HW 8: Строки, Списки, Словари. Функции. Введение в ООП

Часть 1: Строки, Списки, Словари

Упражнение 1: Операции со строкой

Программа:

```python
# Упражнение 1: Операции со строкой

# Дана строка
text = "Python Programming"

# 1. Вывод длины строки
print(f"Длина строки: {len(text)}")

# 2. Вывод символа по индексу 7
print(f"Вывод 7 символа: {text[7]}")

# 3. Вывод последних 3 символов
print(f"Вывод последних 3 символов: {text[-3:]}")

# 4. Проверка наличия подстроки "gram"
if "gram" in text:
    print("Подстрока 'gram' существует в тексте")
else:
    print("Подстрока 'gram' не существует в тексте")

```

Результат:

```
Длина строки: 18
Вывод 7 символа: P
Вывод последних 3 символов: ing
Подстрока 'gram' существует в тексте
```

Упражнение 2: Методы строк и форматинг

Программа:

```python
# Упражнение 2: Методы строк и форматинг

# Дано
email = " USER@DOMAIN.COM "

# 1. Очищаем и форматируем до вида: "user@domain.com"
clean_email = email.strip().lower()
print(f"Email после очистки и форматирования: {clean_email}")

# 2. Разделяем на имя пользователя и домен
username, domain = clean_email.split("@")
print(f"Имя пользователя: {username}")
print(f"Домен: {domain}")

# 3. Создаем, через f-строку: "Username: user, Domain: domain.com"
print(f"Username: {username}, Domain: {domain}")

```

Результат:

```
Email после очистки и форматирования: user@domain.com
Имя пользователя: user
Домен: domain.com
Username: user, Domain: domain.com
```

Упражнение 3: Методы списка

Программа:

```python
# Упаржнение 3: Методы списка

# Исходный список
fruits = ["apple", "banana"]

# 1. Добавляем "orange" в конец списка
fruits.append("orange")
print(f"Список с добавлением 'orange': {fruits}")

# 2. Вставка "grape" по индексу 1
fruits.insert(1, "grape")
print(f"Список со вставкой 'grape' по индексу 1: {fruits}")

# 3. Удаляем "banana"
fruits.remove("banana")
print(f"Список без 'banana': {fruits}")

# 4. Сортировка списка
fruits.sort()
print(f"Список после сортировки: {fruits}")

# 5. Переворачиваем список
fruits.reverse()
print(f"Перевернутый список: {fruits}")

```

Результат:

```
Список с добавлением 'orange': ['apple', 'banana', 'orange']
Список со вставкой 'grape' по индексу 1: ['apple', 'grape', 'banana', 'orange']
Список без 'banana': ['apple', 'grape', 'orange']
Список после сортировки: ['apple', 'grape', 'orange']
Перевернутый список: ['orange', 'grape', 'apple']
```

Упражнение 4: List comprehension и словари

Программа:

```python
# Упражнение 4: List comprehension и словари

# Дано
words = ["hello", "world", "python", "code"]

# 1. Создаем список длин слов, используя списковое включение
lengths_words = [len(word) for word in words]
print(f"Список длин слов: {lengths_words}")

# 2. Создаем список слов длиннее 4 символов
long_words = [word for word in words if len(word) > 4]
print(f"Список слов длиннее 4 символов: {long_words}")

# 3. Создаем словарь: {слово: длина} для всех слов
dict_word = {word: len(word) for word in words}
print(f"Словарь слов и из длин: {dict_word}")

```

Результат:

```
Список длин слов: [5, 5, 6, 4]
Список слов длиннее 4 символов: ['hello', 'world', 'python']
Словарь слов и из длин: {'hello': 5, 'world': 5, 'python': 6, 'code': 4}
```

Упражнения 5 (Опционально):

Программа:

```python
# Упражнение 5 (Опционально):
# Дан список чисел nums и целевое число target
# Найти индексы двух чисел, сумма которых равна target

print("Программа ищет индексы двух чисел, сумма которых равна target")
print()

# Пример 1
print("Пример 1 с target 9 и исходным списком: [2, 7, 11, 15]")
nums_first = [2, 7, 11, 15]
target = 9

for i in range(len(nums_first)):
    for j in range(i + 1, len(nums_first)):
        if nums_first[i] + nums_first[j] == target:
            print(f"Ответ: [{i}, {j}]")

print()

# Пример 2
print("Пример 2 с target 6 и исходным списком: [3, 2, 4] ")
nums_second = [3, 2, 4]
target = 6

for i in range(len(nums_second)):
    for j in range(i + 1, len(nums_second)):
        if nums_second[i] + nums_second[j] == target:
            print(f"Ответ: [{i}, {j}]")

```

Результат:

```
Программа ищет индексы двух чисел, сумма которых равна target

Пример 1 с target 9 и исходным списком: [2, 7, 11, 15]
Ответ: [0, 1]

Пример 2 с target 6 и исходным списком: [3, 2, 4]
Ответ: [1, 2]
```

Часть 2: Функции и ООП

Упражнение 1: Функции без параметров

Программа:

```python
# Упражнение 1: Функции без параметров

from datetime import datetime

def show_current_time():
    current_time = datetime.now()
    print(f"Текущие дата и время: {current_time}")

show_current_time()

```

Результат:

```
Текущие дата и время: 2025-08-31 01:26:38.807820
```

Упражнение 2: Функции с параметрами

Программа:

```python
# Упражнение 2: Функции с параметрами

prices = [1000, 3499, 250]
nds = 0.20

def add_vat(price, rate):
    return price + price * rate

for price in prices:
    final_price = add_vat(price, nds)
    print(f"Цена с НДС: {final_price}")
```

Результат:

```
Цена с НДС: 1200.0
Цена с НДС: 4198.8
Цена с НДС: 300.0
```

Упражнение 3 (Опционально)

- Создайте функцию calculate_average_score(), которая будет вычислять средний балл.
- Функция должна принимать список оценок scores как обязательный аргумент.
- Добавьте опциональный булевый параметр ignore_lowest со значением по умолчанию False.
- Если ignore_lowest равен True, функция должна отбросить наименьшую оценку перед вычислением среднего. Если в списке всего одна оценка, отбрасывать её не нужно.
- Используя цикл, пройдитесь по списку student_data. 2 раза, первый раз учитывая все оценки, а второй раз отбросив худшие оценки

Программа:

```python
# Упражнение 3 (Опционально)

def calculate_average_score(scores, ignor_lowes = False):
    if ignor_lowes and len(scores) > 1:
        scores = sorted(scores)[1:]
    return sum(scores) / len(scores)

student_data = [
    {'name': 'Алексей', 'scores': [85, 92, 78, 95]},
    {'name': 'Марина', 'scores': [65, 70, 58, 82]},
    {'name': 'Светлана', 'scores': [98, 95, 100]}
]

# 1. Без удаления худших оценок
print("Средние баллы для всех оценок:")
for student in student_data:
    avg = calculate_average_score(student['scores'])
    print(f"{student['name']}: {round(avg, 2)}")

print()

# 2. С удалением худщих оценок
print("Средние баллы без худших оценок:")
for student in student_data:
    avg = calculate_average_score(student['scores'], ignor_lowes = True)
    print(f"{student['name']}: {round(avg, 2)}")

```

Результат:

```
Средние баллы для всех оценок:
Алексей: 87.5
Марина: 68.75
Светлана: 97.67

Средние баллы без худших оценок:
Алексей: 90.67
Марина: 72.33
Светлана: 99.0
```