ПМ3 Разработка модулей ПО.

РО 3.1 Понимать и применять принципы объектно-ориентированного и асинхронного программирования.

Тема 6. Коллекции данных. Списки. Методы списков

Цель занятия:

Познакомиться с коллекциями данных в Python, подробно изучить структуру данных «список» (list), её свойства, возможности и основные методы для работы с элементами.

Учебные вопросы:

1. Понятие коллекций данных в Python.
2. Списки (list). Основы.
3. Основные методы списков.

1. Понятие коллекций данных в Python.

Коллекция данных — это структура, которая позволяет хранить несколько значений в одной переменной.

В отличие от обычных переменных, содержащих одно значение, коллекции позволяют работать с наборами элементов:

- хранить списки объектов;
- перебирать элементы;
- изменять отдельные части коллекции;
- сортировать, искать, фильтровать данные.

Виды коллекций в Python

Python предлагает разные типы коллекций, которые различаются по важным характеристикам.

Изменяемые и неизменяемые

Тип	Описание	Примеры
Изменяемые (mutable)	можно менять элементы	list, dict, set
Неизменяемые (immutable)	нельзя менять элементы после создания	tuple, str

Упорядоченные и неупорядоченные

Тип	Порядок	Пример
Упорядоченные	элементы имеют строгий порядок, индексацию	list, tuple, str
Неупорядоченные	порядок хранения не гарантирован	set, dict

2. Списки (list). Основы.

Список (**list**) — это упорядоченная изменяемая коллекция элементов в Python.

Основные свойства:

- Упорядоченность — элементы имеют фиксированный порядок.
- Изменяемость — элементы списка можно менять, удалять, добавлять.
- Гетерогенность — список может содержать объекты разных типов (числа, строки, другие списки).

Списки являются одной из самых используемых структур данных в Python.

Обозначение списков

Литерал списка — квадратные скобки

```
numbers = [1, 2, 3]
data = ["Alice", 25, True]
```

Создание списка с помощью функции **list()**

Используется для преобразования объектов в список:

```
chars = list("Hello")  # ['H', 'e', 'l', 'l', 'o']
nums = list(range(5))  # [0, 1, 2, 3, 4]
```

Пустой список:

```
empty = []
empty2 = list()
```

Хранение элементов разных типов

В одном списке могут находиться элементы любых типов:

```
mixed = [10, "text", 3.14, False, [1, 2, 3]]
```

Python не навязывает типизацию элементов списка.

Вложенные списки

Список может содержать другие списки → двумерные или многомерные структуры.

Пример двумерного списка (матрица 3×3):

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

Доступ к элементу:

```
print(matrix[1][2])  # 6
```

Длина списка: **len()**

Функция **len()** возвращает количество элементов в списке:

```
items = ["apple", "banana", "cherry"]
print(len(items))  # 3
```

Индексация

Списки поддерживают доступ по индексу. Последовательность элементов в списке начинается с 0, как у символов в строке.

0	1	2	3
‘Петров’	‘Николай’	‘Иванович’	25

Список “Данные пользователя”

```
a = [10, 20, 30]
print(a[0])   # 10
print(a[-1])  # 30 (последний элемент)
```

Срезы

Позволяют получать подсписки:

```
numbers = [0, 1, 2, 3, 4, 5]
print(numbers[1:4])    # [1, 2, 3]
print(numbers[:3])     # [0, 1, 2]
print(numbers[::2])    # [0, 2, 4]
```

Срезы создают новый список, а не изменяют исходный.

Изменяемость списков

Список можно менять «на месте»:

```
a = [1, 2, 3]
a[1] = 10       # замена элемента
print(a)        # [1, 10, 3]
```

Можно изменять части списка срезами:

```
a[0:2] = [100, 200]
```

Добавление и удаление (подробнее в следующем вопросе — о методах списков):

3. Основные методы списков.

Списки в Python обладают большим набором встроенных методов, позволяющих удобно изменять, расширять и обрабатывать данные. Ниже приведены наиболее важные и часто используемые методы.

Добавление элементов

append(x) — добавить элемент в конец списка.

Добавляет один элемент как единое целое:

```
a = [1, 2, 3]
a.append(4)    # [1, 2, 3, 4]
```

extend(iterable) — расширить список элементами другой коллекции.

Добавляет каждый элемент переданного объекта:

```
a = [1, 2]
a.extend([3, 4]) # [1, 2, 3, 4]
```

insert(i, x) — вставить элемент на позицию **i**.

Элементы сдвигаются вправо:

```
a = [10, 20, 30]
a.insert(1, "hello") # [10, 'hello', 20, 30]
```

Удаление элементов

remove(x) — удалить первое вхождение элемента.

Если элемент не найден → ошибка:

```
a = [1, 2, 3, 2]
a.remove(2)          # [1, 3, 2]
```

pop(i) — удалить элемент по индексу и вернуть его

Если индекс не указан → удаляет последний:

```
a = [1, 2, 3]
x = a.pop(1)    # возвращает 2
# список → [1, 3]
```

clear() — очищает список полностью.

```
a = [1, 2, 3]
a.clear()       # []
```

Методы поиска и работы с содержимым.

index(x) — индекс первого вхождения

```
a = ["a", "b", "c", "b"]
a.index("b")    # 1
```

count(x) — сколько раз элемент встречается

```
a = [1, 2, 2, 3, 2]
a.count(2)    # 3
```

Методы сортировки.

sort() — сортирует список на месте

```
a = [3, 1, 2]
a.sort()          # [1, 2, 3]
```

Можно указывать параметры:

a.sort(reverse=True) # сортировка по убыванию

sorted(iterable) — возвращает новый отсортированный список

Исходный список не изменяется:

```
a = [3, 1, 2]
b = sorted(a)
# a → [3, 1, 2]
# b → [1, 2, 3]
```

Разворот списка

reverse() — разворачивает список на месте

```
a = [1, 2, 3]
a.reverse()
# [3, 2, 1]
```

reversed(list) — возвращает итератор, исходный список не изменяется.

Нужно преобразовывать в список:

```
a = [1, 2, 3]
b = list(reversed(a))
# [3, 2, 1]
```

Копирование списка

copy() — неглубокая копия, создаёт новый список, но не копирует вложенные объекты.

```
a = [1, 2, 3]
b = a.copy()
```

Аналог через срез:

```
b = a[:]
```

Полезные операции вне методов.

Проверка элемента: in

```
3 in [1, 2, 3]   # True
```

Длина списка: len()

```
len([10, 20, 30])  # 3
```

Минимум и максимум: min(), max()

```
min([3, 1, 2])  # 1
max([3, 1, 2])  # 3
```

Сумма числовых элементов: sum()

```
sum([1, 2, 3])  # 6
```

Контрольные вопросы:

- Чем коллекция отличается от обычной переменной?
- Какие коллекции есть в Python?
- Чем список отличается от кортежа?
- Что означает «список — изменяемый тип данных»?
- Как работают индексы и срезы в списках?
- Назовите методы добавления элементов в список.
- Назовите методы удаления элементов.
- Для чего используются sort() и reverse()?
- Чем отличаются append() и extend()?
- Как перебрать список с индексами?

Домашнее задание:

https://ru.hexlet.io/courses/js-asynchronous-programming

Материалы лекций:

https://github.com/ShViktor72/Education

Обратная связь:

colledge20education23@gmail.com