

Лабораторная работа №3

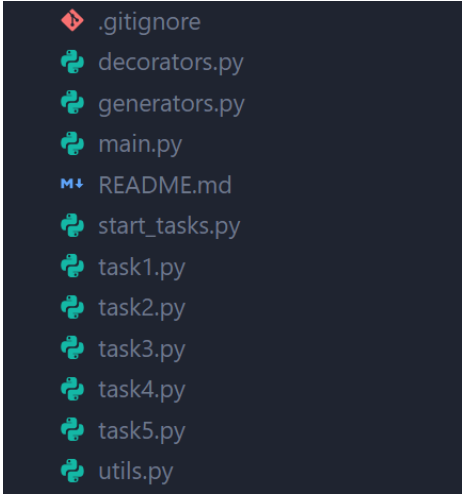
Тема: стандартные типы данных, коллекции, функции, модули.

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: студент группы 253502 Красёв П.А.

Вариант 11.

Структура лабораторной работы:



A screenshot of a file explorer window with a dark background. It lists the following files and folders: .gitignore (with a red diamond icon), decorators.py, generators.py, main.py (all with green Python icons), README.md (with a blue folder icon), start_tasks.py, task1.py, task2.py, task3.py, task4.py, task5.py, and utils.py (all with green Python icons).

-  .gitignore
-  decorators.py
-  generators.py
-  main.py
-  README.md
-  start_tasks.py
-  task1.py
-  task2.py
-  task3.py
-  task4.py
-  task5.py
-  utils.py

Файл main.py

```
1 from start_tasks import start_task1, start_task2, start_task3, start_task4, start_task5
2
3 # Lab3. Standard data types, collections, functions, modules.
4 # Performed by a student of group 253502, Krasnov Pavel.
5 # Fulfillment date: 21.03.2024
6
7
8 def main():
9     task = -1
10    while (True):
11        print("Enter the number of task from 1 to 5, or 0 to exit:")
12        try:
13            task = int(input())
14            if task < 0 or task > 5:
15                raise Exception
16        except:
17            print(
18                'Incorrect input, try again! Input only one number between 1 and 5, or 0 to exit\n')
19            continue
20        if task == 1:
21            print('Task 1')
22            start_task1()
23            continue
24        elif task == 2:
25            print('Task 2')
26            start_task2()
27            continue
28        elif task == 3:
29            print('Task 3')
30            start_task3()
31            continue
32        elif task == 4:
33            print('Task 4')
34            start_task4()
35            continue
36        elif task == 5:
37            print('Task 5')
38            start_task5()
39            continue
40        elif task == 0:
41            break
42    print('Program finished!')
43
44
45 if __name__ == '__main__':
46     main()
```

Функции генераторы для создания последовательностей:

```
1 import random
2 import string
3
4
5 def int_sequence(size: int):
6     ''' Function for generating int sequence of arbitrary size\
7     Numbers are generated in the interval [-150, 150]. '''
8     for i in range(size):
9         yield random.randint(-150, 150)
10
11
12 def literal_sequence(size: int) -> str:
13     ''' Function for generating string of arbitrary size\
14     Using all printable ASCII characters. '''
15     return ''.join(random.choices(string.printable, k=size))
16
17
18 def float_sequence(size: int):
19     ''' Function for generating float sequence of arbitrary size\
20     Numbers are generated in the interval [-10, 10] with 2 decimal places. '''
21     for i in range(size):
22         yield float(format(random.uniform(-10, 10), '.2f'))]
```

Декораторы, используемые в лабораторной:

```
1 from utils import stopFunction
2
3
4 def task1Decorator(func) -> None:
5     ''' Decorator for displaying task1 info. '''
6     def printInfo():
7         help(func)
8         print("Function: ln((x+1)/(x-1))")
9         print("Taylor's row: 2*sum(1/(2*n+1)*x^(2n+1), n=0..inf)\n")
10        func()
11    return printInfo
12
13
14 def chooseInputType(func) -> None:
15     ''' Decorator for checking the choice between 1 and 2 (Input by keyboard or random generator) and running task. '''
16     def inputType():
17         help(func)
18         while True:
19             while True:
20                 print("Enter your input type:\n1 - Keyboard\n2 - Random generator\n")
21                 try:
22                     choice = int(input())
23                     if choice != 1 and choice != 2:
24                         raise Exception
25                 except:
26                     print('Incorrect input, try again! Input only one number\n1 - Keyboard\n2 - Random generator\n')
27                     continue
28                 break
29             res = func(choice)
30             if res is None and stopFunction() != "0":
31                 print()
32                 continue
33             break
34         print('Task exit\n')
35    return inputType
36
```

