Лабораторная работа №3

ДОВНАР А.Н.

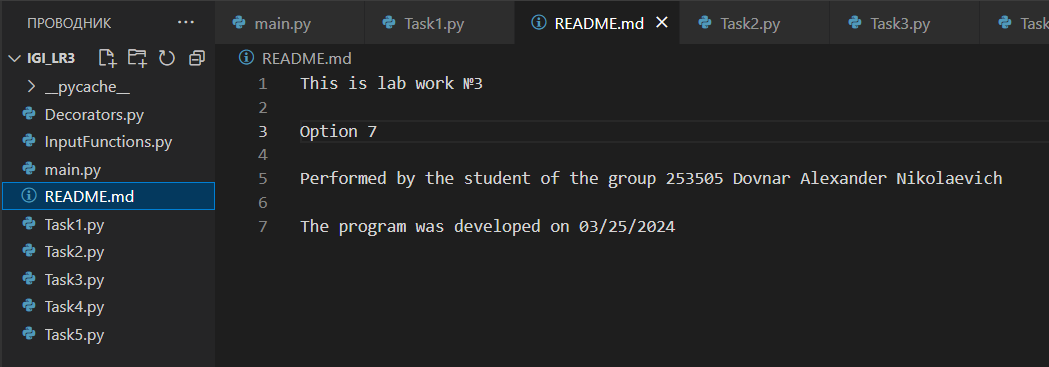
253505

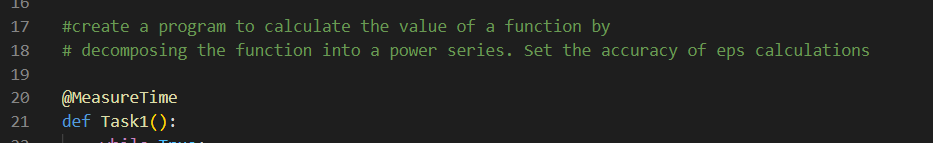
**Тема**: Стандартные типы данных, коллекции, функции, модули.

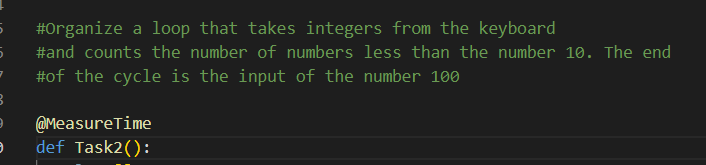
**Цель**: освоить базовый синтаксис языка Python, приобрести навыки работы со стандартными типами данных, коллекциями, функциями, модулями и закрепить их на примере разработки интерактивных приложений.

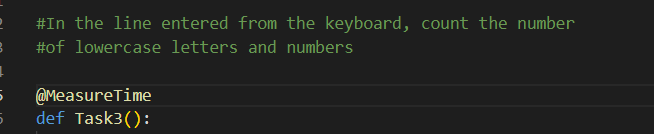
**Требования к выполнению**

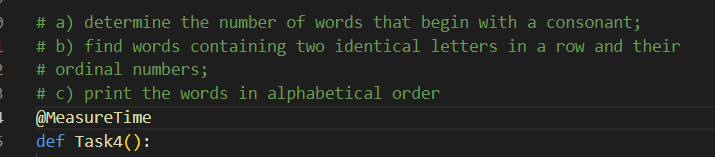
1. Программа должна быть снабжена комментариями на английском языке, в которых необходимо указать краткое предназначение программы, номер лабораторной работы и название, версию программы, Ф.И.О. разработчика и дату разработки.

