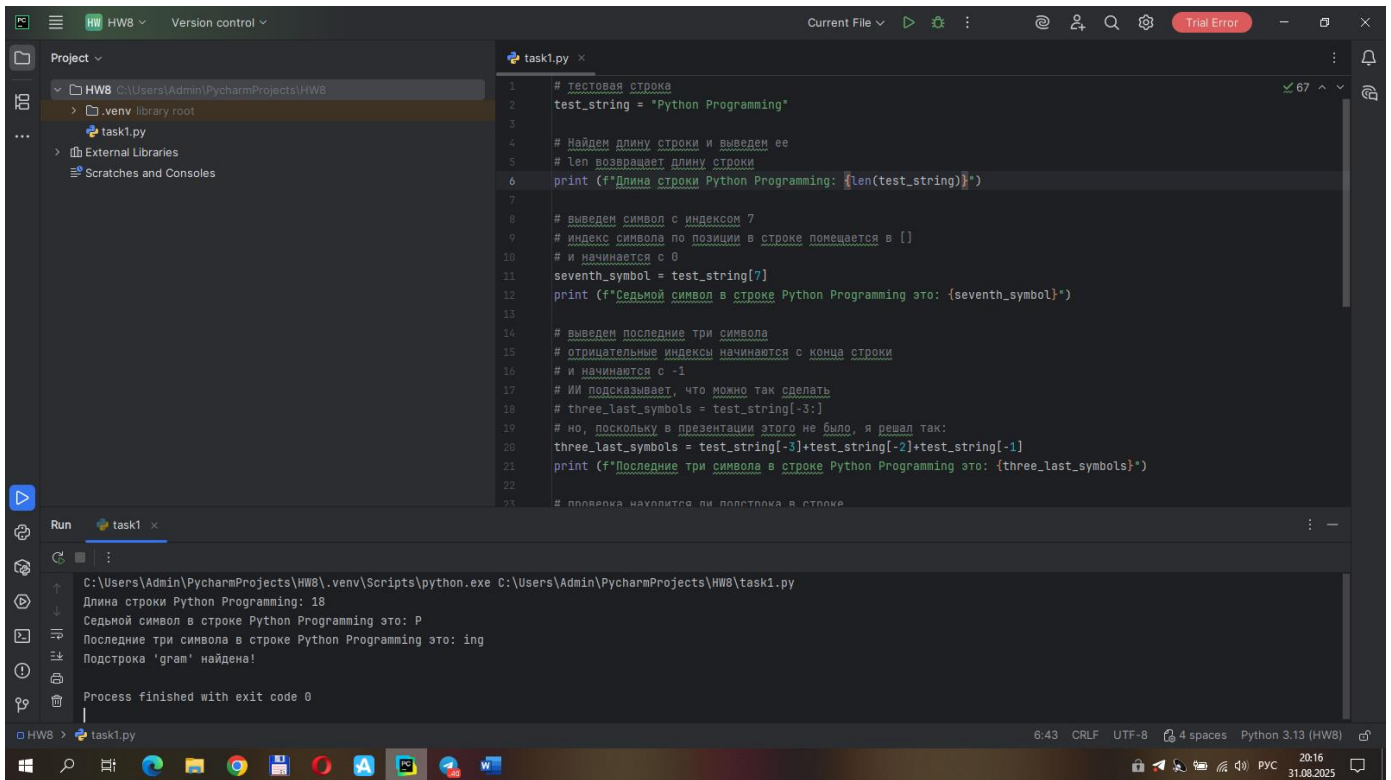


1. ЧАСТЬ 1. Строки, Списки, Словари.