Функции проверки ввода:

```
1 def stopFunction() -> str:
2     ''' Function stops running task. '''
3     return input('Do you want to continue? (0 -> No | anything else -> Yes): ')
4
5
6 def validateNumberInput(inputMessage: str, errorMessage: str, numberType, condition=lambda x: type(x) == str) -> int:
7     ''' Function validates input of a number with a given type and condition. '''
8     while True:
9         try:
10            N = numberType(
11                input(f'Enter {inputMessage}: ')
12            )
13            if condition(N):
14                print(f'Wrong input -> ({errorMessage})! Try again.\n')
15                continue
16            except ValueError:
17                print('Wrong input! Try again.\n')
18                continue
19            break
20    return N
21
22
23 def validateSequenceSize(name: str) -> int:
24     ''' Function validates input of size of sequence. '''
25     return validateNumberInput(
26         f'size of {name}', 'size > 0', int, condition=lambda x: x <= 0)
27
```

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. Предусмотреть максимальное количество итераций, равное 500.

$$\ln((x + 1)/(x - 1)) = 2 * \sum_{n=0}^{\infty} 1/((2n + 1) * x^{2n+1}), |x| > 1$$

Функция, запускающая задание:

```
11 @task1Decorator
12 def start_task1() -> None:
13     ''' A program for calculating the value of a function using a Taylor's row of the function
14     with a given accuracy. '''
15
16     while True:
17         eps, x = task1.validateInput()
18         if eps == 0:
19             break
20         task1.displayResult(x, eps)
21         if stopFunction() != "0":
22             print()
23             continue
24         break
25     print('Task exit\n')
```

Функции проверки ввода, подсчёта суммы ряда Тейлора и вывода границ таблицы:

```
1 from math import log
2 from utils import validateNumberInput
3
4
5 def validateInput() -> tuple[float, float]:
6     ''' Function for validation input of accuracy and x value. '''
7     eps = validateNumberInput(
8         'accuracy (0 < acc < 1) (0 -> Exit function)', '0 < acc < 1', float, condition=lambda x: x < 0 or x >= 1)
9     if eps == 0:
10         return eps, 0
11     x = validateNumberInput(
12         'x (|x| > 1)', '|x| must be bigger than 1', float, condition=lambda x: abs(x) <= 1)
13     return eps, x
14
15
16 def countTaylor(x: float, eps: float, mathResult: float) -> tuple[float, int]:
17     ''' Function that counts sum of Taylor's row with the required accuracy. '''
18     tailorSum = 0
19     N = -1
20     while N + 1 < 500:
21         N += 1
22         tailorSum += 1/((2*N+1)*x**(2*N+1))
23         if (abs(2*tailorSum - mathResult) >= eps/10):
24             continue
25         break
26     return 2*tailorSum, N+1
27
28
29 def printBorders(type: str, length: int) -> None:
30     ''' Functions prints result table borders. '''
31     for i in range(5):
32         if i == 0:
33             print("┌" + "-" * length,
34                 end="┐") if type == 'top' else print("\n┌" + "-" * length, end="┐") \
35                 if type == 'mid' else print("\n┌" + "-" * length, end="┐")
36         elif i != 4:
37             print("-" * (length + 3), end="┐") if type == 'top' else print("-" * (length + 3), end="┐") \
38             if type == 'mid' else print("-" * (length + 3), end="┐")
39         else:
40             print("-" * (length + 3), end="┐\n") if type == 'top' else print("-" * (length + 3), end="┐\n") \
41             if type == 'mid' else print("-" * (length + 3), end="┐\n")
42
```