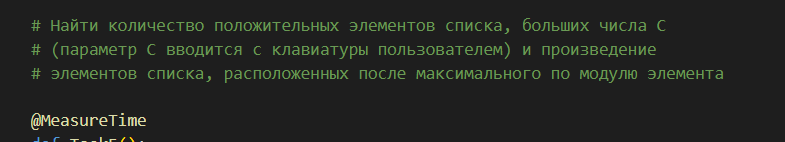




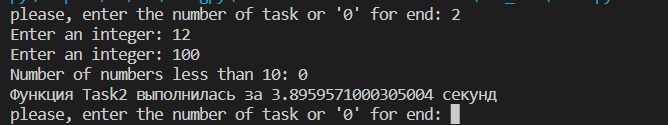




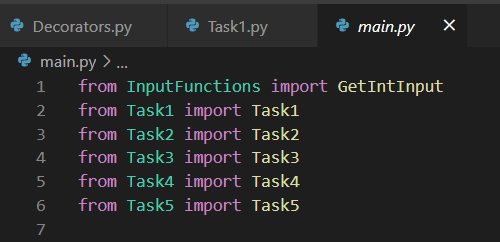


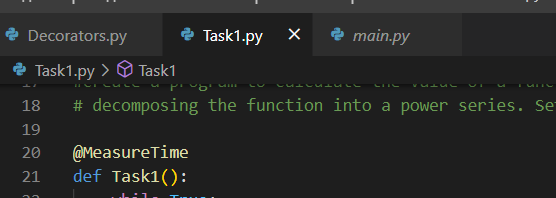


1. Программа должна быть снабжена дружелюбным и интуитивно понятным интерфейсом



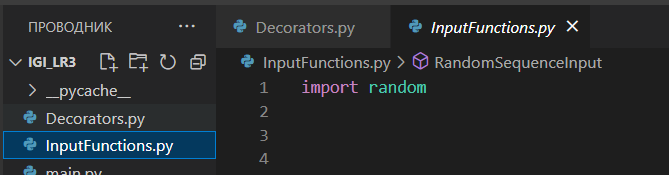
1. Выполнить документирование кода для получения справки по каждой функции
2. Каждое задание оформить в виде отдельной бизнес-функции.



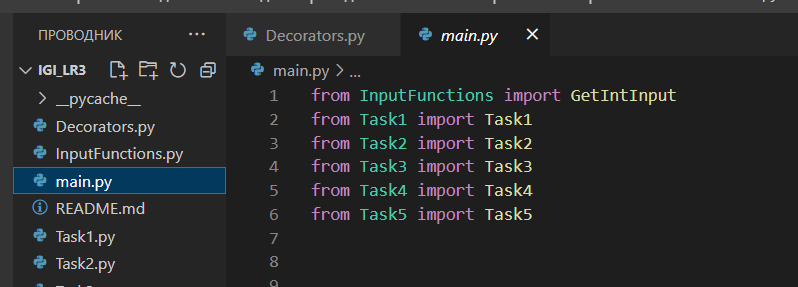


И т.д.

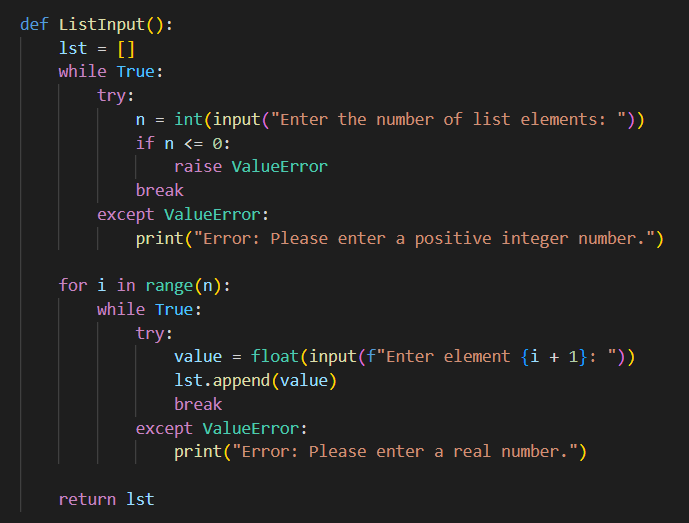
1. Все функции необходимо сгруппировать в модулях, согласно их логике их работы.



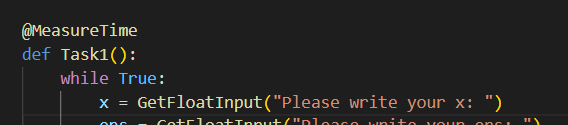
1. Разработанные основные функции, размещенные в отдельных модулях, нужно подключить в другом модуле, где будет происходить тестирование данных функций.



1. Размерность списка задается пользователем.

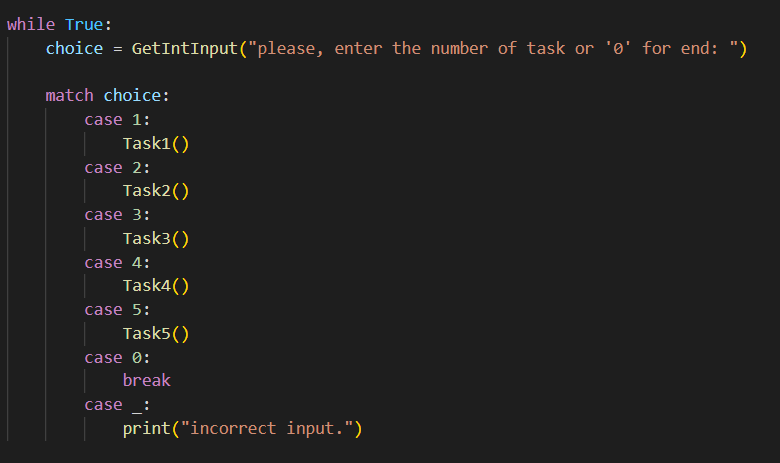


1. Продемонстрировать использование **декоратора** в любом из заданий





1. В программах предусмотреть возможность повторного выполнения без выхода из программы и защиту от ввода некорректных пользовательских данных. Для этих целей рекомендуется разработать отдельные функции.