Упражнение 1.



```
1 # тестовая строка
2 test_string = "Python Programming"
3
4 # Найдём длину строки и выведем ее
5 # len возвращает длину строки
6 print(f'Длина строки Python Programming: {len(test_string)}')
7
8 # выведем символ с индексом 7
9 # индекс символа по позиции в строке помещается в []
10 # и начинается с 0
11 seventh_symbol = test_string[7]
12 print(f'Седьмой символ в строке Python Programming это: {seventh_symbol}')
13
14 # выведем последние три символа
15 # отрицательные индексы начинаются с конца строки
16 # и начинаются с -1
17 # ИИ подсказывает, что можно так сделать
18 # three_last_symbols = test_string[-3:]
19 # но, поскольку в презентации этого не было, я решил так:
20 three_last_symbols = test_string[-3]+test_string[-2]+test_string[-1]
21 print(f'Последние три символа в строке Python Programming это: {three_last_symbols}')
22
23 # проверка находит ли подстрока в строке
```

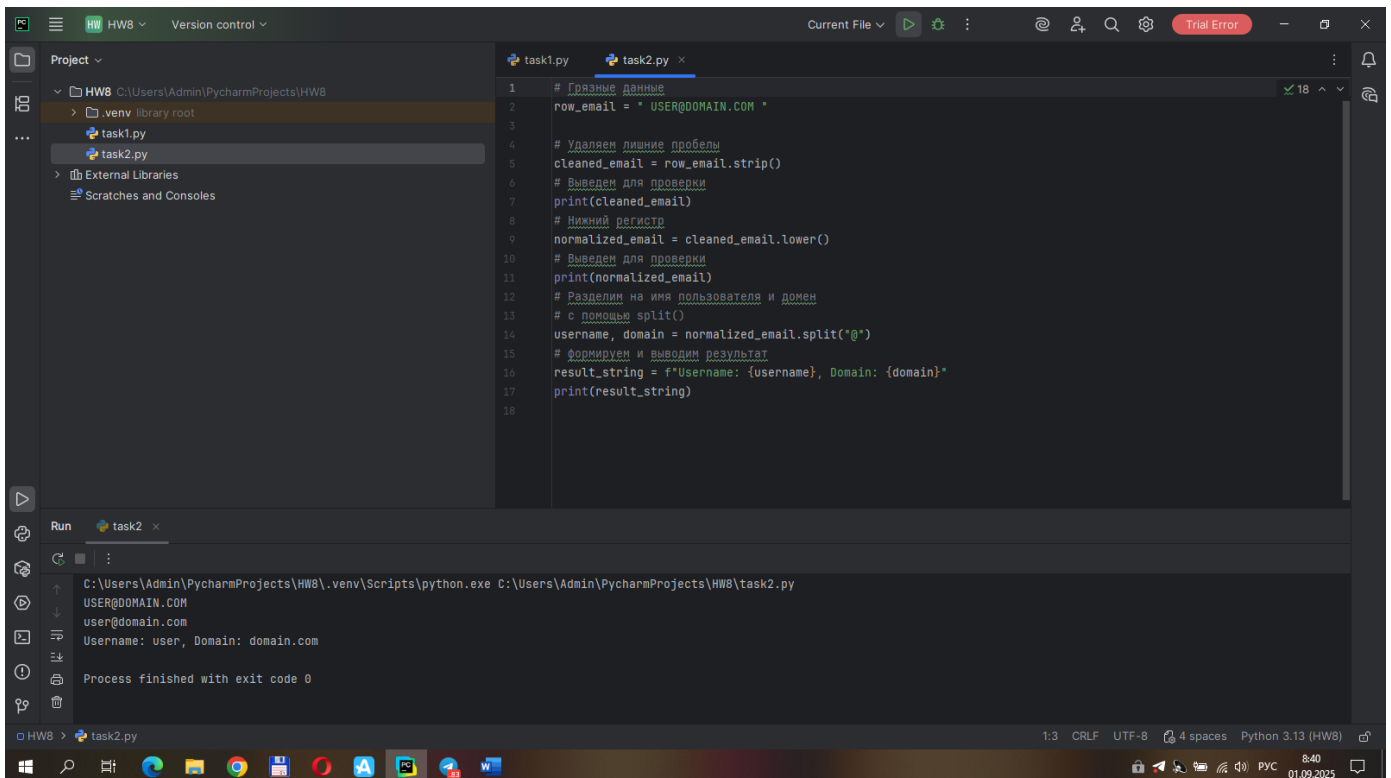
Run task1

C:\Users\Admin\PycharmProjects\HW8\.venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\HW8\task1.py

Длина строки Python Programming: 18
Седьмой символ в строке Python Programming это: P
Последние три символа в строке Python Programming это: ing
Подстрока 'gram' найдена!