Основная функция задания:

```
44 def displayResult(x: float, eps: float) -> None:
45     ''' Function displays th result. '''
46     mathResult = log((x+1)/(x-1))
47     tailorSum, iterations = countTailor(x, eps, mathResult)
48     length = max(len(str(tailorSum)), len(str(iterations)),
49                 len(str(eps)), len(str(x)), 9)
50     data = [{"x", "n", "F(x)", "Math F(x)", "eps"}, [
51         x, iterations, round(tailorSum, len(str(eps))), round(mathResult, len(str(eps))), eps]]
52     printBorders('top', length)
53     for i in range(2):
54         for j in range(5):
55             if j == 0:
56                 print("| " + str(data[i][j]) + " " *
57                     (length - len(str(data[i][j]))), end=" | ",)
58             else:
59                 print(str(data[i][j]).format() + " " * (length + 3 -
60                     len(str(data[i][j]))), end=" | ")
61     printBorders('mid', length) if i != 1 else printBorders('bot', length)
62
```

Пример использования:

```
PS D:\IGI\253502_KRASOV_11\IGI\LR3> python main.py
Enter the number of task from 1 to 5, or 0 to exit:
1
Task 1
Help on function start_task1 in module start_tasks:

start_task1() -> None
    A program for calculating the value of a function using a Taylor's row of the function
    with a given accuracy.

Function: ln((x+1)/(x-1))
Taylor's row: 2*sum(1/(2*n+1)*x^(2n+1), n=0..inf)
Enter accuracy (0 < acc < 1) (0 -> Exit function): 0.0001
Enter x (|x| > 1): 2



| x   | n | F(x)     | Math F(x) | eps    |
|-----|---|----------|-----------|--------|
| 2.0 | 7 | 1.098607 | 1.098612  | 0.0001 |


Do you want to continue? (0 -> No | anything else -> Yes): yes

Enter accuracy (0 < acc < 1) (0 -> Exit function): 0.001
Enter x (|x| > 1): 1.125



| x     | n  | F(x)    | Math F(x) | eps   |
|-------|----|---------|-----------|-------|
| 1.125 | 31 | 2.83313 | 2.83321   | 0.001 |


Do you want to continue? (0 -> No | anything else -> Yes): 0
Task exit

Enter the number of task from 1 to 5, or 0 to exit:
0
Program finished!
PS D:\IGI\253502_KRASOV_11\IGI\LR3>
```

Задание 2. В соответствии с заданием своего варианта составить программу для нахождения суммы последовательности чисел. Организовать цикл, который принимает целые числа и вычисляет наименьшее из них

Функция, запускающая задание:

```
28 @chooseInputType
29 def start_task2(inputType: int) -> None:
30     ''' Program for finding the sum of a sequence of integer numbers and minimal number. '''
31
32     if (inputType == 1):
33         print('Filling an array (0 -> stop enter):')
34         inputArray = task2.validateInput()
35     else:
36         inputArray = int_sequence(validateSequenceSize('sequence'))
37     task2.displayResults(inputArray)
```

Функция проверки ввода и основная функция задания:

```
1 from utils import validateNumberInput
2
3
4 def validateInput() -> list[int]:
5     ''' Function for validation input of sequence. '''
6     collection: list[int] = list()
7     while True:
8         n = validateNumberInput(
9             'the integer number', '', int)
10        if n == 0:
11            break
12        collection.append(n)
13        continue
14    return collection
15
16
17 def displayResults(inputArray: list[int]) -> None:
18     ''' Function analysing sequence and displays results. '''
19     print('Input array:')
20     for i in range(len(inputArray)):
21         print(f'Element #{i + 1} -> {inputArray[i]}')
22     if (len(inputArray) > 0):
23         print(f"\nSum of elements: {sum(inputArray)}")
24         print(f"Minimal value: {min(inputArray)}")
25     else:
26         print('Array is empty!')
```

Пример использования:

```

PS D:\IGI\253502_KRASOV_11\IGI\LR3> python main.py
Enter the number of task from 1 to 5, or 0 to exit:
2
Task 2
Help on function start_task2 in module start_tasks:

start_task2(inputType: int) -> None
    Program for finding the sum of a sequence of integer numbers and minimal number.

Enter your input type:
1 - Keyboard
2 - Random generator

1
Filling an array (0 -> stop enter):
Enter the integer number: 1
Enter the integer number: 2
Enter the integer number: 3
Enter the integer number: -4
Enter the integer number: 5
Enter the integer number: -6
Enter the integer number: 0
Input array:
Element #1 -> 1
Element #2 -> 2
Element #3 -> 3
Element #4 -> -4
Element #5 -> 5
Element #6 -> -6

Sum of elements: 1
Minimal value: -6
Do you want to continue? (0 -> No | anything else -> Yes): 1

Enter your input type:
1 - Keyboard
2 - Random generator

2
Enter size of sequence: 6
Input array:
Element #1 -> -110
Element #2 -> -4
Element #3 -> 146
Element #4 -> -121
Element #5 -> 77
Element #6 -> 70

Sum of elements: 58
Minimal value: -121
Do you want to continue? (0 -> No | anything else -> Yes): 0
Task exit

Enter the number of task from 1 to 5, or 0 to exit:
0
Program finished!
PS D:\IGI\253502_KRASOV_11\IGI\LR3>

