generate list")
    print("2 - Initialize manually")
    while True:
        flag = input()
        if flag == '1':
            arr = seq_generate(n)
            break
        elif flag == '2':
            arr = inp_seq_float(n)
            break
        else:
            print("Wrong num")
    PrintOptions()
    while True:
        flag = input()
        if flag == '1':
            print("\nNum of nums greater than C -", FindHigher(arr, c))
            PrintOptions()
        elif flag == '2':
            print("\nMultiplying numbers after the maximum number -", FindMul(arr))
            PrintOptions()
        elif flag == '3':
            print("\nList:", arr)
            PrintOptions()
        elif flag == '4':
            return
        else:
            print("Wrong num")
```

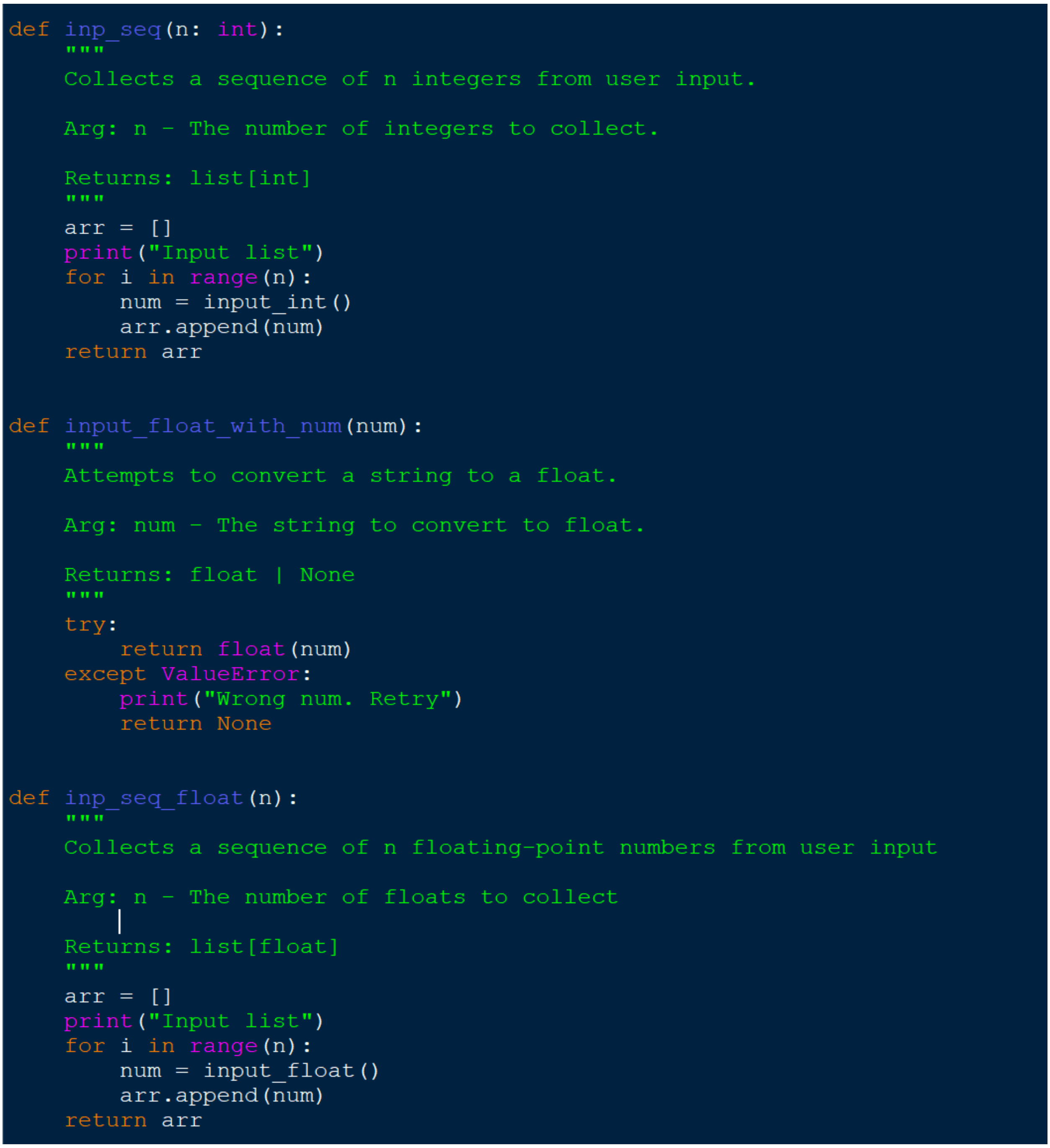