Process finished with exit code 0

Упражнение 2.



```
1 # Грязные данные
2 row_email = " USER@DOMAIN.COM "
3
4 # Удаляем лишние пробелы
5 cleaned_email = row_email.strip()
6 # Выведем для проверки
7 print(cleaned_email)
8 # Нижний регистр
9 normalized_email = cleaned_email.lower()
10 # Выведем для проверки
11 print(normalized_email)
12 # Разделим на имя пользователя и домен
13 # с помощью split()
14 username, domain = normalized_email.split("@")
15 # Формируем и выводим результат
16 result_string = f"Username: {username}, Domain: {domain}"
17 print(result_string)
18
```

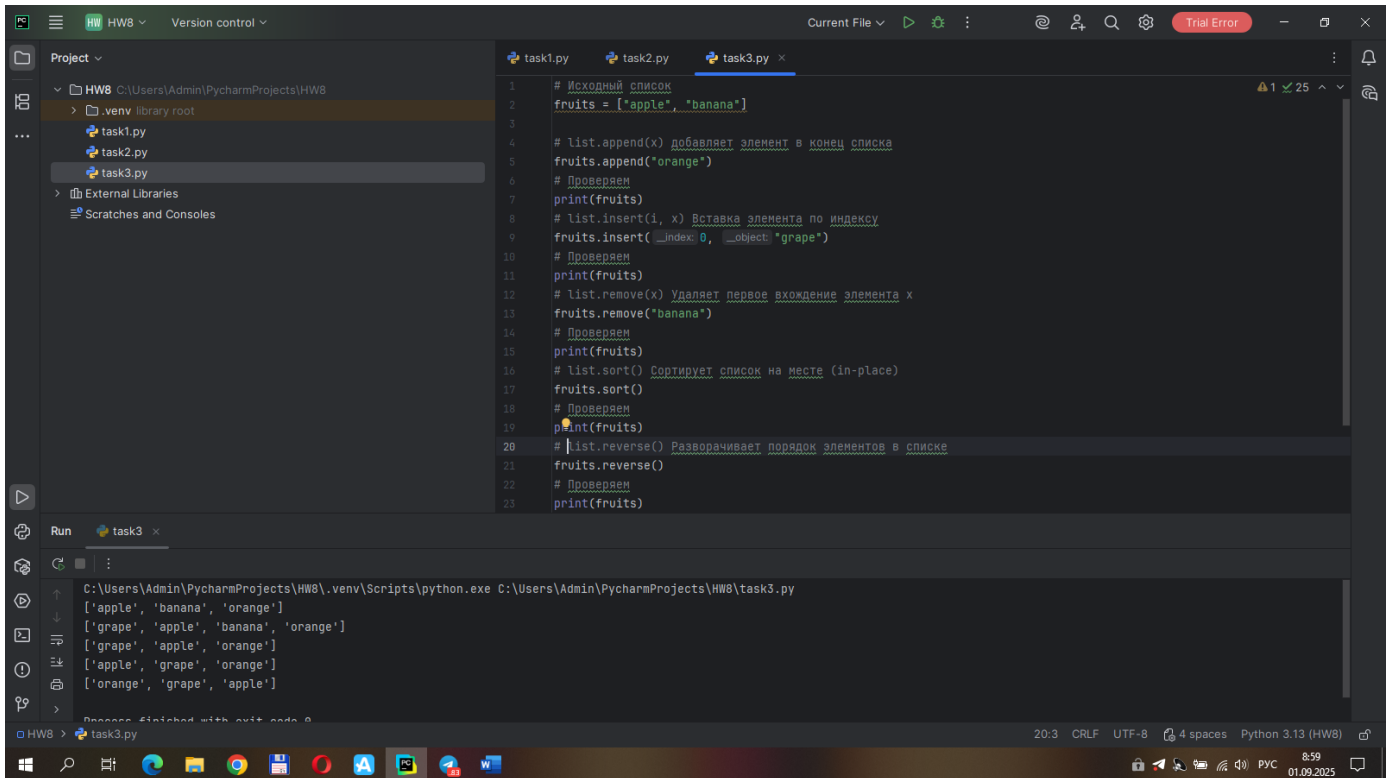
Run task2

C:\Users\Admin\PycharmProjects\HW8\.venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\HW8\task2.py

USER@DOMAIN.COM
user@domain.com
Username: user, Domain: domain.com

Process finished with exit code 0

Упражнение 3.

