

|  |
| --- |
|  |
| Язык и технология программирования |
| Лабораторная работа 9 |

|  |
| --- |
| Жолдас Қайсар ИС 24-22 |

Задание 1

def print\_numbers(n):

# Базовое условие — когда дошли до 1

if n == 1:

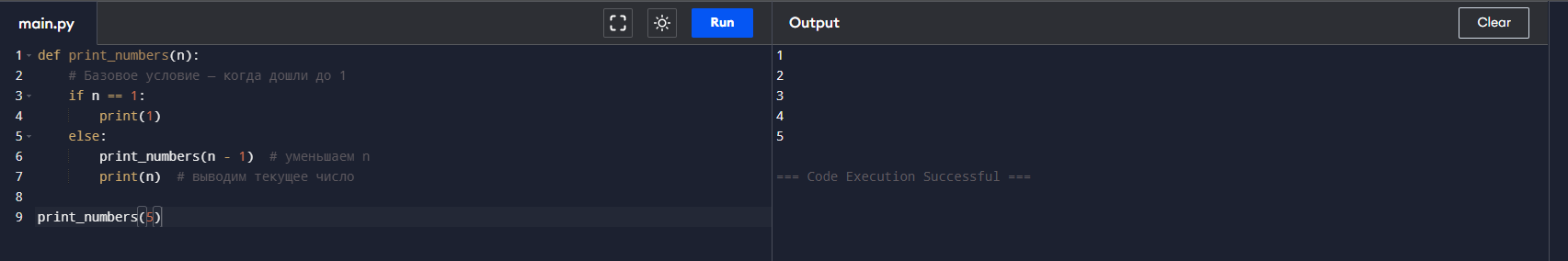
print(1)

else:

print\_numbers(n - 1) # уменьшаем n

print(n) # выводим текущее число

print\_numbers(5)



Функция вызывает саму себя, пока не дойдёт до 1.

После этого каждый уровень рекурсии «разворачивается», печатая числа по порядку.вычисляет факториал числа n с помощью рекурсивного вызова функции factorial().

Задание 2

def count\_chars(s):

# Базовое условие — если строка пустая

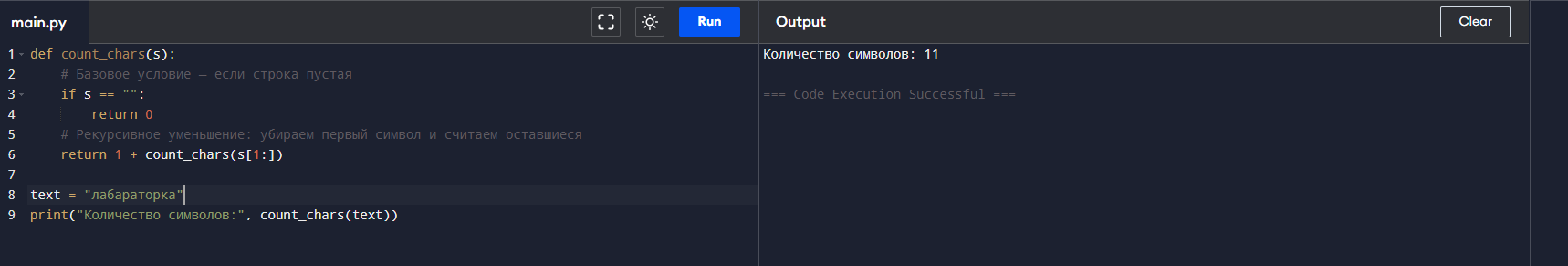
if s == "":

return 0

# Рекурсивное уменьшение: убираем первый символ и считаем оставшиеся

return 1 + count\_chars(s[1:])

text = "лабараторка"

print("Количество символов:", count\_chars(text))

Каждый вызов функции отрезает один символ от строки.

Когда строка становится пустой, функция возвращает 0 — рекурсия останавливается.

Задание 3

a = int(input())

b = int(input())

def power(a, b):

# Базовое условие — степень 0

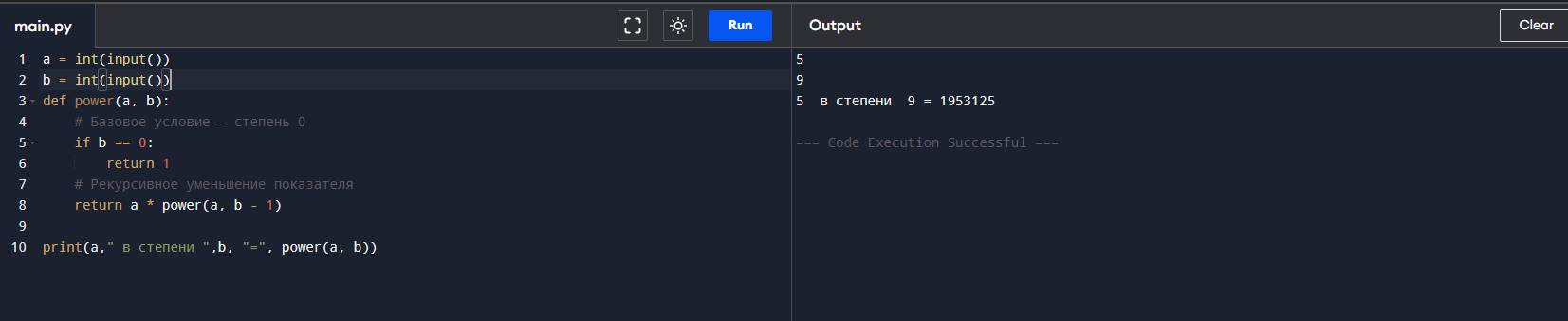
if b == 0:

return 1

# Рекурсивное уменьшение показателя

return a \* power(a, b - 1)

print(a," в степени ",b, "=", power(a, b))

 Каждый рекурсивный вызов умножает число a на результат от a^(b-1).

Когда показатель становится 0, возвращается 1 — это точка остановки.

Контрольные вопросы:

1. Что такое рекурсия?

Рекурсия — это метод, при котором функция вызывает саму себя для решения подзадачи, пока не достигнет базового условия.

2. Что представляет собой базовое условие рекурсии?

Это условие, при котором функция прекращает вызывать саму себя и возвращает результат.

3. Чем отличается итерация от рекурсии?

Итерация использует циклы (for, while), а рекурсия — вызовы самой функции.

4. Когда рекурсия неэффективна?

Когда глубина рекурсии велика и происходит переполнение стека (например, при вычислении больших чисел Фибоначчи без мемоизации).

5. Как можно оптимизировать рекурсивные функции?

Использовать мемоизацию, хвостовую рекурсию или заменить рекурсию итерацией.