```
Input C
Input integer: 5

Input list size
Input integer: 6

1 - Generate list
2 - Initialize manually
1

1 - Get num of nums < 0 and < C
2 - Get multiplying numbers after the maximum number
3 - Get list
4 - End
1

Num of nums greater than C - 1

1 - Get num of nums < 0 and < C
2 - Get multiplying numbers after the maximum number
3 - Get list
4 - End
2
There are no elements after max elem

Multiplying numbers after the maximum number - 0

1 - Get num of nums < 0 and < C
2 - Get multiplying numbers after the maximum number
3 - Get list
4 - End
3

List: [1, 2, 3, 4, 5, 6]

1 - Get num of nums < 0 and < C
2 - Get multiplying numbers after the maximum number
3 - Get list
4 - End
```

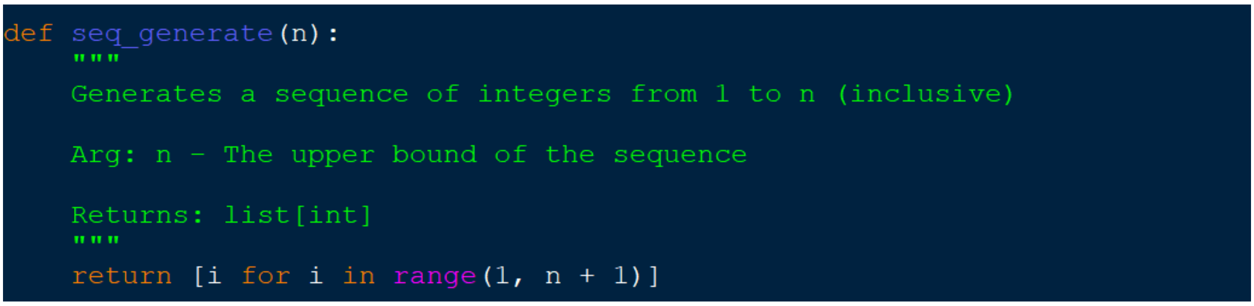
Функции проверки на ввод и инициализации списка.



```
"""
Lab Work #1
Gordashuk Vladislav
This module implements the functions of checking the input
and initialization of the list
Version: 1.0
Date: 30.03.2025
"""

def input_int():
    """
    Continuously prompts the user to input an integer until a valid one is entered.

    Arg: -

    Returns: int

    """
    while True:
        try:
            num = int(input("Input integer: "))
            return num
        except ValueError:
            print("Wrong num. Retry")


def input_float():
    """
    Continuously prompts the user to input a float until a valid one is entered.

    Arg: -

    Returns: float
    """
    while True:
        try:
            num = float(input("Input float: "))
            return num
        except ValueError:
            print("Wrong num. Retry")


def input_int_with_num(num: str):
    """
    Attempts to convert a string to an integer.

    Arg: num - The string to convert to integer.