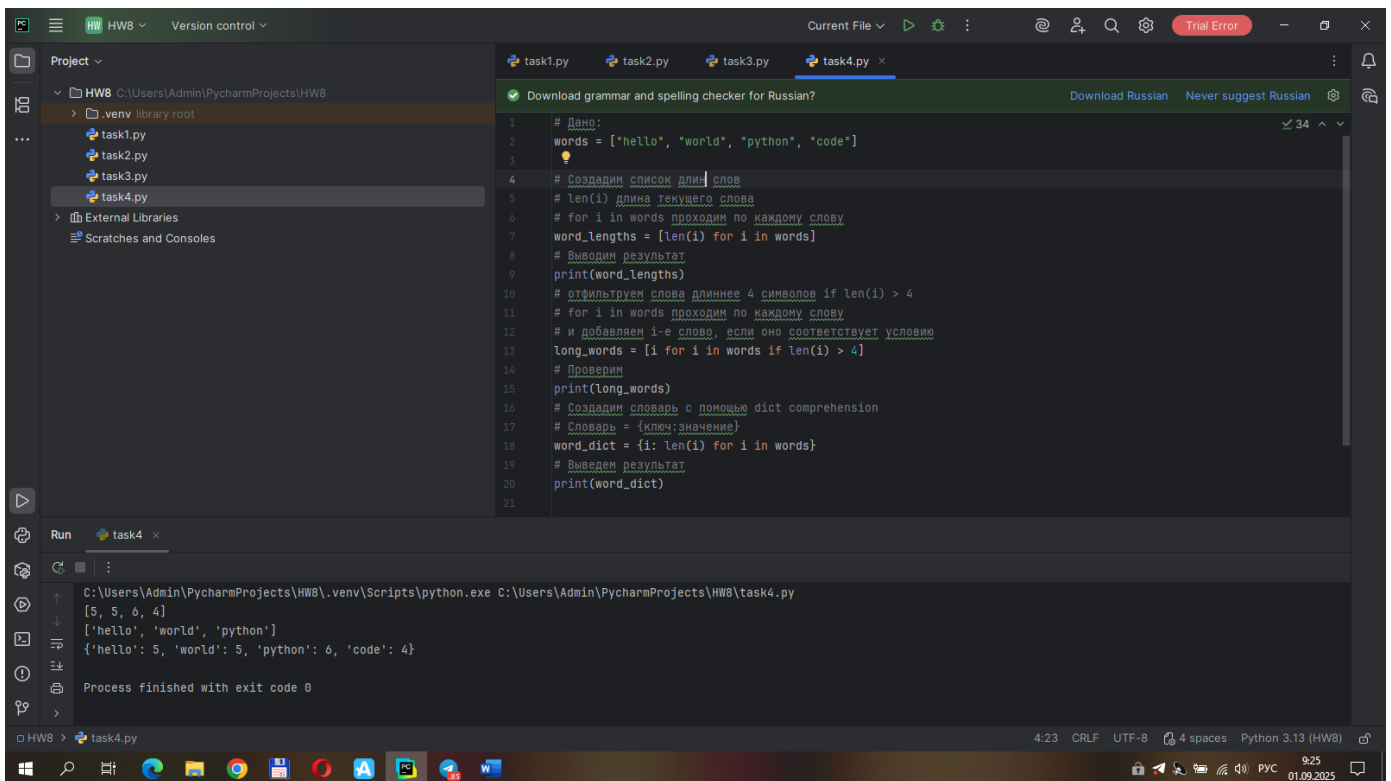


```
1 # Исходный список
2 fruits = ["apple", "banana"]
3
4 # list.append(x) добавляет элемент в конец списка
5 fruits.append("orange")
6 # Проверяем
7 print(fruits)
8 # list.insert(i, x) вставляет элемент по индексу
9 fruits.insert(0, "grape")
10 # Проверяем
11 print(fruits)
12 # list.remove(x) удаляет первое вхождение элемента x
13 fruits.remove("banana")
14 # Проверяем
15 print(fruits)
16 # list.sort() сортирует список на месте (in-place)
17 fruits.sort()
18 # Проверяем
19 print(fruits)
20 # list.reverse() разворачивает порядок элементов в списке
21 fruits.reverse()
22 # Проверяем
23 print(fruits)
```

