ПЗ №6 Работа со списками в Python

1. Тема работы

Работа со списками в Python. Создание, модификация и обработка списков. Использование методов списков.

2. Цель работы

Изучить основы работы со списками в Python, освоить методы создания и модификации списков, научиться применять встроенные методы для обработки списков.

3. Задачи работы

- Научиться создавать списки различными способами
- Освоить основные методы работы со списками
- Изучить способы обработки и фильтрации списков
- Написать 4 программы для работы со списками согласно индивидуальному варианту

4. Необходимое оборудование и ПО

- Компьютер с ОС Windows 10
- Установленные: VS Code, Python
- Установленные расширения VS Code: Python, Python Debugger

---

5. Ход работы

Часть 1: Повторение и подготовка

1. **Откройте вашу папку проекта** в VS Code (например, `Иванов_СА-501`)
2. **Создайте новую папку** `Practice_6` внутри вашей основной папки
3. **Создайте файлы** для заданий: `list1.py`, `list2.py`, `list3.py`, `list4.py`

Часть 2: Разбор примера программы

Пример 1: Базовые операции со списками

```
# Демонстрация основных операций со списками
```

```
# Создание списков
numbers = [1, 2, 3, 4, 5]
fruits = ['яблоко', 'банан', 'апельсин']
mixed = [1, 'текст', 3.14, True]

print("Исходные списки:")
print(f"numbers: {numbers}")
print(f"fruits: {fruits}")
print(f"mixed: {mixed}")

# Добавление элементов
numbers.append(6)  # добавление в конец
fruits.insert(1, 'киви')  # вставка по индексу

print("\nПосле добавления элементов:")
print(f"numbers: {numbers}")
print(f"fruits: {fruits}")

# Удаление элементов
removed = fruits.pop(2)  # удаление по индексу
numbers.remove(3)  # удаление по значению

print(f"\nУдалён элемент: {removed}")
print(f"fruits после удаления: {fruits}")
print(f"numbers после удаления: {numbers}")

# Срезы
print(f"\nСрезы numbers:")
print(f"первые 3: {numbers[:3]}")
print(f"последние 2: {numbers[-2:]}")
print(f"с 2 по 4: {numbers[1:4]}")
```

Пример 2: Методы списков

```
# Работа с методами списков

numbers = [5, 2, 8, 1, 9, 3, 2, 7]

print(f"Исходный список: {numbers}")

# Сортировка
```

```
numbers.sort()
print(f"После сортировки: {numbers}")

# Поиск
index = numbers.index(7)
count = numbers.count(2)
print(f"Индекс числа 7: {index}")
print(f"Количество двоек: {count}")
```

Пошаговый разбор:

1. **Создание списков** - квадратные скобки `[]`
2. **Метод** `append()` - добавление элемента в конец
3. **Метод** `insert()` - вставка элемента по индексу
4. **Метод** `pop()` - удаление элемента по индексу
5. **Метод** `remove()` - удаление элемента по значению
6. **Срезы** - получение части списка `[start:end:step]`

Запуск программы:

- Сохраните файл (Ctrl+S)
- Запустите через Ctrl+F5 или в терминале:

  ```
  python list1.py
  ```

Часть 3: Выполнение задания по варианту

1. **Получите задание** у преподавателя согласно вашему варианту
2. **Реализуйте программы** в соответствующих файлах
3. **Протестируйте программы** на разных входных данных
4. **Убедитесь**, что работа со списками выполняется корректно

---

6. Варианты заданий

Вариант 1:

1. Напишите программу, которая создаёт список из 10 чисел введенных пользователем и находит максимальный и минимальный элементы.
2. Напишите программу, которая удаляет из списка, введенного пользователем, все повторяющиеся элементы.
3. Напишите программу, которая сортирует список строк по длине.

4. * Напишите программу, которая реализует простой стек (добавление, удаление, просмотр верхнего элемента).

Вариант 2:

1. Напишите программу, которая создаёт список чётных чисел от 2 до 20 и вычисляет их среднее арифметическое.
2. Напишите программу, которая объединяет два списка и удаляет дубликаты.
3. Напишите программу, которая находит все простые числа в заданном диапазоне.
4. * Напишите программу, которая реализует очередь (добавление в конец, удаление из начала).

Вариант 3:

1. Напишите программу, которая создаёт список квадратов чисел от 1 до 10 и выводит их в обратном порядке.
2. Напишите программу, которая фильтрует список чисел, введенных пользователем, оставляя только те, которые делятся на 3 или на 5.
3. Напишите программу, которая находит второй по величине элемент в списке, заданном пользователем.
4. * Напишите программу, которая реализует функцию шифрования Цезаря для списка символов.

Вариант 4:

1. Напишите программу, которая создаёт список из введённых пользователем чисел и вычисляет их сумму и произведение.
2. Напишите программу, которая разбивает список на два: с чётными и нечётными числами.
3. Напишите программу, которая находит наиболее часто встречающийся элемент в списке, заданным пользователем.
4. * Напишите программу, которая реализует очередь (добавление в конец, удаление из начала).

Вариант 5:

1. Напишите программу, которая создаёт список из первых 10 чисел Фибоначчи.
2. Напишите программу, которая сдвигает список на N элементов вправо.
3. Напишите программу, которая находит второй наименьший элемент в списке, заданном пользователем.
4. * Напишите программу, которая реализует бинарный поиск в отсортированном списке.

7. Отчёт по выполнению работы

Отчёт должен содержать:

1. **Цель работы**
2. **Задание по варианту** (текст всех 4 заданий)
3. **Листинги программ** (код всех 4 программ)
4. **Скриншоты выполнения программ** (результаты работы в терминале)
5. **Выводы**:
   - С какими трудностями столкнулись при работе со списками?
   - Какие методы списков оказались наиболее полезными?
   - В чём преимущество списковых включений?
   - Какая программа была самой сложной и почему?

---

8. Контрольные вопросы

1. Что такое список в Python и чем он отличается от других типов данных?
2. Какие способы создания списков вы знаете?
3. Как добавить элемент в конец списка? А в начало?
4. Чем отличаются методы `pop()`, `remove()` и `del`?
5. Что такое срезы списка и как они работают?
6. Как отсортировать список? Чем отличается `sort()` от `sorted()`?
7. Как создать многомерный список и как обратиться к его элементам?
8. Какие методы для поиска элементов в списке вы знаете?
9. Как объединить два списка?

---

9. Дополнительные задания

1. Реализуйте матричные операции (сложение, умножение) используя вложенные списки
2. Напишите программу для решения системы линейных уравнений методом Гаусса
3. Напишите программу, которая сжимает список с использованием RLE-кодирования

---

10. Полезные методы списков для справки

```
# Основные методы списков
list.append(x)      # Добавить элемент в конец
list.insert(i, x)   # Вставить элемент по индексу
list.remove(x)      # Удалить первый элемент равный x
list.pop([i])       # Удалить элемент по индексу (или последний)
list.sort()         # Отсортировать список
list.reverse()      # Развернуть список
list.index(x)       # Индекс первого вхождения x
list.count(x)       # Количество вхождений x
list.copy()         # Поверхностная копия списка
```