```

Задание 3. В соответствии с заданием своего варианта составить программу для анализа текста, вводимого с клавиатуры. В строке, введенной с клавиатуры, подсчитать количество заглавных гласных английских букв.

Функция, запускающая задание:

```

40 @chooseInputType
41 def start_task3(inputType: int) -> None:
42     ''' Program for analyzing text entered from the keyboard: counts the number of capital vowels. '''
43
44     if (inputType == 1):
45         inputText = input('Input your string. (Type `stop` to exit): ')
46     else:
47         inputText = literal_sequence(validateSequenceSize('string'))
48     if inputText.lower() == 'stop':
49         return 'stop'
50     task3.displayResult(inputText)

```

Функции форматирования введенной строки, подсчёта букв и вывода:

```
1 def reformString(inputText: str) -> str:
2     terminators = ['\n', '\r', '\t', '\0', '\x0a',
3                     '\x0b', '\x0c', '\x0d', '\x08', '\x09', '\x1b']
4     for el in terminators:
5         inputText = inputText.replace(el, ' ')
6     return inputText
7
8
9 def countInputVowels(inputText: str) -> tuple[dict[str, int], dict[str, list[int]]]:
10     ''' Function counts ammount of each vowel and their positions in the input string. '''
11     vowels = ['A', 'E', 'I', 'O', 'U']
12     countVowels: dict[str, int] = dict.fromkeys(vowels, 0)
13     positions: dict[str, list[int]] = {v: list() for v in vowels}
14     for i in range(len(inputText)):
15         if inputText[i] in vowels:
16             countVowels[inputText[i]] += 1
17             positions[inputText[i]].append(i)
18     return countVowels, positions
19
20
21 def displayPositions(inputText: str, positions: list[int]) -> None:
22     ''' Function displays the positions of vowel in the string. '''
23     if len(positions) == 0:
24         print('No references\n')
25         return
26     print(inputText)
27     for i in range(len(inputText)):
28         if i not in positions:
29             print(' ', end='')
30             continue
31         print('*', end='')
32         positions.remove(i)
33         if len(positions) == 0:
34             print()
35             break
```

Основная функция задания:

```
38 def displayResult(inputText: str) -> str | None:
39     ''' Fucntion displays the result. '''
40     print('Input string:')
41     print(repr(inputText))
42     countVowels, positions = countInputVowels(inputText)
43     inputText = reformString(inputText)
44     for vowel in countVowels.keys():
45         print(f'Vowel: `{vowel}` -> Ammount: {countVowels[vowel]}')
46         displayPositions(inputText, positions[vowel])
47
```

Пример использования:


```

PS D:\IGI\253502_KRASYOV_11\IGI\LR3> python main.py
Enter the number of task from 1 to 5, or 0 to exit:
3
Task 3
Help on function start_task3 in module start_tasks:

start_task3(inputType: int) -> None
    Program for analyzing text entered from the keyboard: counts the number of capital vowels.

Enter your input type:
1 - Keyboard
2 - Random generator

1
Input your string. (Type `stop` to exit): A\nA\nA\tUII\0E\rE\nFFF
Input string:
'A\nA\nA\tUII\0E\rE\nFFF'
Vowel: `A` -> Ammount: 3
A\nA\nA\tUII\0E\rE\nFFF
* * *
Vowel: `E` -> Ammount: 2
A\nA\nA\tUII\0E\rE\nFFF
* *
Vowel: `I` -> Ammount: 2
A\nA\nA\tUII\0E\rE\nFFF
**
Vowel: `O` -> Ammount: 0
No references

Vowel: `U` -> Ammount: 1
A\nA\nA\tUII\0E\rE\nFFF
*
Do you want to continue? (0 -> No | anything else -> Yes): 1

Enter your input type:
1 - Keyboard
2 - Random generator

2
Enter size of string: 20
Input string:
"\nZ44'Ir<_B>G=i1rLBeR"
Vowel: `A` -> Ammount: 0
No references

Vowel: `E` -> Ammount: 0
No references

Vowel: `I` -> Ammount: 1
Z44'Ir<_B>G=i1rLBeR
*
Vowel: `O` -> Ammount: 0
No references

Vowel: `U` -> Ammount: 0
No references

Do you want to continue? (0 -> No | anything else -> Yes): 0

```

Задание 4. 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.»