Run task3

```
C:\Users\Admin\PycharmProjects\HW8\.venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\HW8\task3.py
['apple', 'banana', 'orange']
['grape', 'apple', 'banana', 'orange']
['grape', 'apple', 'orange']
['apple', 'grape', 'orange']
['orange', 'grape', 'apple']
```

Упражнение 4.



```
1 # Дано:
2 words = ["hello", "world", "python", "code"]
3
4 # Создадим список длин слов
5 len(i) длина текущего слова
6 # for i in words проходим по каждому слову
7 word_lengths = [len(i) for i in words]
8 # Выводим результат
9 print(word_lengths)
10 # отфильтруем слова длиннее 4 символов if len(i) > 4
11 # for i in words проходим по каждому слову
12 # и добавляем i-е слово, если оно соответствует условию
13 long_words = [i for i in words if len(i) > 4]
14 # Проверим
15 print(long_words)
16 # Создадим словарь с помощью dict comprehension
17 # Словарь = {ключ: значение}
18 word_dict = {i: len(i) for i in words}
19 # Выведем результат
20 print(word_dict)
```

Run task4

```
C:\Users\Admin\PycharmProjects\HW8\.venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\HW8\task4.py
[5, 5, 6, 4]
['hello', 'world', 'python']
{'hello': 5, 'world': 5, 'python': 6, 'code': 4}
```

Упражнение 5.

```
4 # Решим ее с помощью enumerate
5 # встроенной функции Python, которая позволяет перебирать элементы
6 # и одновременно получать их индексы.
7 # Вводные данные:
8 nums = [3, 2, 4]
9 target = 6
10
11 # Создадим словарь, который хранит текущие числа и индексы
12 seen_nums = {}
13 # i-индекс, num-значение
14 for i, num in enumerate(nums):
15     # число, которое ищем, это разность, между target и текущим числом
16     pair_num = target - num
17     # и если такое число найдено в словаре
18     if pair_num in seen_nums:
19         # Выводим результат
20         print([seen_nums[pair_num], i])
21         break
22     # Сохраняем текущее число и его индекс в словарь
23     # Где ключ - это число, а индекс - это значение
24     seen_nums[num] = i
```

Run task5

C:\Users\Admin\PycharmProjects\HW8\.venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\HW8\task5.py
[1, 2]

Process finished with exit code 0

Часть 2: Функции и ООП

Упражнение 1.

