Практическая №6 Задания 1,2,3.

Задача 1

Постановка задачи.

Дано целое число N (>2). Сформировать и вывести целочисленный список размера 10, содержащий 10 первых элементов последовательности чисел Фибоначчи FK: F1 = 1, F2 = 1, FK = FK-2 + FK-1, K = 3,4,... .

Текст программы:

```
#Дано целое число N (>2). Сформировать и вывести целочисленный список размера
#10, содержащий 10 первых элементов последовательности чисел Фибоначчи FK: F1
#= 1, F2 = 1, FK = FK-2 + FK-1, K = 3,4,... .
1 usage  new *
def fibonacci_sequence(n):
    fib_sequence = [1, 1]
    for k in range(2, n):
        next_fib = fib_sequence[k-1] + fib_sequence[k-2]
        fib_sequence.append(next_fib)
    return fib_sequence

N = int(input("Введите целое число N (> 2): "))
if N > 2:
    fib_list = fibonacci_sequence(10)
    print(fib_list)
else:
    print("Число N должно быть больше 2")
```

Протокол работы программы:

```
Введите целое число N (> 2): 3
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55]

Process finished with exit code 0
```

Задание 2

Постановка задачи:

Дан список A размера N и целые числа K и L (1 < K < L < N). Переставить в обратном порядке элементы списка, расположенные между элементами AK и AL, включая эти элементы.

Текст программы:

```python
#Дан список A размера N и целые числа K и L (1 < K < L < N). Переставить в обратном
#порядке элементы списка, расположенные между элементами AK и AL, включая эти
#элементы.
1 usage  new *
def reverse_elements_between(A, K, L):
    if 1 < K < L < len(A):
        sublist_to_reverse = A[K-1:L]
        A[K-1:L] = sublist_to_reverse[::-1]
    else:
        print("Условие 1 < K < L < N не выполняется")

input_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
K = 3
L = 7
reverse_elements_between(input_list, K, L)
print(input_list)
```

Протокол работы программы:

```
[1, 2, 7, 6, 5, 4, 3, 8, 9, 10]

Process finished with exit code 0
```

Задача 3

Постановка задачи.

Дан список размера N. Обнулить все его локальные максимумы (то есть числа, большие своих соседей)

Текст программы:

```python
#Дан список размера N. Обнулить все его локальные максимумы (то есть числа,большие своих соседей).
1 usage  new *
def zero_local_max(numbers):
    n = len(numbers)
    for i in range(1, n - 1):
        if numbers[i] > numbers[i - 1] and numbers[i] > numbers[i + 1]:
            numbers[i] = 0
    return numbers

input_list = [1, 3, 2, 5, 4, 6, 8, 5, 7]
result = zero_local_max(input_list)
print(result)
```

Протокол работы программы:

```
[1, 0, 2, 0, 4, 6, 0, 5, 7]

Process finished with exit code 0
```

Вывод: в процессе выполнения практического занятия выработал навыки составления программ с функциями в IDE PyCharm Community