Результаты работы программы:

TASK1

from math import cos, factorial

from Decorators import MeasureTime

from InputFunctions import GetFloatInput

def CalculateMyCos(x, eps):

    Result = 0

    n = 0

    while eps is not None and abs(cos(x) - Result) > eps and n + 1 < 500:

        Result += ((-1) \*\* n) \* ((x \*\* (2 \* n)) / factorial(2 \* n))

        n += 1

    if eps is not None:

        print(f"x = {x}, n = {n}, F(x) = {Result}, Math F(x) = {cos(x)}, eps = {eps}")

    else:

        print("Incorrect input")

    return

#create a program to calculate the value of a function by

# decomposing the function into a power series. Set the accuracy of eps calculations

@MeasureTime

def Task1():

    while True:

        x = GetFloatInput("Please write your x: ")

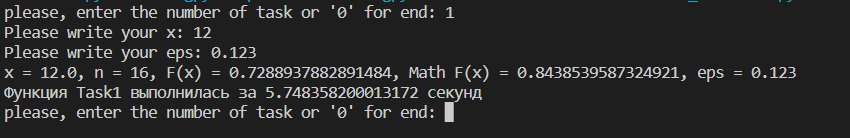
        eps = GetFloatInput("Please write your eps: ")

        if abs(eps) > 1 or eps < 0:

            print("Incorrect input!")

            continue

        CalculateMyCos(x, eps)

        return

TASK2

from Decorators import MeasureTime

from InputFunctions import GetIntInput

#Organize a loop that takes integers from the keyboard

#and counts the number of numbers less than the number 10. The end

#of the cycle is the input of the number 100

@MeasureTime

def Task2():

    lst=[]

    count = 0  # Initialize the counter

    while True:  # Infinite loop

        num = GetIntInput("Enter an integer: ")  # User input

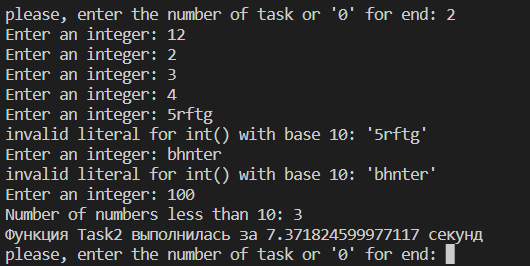
        if num == 100:  # Check for loop termination

            break

        if num < 10:  # Check the condition (number less than 10)

            count += 1  # Increment the counter

    print("Number of numbers less than 10:", count)  # Output the result



TASK3

from Decorators import MeasureTime

def FindingNumOfDigitsAndLowaerCase(str):

    count=0

    for char in str:

        if char.islower() or char.isdigit():

            count+=1

    print(f"Number of digits and lowercase letters: {count}")

    return

#In the line entered from the keyboard, count the number

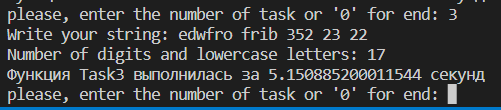
#of lowercase letters and numbers

@MeasureTime

def Task3():

    str = input("Write your string: ")

    FindingNumOfDigitsAndLowaerCase(str)



TASK4:

from Decorators import MeasureTime

from InputFunctions import GetIntInput

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."

def SplittingText(text):

    lst = text.split(" ")

    for i in range(len(lst)):

        lst[i] = lst[i].replace(",", "")

    return lst

def SearchNumOfWordsFirstConsonant(lst):

    consonant\_count = 0

    consonants = "bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ"

    for word in lst:

        if word and word[0] in consonants:

            consonant\_count += 1

    print("Number of words starting with a consonant:", consonant\_count)

def SearchWordsConsecutiveLetters(lst):

    matching\_words = []

    for i, word in enumerate(lst):

        for j in range(len(word) - 1):

            if word[j] == word[j + 1]:

                matching\_words.append((word, i + 1))

    print("Words with consecutive identical letters and their indices:")

    for word, index in matching\_words:

        print("Word:", word, "Index:", index)

    return

def SortWords(lst):

    word\_list = [word.lower() for word in lst]

        # Sorting words in alphabetical order

    word\_list.sort()

    print("Words in alphabetical order:")

    for word in word\_list:

        print(word)