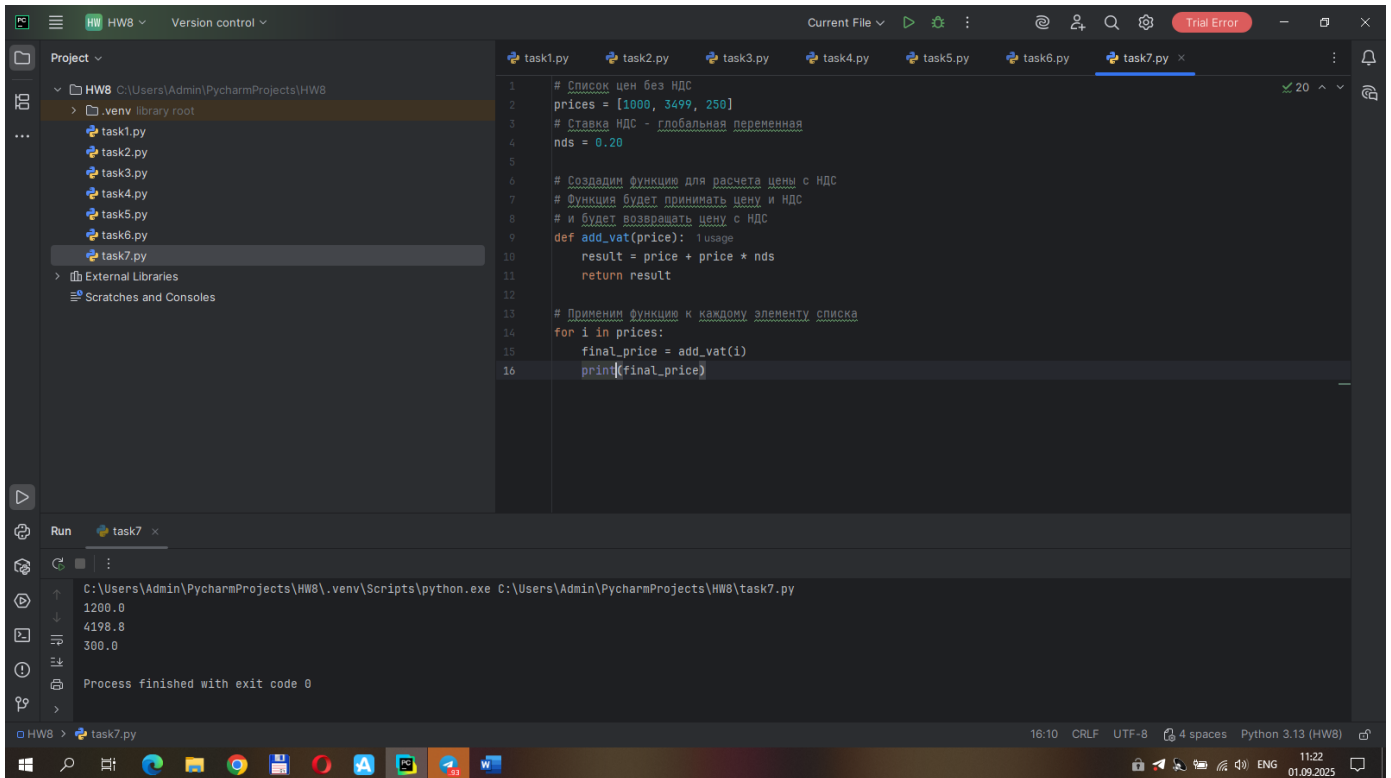
```
1 # Импортируем модуль для работы с датой и временем
2 import datetime
3
4 # Создаем функцию без параметров
5 def show_current_time():
6     # Создадим переменную now, в которую сохраняем объект
7     # с текущей датой и временем
8     # первый datetime - название модуля,
9     # второй datetime - название класса, now - метод
10     now = datetime.datetime.now()
11     # И выводим результат
12     print(now)
13
14 # вызов функции
15 show_current_time()
```

Run task6

C:\Users\Admin\PycharmProjects\HW8\.venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\HW8\task6.py
2025-09-01 10:53:50.092552

Process finished with exit code 0

Упражнение 2.



```
1 # Список цен без НДС
2 prices = [1000, 3499, 250]
3 # Ставка НДС - глобальная переменная
4 nds = 0.20
5
6 # Создадим функцию для расчета цены с НДС
7 # Функция будет принимать цену и НДС
8 # и будет возвращать цену с НДС
9 def add_vat(price): 1 usage
10     result = price + price * nds
11     return result
12
13 # Применим функцию к каждому элементу списка
14 for i in prices:
15     final_price = add_vat(i)
16     print(final_price)
```