- определить количество слов в строке;
- найти самое длинное слово и его порядковый номер;
- вывести каждое нечетное слово

Функция, запускающая задание:

```
53 def start_task4() -> None:
54     ''' Program for parsing a string initialized in program code:\na) determine the number of words in a line;\n
55 b) find the longest word and its serial number;\nc) print every odd word '''
56     task4.analiseString()
57     print('Task exit\n')
```

Функции очистки строки от знаков препинания, поиска самого длинного слова, вывода нечётных слов и основная функция задания:

```
4 def clearString() -> list[str]:
5     ''' Function converts a string to replace all non-letter characters with spaces. '''
6     clearString = "".join(
7         el for el in initialString if el.isalpha() or el == ' ' or el == '-')
8     return clearString.split(' ')
9
10
11 def findMaxLenWord(words: list[str]) -> None:
12     ''' Function finds max len word and its position in the string. '''
13     maxLenWords: list[tuple[int, str]] = [(0, '')]
14     for id, word in enumerate(words):
15         if len(word) > len(maxLenWords[0][1]):
16             maxLenWords.clear()
17             maxLenWords.append((id, word))
18         elif len(word) == len(maxLenWords[0][1]):
19             maxLenWords.append((id, word))
20     print(
21         f"Max length word{'s' if len(maxLenWords) > 1 else ''} => Max length: {len(maxLenWords[0][1]):}")
22     for id, word in maxLenWords:
23         print(f'{word} -> Position: {id}')
24
25
26 def displayOddWords(words: list[str]) -> None:
27     ''' Function prints odd words of the string. '''
28     print('Odd words:')
29     for i in range(0, len(words), 2):
30         print(words[i], end=" ")
31
32
33 def analiseString() -> None:
34     ''' Function analyses string and displays results. '''
35     print('Initial string:')
36     print(initialString)
37     print("\nAnalised string:")
38     words = clearString()
39     print(f'Ammount of words: {len(words)}')
40     findMaxLenWord(words)
41     displayOddWords(words)
42     print('Task exit\n')
```

Пример использования:

```
PS D:\IGI\253502_KRASVOY_11\IGI\LR3> python main.py
Enter the number of task from 1 to 5, or 0 to exit:
4
Task 4
Help on function start_task4 in module start_tasks:

start_task4() -> None
  Program for parsing a string initialized in program code:
  a) determine the number of words in a line;
  b) find the longest word and its serial number;
  c) print every odd word

Initial string:
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.

Analised string:
Ammount of words: 55

Max length words => Max length: 11:
considering -> Position: 3
daisy-chain -> Position: 30

Odd words:
So was in own as as could the day her very and whether pleasure making daisy-chain be the of up picking daisies suddenly White with eyes close her Task exit

Task exit

Enter the number of task from 1 to 5, or 0 to exit:
0
Program finished!
PS D:\IGI\253502_KRASVOY_11\IGI\LR3>
```

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 элементов списка, лежащих в диапазоне от А до В (параметры А и В вводятся с клавиатуры пользователем) и сумму элементов списка, расположенных после максимального элемента

Функция запуска задания:

```
60 @chooseInputType
61 def start_task5(inputType: int) -> None:
62     ''' Program for processing real lists:\nthe number of list elements lying in the range from A to B (parameters A\
63 and B are entered from the keyboard by the user)\nand the sum of the list elements located after the maximum element '''
64
65     if (inputType == 1):
66         collection = task5.validateInputArray()
67     else:
68         collection = float_sequence(validateSequenceSize('sequence'))
69     if collection is None:
70         return 'stop'
71     task5.analiseCollection(collection)
```