# a) determine the number of words that begin with a consonant;

# b) find words containing two identical letters in a row and their

# ordinal numbers;

# c) print the words in alphabetical order

@MeasureTime

def Task4():

    lst=""

    choice=GetIntInput("how do you want to input text?\n\

                       1)From Task?\n\

                       2)User input\n\

                       0)Exit\n\

                       Write your answer: ")

    match choice:

        case 1:

            lst=text

        case 2:

            lst = input("Write your text: ")

        case 0:

            return

        case \_:

            print("incorrect input.")

    lst = SplittingText(text)

    while True:

        print("please, enter option from 1 to 3 or '0' to exit:\n\

                           1)count of words with first consonant\n\

                           2)Words with consecutive identical letters and their indices\n\

                           3)Words in alphabetical order \n")

        choice = GetIntInput("please, enter option from 1 to 3 or '0' to exit: ")

        if choice == 1:

            SearchNumOfWordsFirstConsonant(lst)

        elif choice == 2:

            SearchWordsConsecutiveLetters(lst)

        elif choice == 3:

            SortWords(lst)

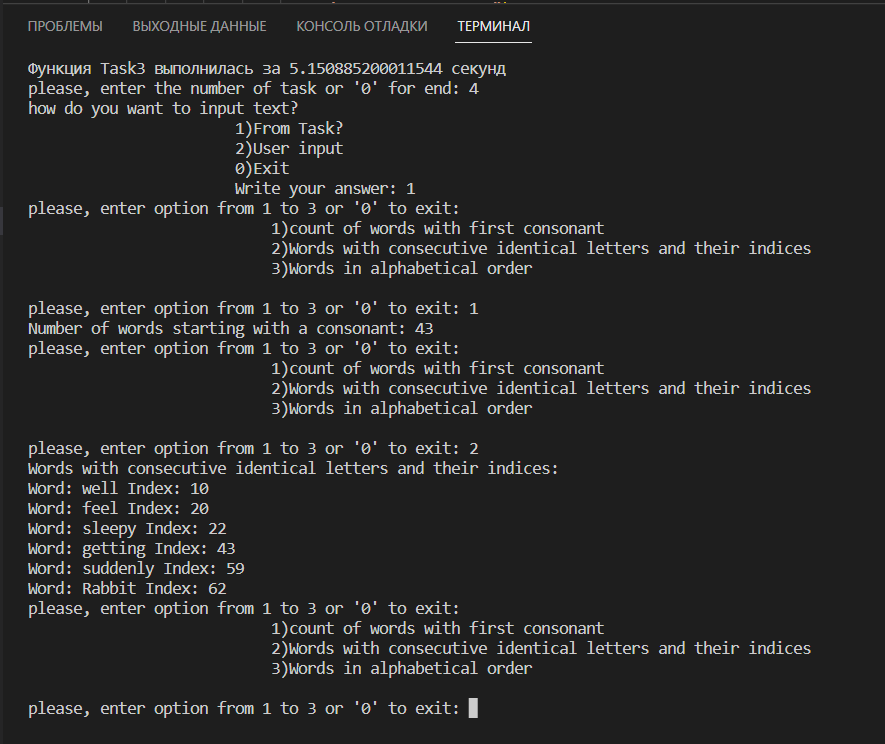
        elif choice == 0:

            return

        else:

            print("incorrect input.")

            continue



TASK5:

from InputFunctions import ListInput,GetFloatInput,GetIntInput,RandomSequenceInput

from Decorators import MeasureTime

def process\_list(lst, C):

    """Function to perform the main task"""

    positive\_count = 0

    product = 1.0

    max\_abs\_value = None

    max\_abs\_index = None

    for i, num in enumerate(lst):

        if num > C:

            positive\_count += 1

        if max\_abs\_value is None or abs(num) > abs(max\_abs\_value):

            max\_abs\_value = num

            max\_abs\_index = i

    if max\_abs\_index is not None and max\_abs\_index < len(lst) - 1:

        product = 1.0

        for num in lst[max\_abs\_index + 1:]:

            product \*= num

    print("Number of positive elements in the list greater than C:", positive\_count)

    print("Product of the elements in the list after the maximum absolute value:", product)

    return

# Найти количество положительных элементов списка, больших числа C

# (параметр C вводится с клавиатуры пользователем) и произведение

# элементов списка, расположенных после максимального по модулю элемента

@MeasureTime

def Task5():

    lst = []

    choice=GetIntInput("how do you want to complete the sequence?\n\

                       1)Random\n\

                       2)User input\n\

                       0)Exit\n")

    match choice:

        case 1:

            lst=RandomSequenceInput()

        case 2:

            lst = ListInput()

        case 0:

            return

        case \_:

            print("incorrect input.")

    C = GetFloatInput("Enter the number C: ")

    process\_list(lst,C)

    print(lst)

    return