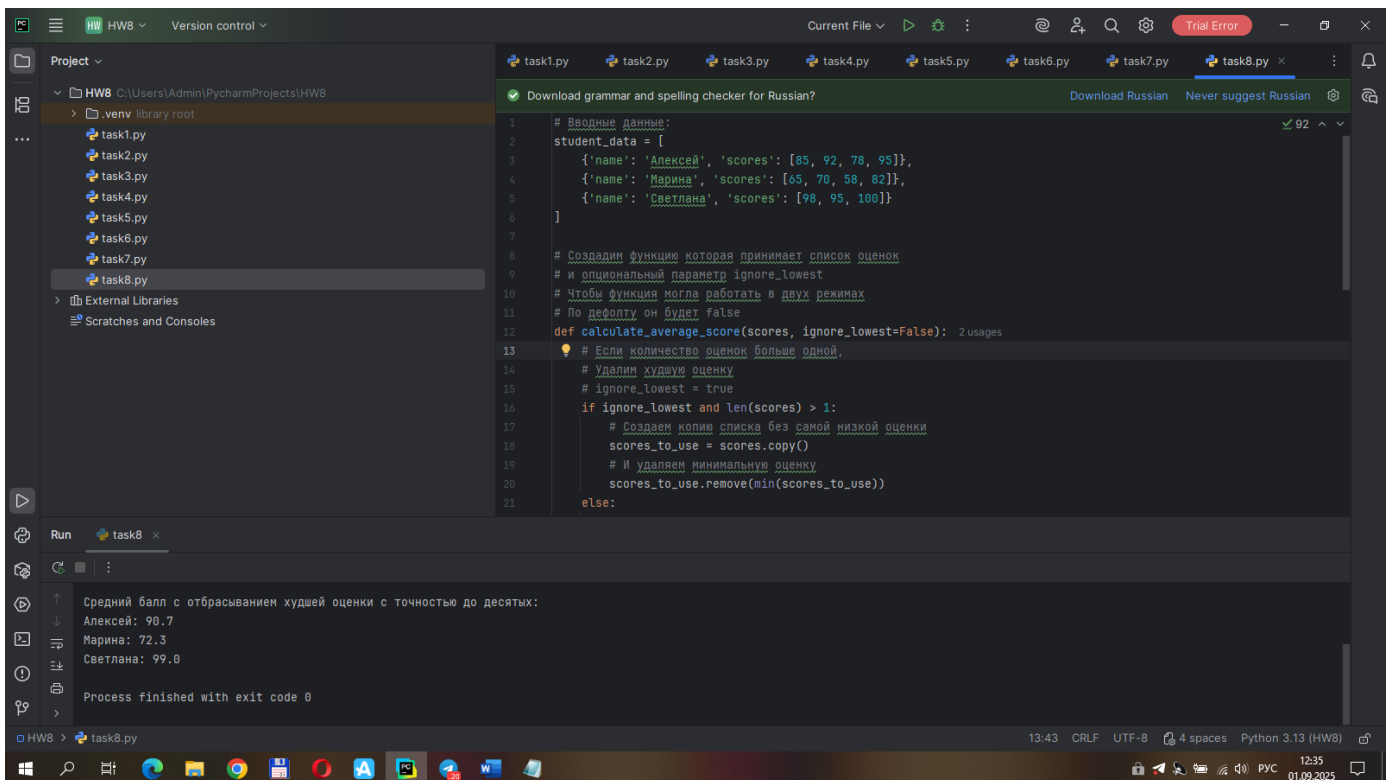
Run task7.py

C:\Users\Admin\PycharmProjects\HW8\.venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\HW8\task7.py

1200.0
4198.8
300.0

Process finished with exit code 0

Упражнение 3.



```
1 # Вводные данные:
2 student_data = [
3     {'name': 'Алексей', 'scores': [85, 92, 78, 95]},
4     {'name': 'Марина', 'scores': [65, 70, 58, 82]},
5     {'name': 'Светлана', 'scores': [98, 95, 100]}
6 ]
7
8 # Создадим функцию которая принимает список оценок
9 # и опциональный параметр ignore_lowest
10 # Чтобы функция могла работать в двух режимах
11 # По дефолту он будет false
12 def calculate_average_score(scores, ignore_lowest=False): 2 usages
13     # Если количество оценок больше одной,
14     # Удалим худшую оценку
15     # ignore_lowest = true
16     if ignore_lowest and len(scores) > 1:
17         # Создаем копию списка без самой низкой оценки
18         scores_to_use = scores.copy()
19         # И удаляем минимальную оценку
20         scores_to_use.remove(min(scores_to_use))
21     else:
```

Run task8.py

Средний балл с отбрасыванием худшей оценки с точностью до десятых:

Алексей: 90.7
Марина: 72.3
Светлана: 99.0

Process finished with exit code 0