Функции проверки ввода последовательности и границ диапазона:

```
4 def validateInputArray() -> list[float]:
5     ''' Function validates input of sequence and its lenght. '''
6     N = validateNumberInput(
7         'length of array (len > 0) (0 -> Exit function)', 'len > 0', int, condition=lambda x: x < 0)
8     if N == 0:
9         return None
10    k = 0
11    collection = list[float]()
12    while k < N:
13        el = validateNumberInput(
14            f'#{k + 1} element (float number)', '', float)
15        collection.append(el)
16        k += 1
17    return collection
18
19
20 def validateInputBorders() -> tuple[int, int]:
21     ''' Function validates input of interval borders. '''
22     while True:
23         A = validateNumberInput('left border: ', '', int)
24         B = validateNumberInput(
25             'right border: ', '', int)
26         if B < A:
27             print('Wrong input -> (left border <= right border)! Try again.')
28             continue
29         break
30    return A, B
```

Функция вывода последовательности, поиска максимального элемента и количества элементов в диапазоне

```
53 def displayCollection(collection: list[float]) -> None:
54     ''' Function displays collection. '''
55     print('Entered collection:')
56     for el in collection:
57         print(el, end=" ")
58     print()
59
60
61 def findAmountInInterval(collection: list[float], A: int, B: int) -> int:
62     ''' Function counts an amount of collection elements in the interval. '''
63     return sum(1 if el >= A and el <= B else 0 for el in collection)
64
65
66 def findMaxElement(collection: list[float]) -> tuple[int, float]:
67     ''' Function finds collection max element and its position. '''
68     return max(enumerate(collection), key=lambda el: el[1])
69
70
71 def findSumInterval(collection: list[float], start: int) -> float:
72     ''' Function finds sum of elements from the `start` position to the end of collection. '''
73     return sum(collection[i] for i in range(start, len(collection)))
```

Основная функция задания

```
56 def analyseCollection(collection: list[float]) -> str | None:
57     ''' Function analyses function and displays results. '''
58     displayCollection(collection)
59     A, B = validateInputBorders()
60     print('Analised collection:')
61     print(
62         f'Ammount of elements in [{A}, {B}]: {findAmountInInterval(collection, A, B)}')
63     maxElement = findMaxElement(collection)
64     print(f'Max element: {maxElement[1]}')
65     print(
66         f'Sum of elements after Max: {findSumInterval(collection, maxElement[0] + 1)}')
```

Пример использования:

```
PS D:\IGI\253502_KRASYOV_11\IGI\LR3> python main.py
Enter the number of task from 1 to 5, or 0 to exit:
5
Task 5
Help on function start_task5 in module start_tasks:

start_task5(inputType: int) -> None
    Program for processing real lists:
    the number of list elements lying in the range from A to B (parameters A and B are entered from the keyboard by the user)
    and the sum of the list elements located after the maximum element

Enter your input type:
1 - Keyboard
2 - Random generator

1
Enter length of array (len > 0) (0 -> Exit function): 10
Enter #1 element (float number): 1.25
Enter #2 element (float number): -34.08
Enter #3 element (float number): 5.3
Enter #4 element (float number): 3.33
Enter #5 element (float number): -0.3333
Enter #6 element (float number): 20.0
Enter #7 element (float number): 9.4
Enter #8 element (float number): 2
Enter #9 element (float number): 1.6
Enter #10 element (float number): 0
Entered collection:
1.25 -34.08 5.3 3.33 -0.3333 20.0 9.4 2.0 1.6 0.0
Enter left border: : -5
Enter right border: : 5
Analised collection:
Ammount of elements in [-5, 5]: 6
Max element: 20.0
Sum of elements after Max: 13.0
Do you want to continue? (0 -> No | anything else -> Yes): 1
```

```
Enter your input type:
1 - Keyboard
2 - Random generator

2
Enter size of sequence: 10
Entered collection:
1.44 -6.09 1.68 -6.94 7.52 6.88 4.4 6.2 -9.33 6.37
Enter left border: : -1 1
Wrong input! Try again.

Enter left border: : -1
Enter righth border: : 1
Analised collection:
Ammount of elements in [-1, 1]: 0
Max element: 7.52
Sum of elements after Max: 14.52
Do you want to continue? (0 -> No | anything else -> Yes): 0
Task exit

Enter the number of task from 1 to 5, or 0 to exit:
0
Program finished!
PS D:\IGI\253502_KRASYOV_11\IGI\LR3> █
```