    Returns: int | None
    """
    try:
        return int(num)
    except ValueError:
        print("Wrong num. Retry")
        return None
```

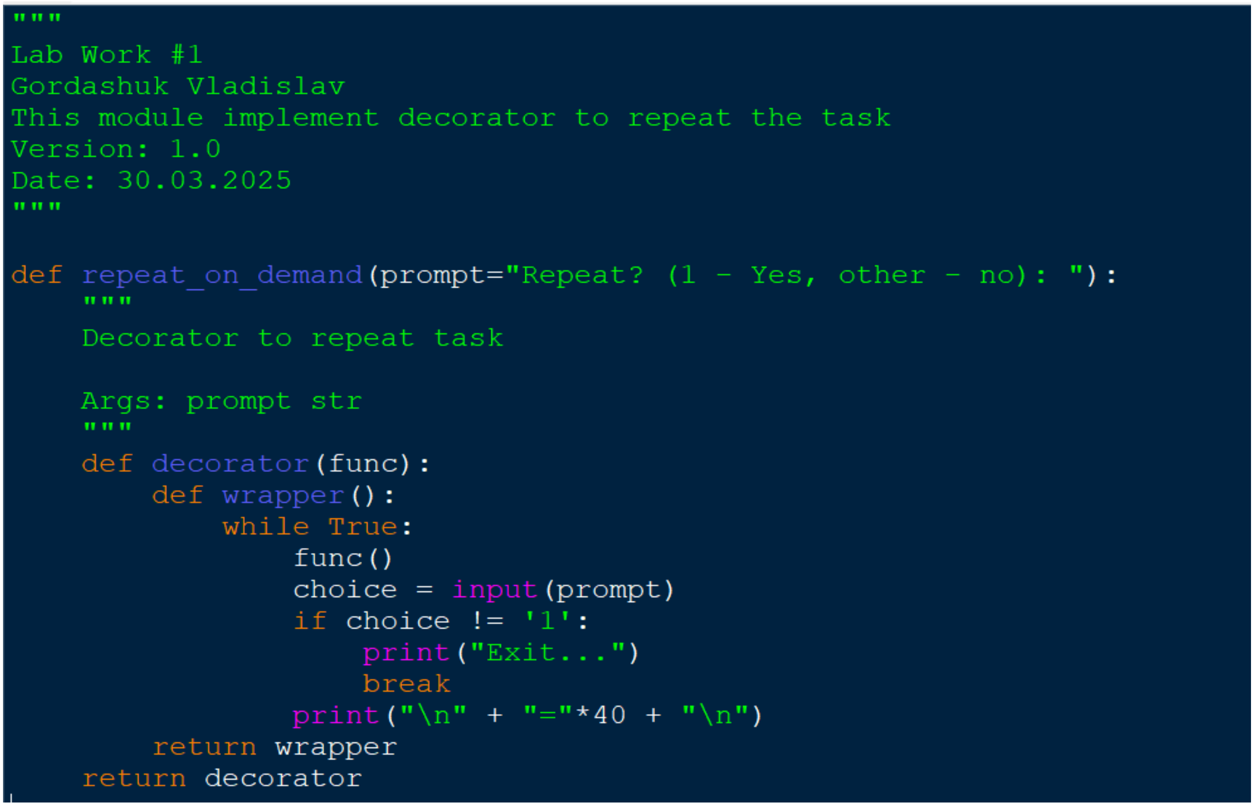


```
def inp_seq(n: int):
    """
    Collects a sequence of n integers from user input.

    Arg: n - The number of integers to collect.

    Returns: list[int]
    """
    arr = []
    print("Input list")
    for i in range(n):
        num = input_int()
        arr.append(num)
    return arr


def input_float_with_num(num):
    """
    Attempts to convert a string to a float.

    Arg: num - The string to convert to float.

    Returns: float | None
    """
    try:
        return float(num)
    except ValueError:
        print("Wrong num. Retry")
        return None


def inp_seq_float(n):
    """
    Collects a sequence of n floating-point numbers from user input

    Arg: n - The number of floats to collect

    Returns: list[float]
    """
    arr = []
    print("Input list")
    for i in range(n):
        num = input_float()
        arr.append(num)
    return arr
```



```
def seq_generate(n):
    """
    Generates a sequence of integers from 1 to n (inclusive)

    Arg: n - The upper bound of the sequence

    Returns: list[int]
    """
    return [i for i in range(1, n + 1)]
```

Декоратор.



```
"""
Lab Work #1
Gordashuk Vladislav
This module implement decorator to repeat the task
Version: 1.0
Date: 30.03.2025
"""

def repeat_on_demand(prompt="Repeat? (1 - Yes, other - no): "):
    """
    Decorator to repeat task

    Args: prompt str
    """
    def decorator(func):
        def wrapper():
            while True:
                func()
                choice = input(prompt)
                if choice != '1':
                    print("Exit...")
                    break
                print("\n" + "="*40 + "\n")
        return wrapper
    return decorator
```