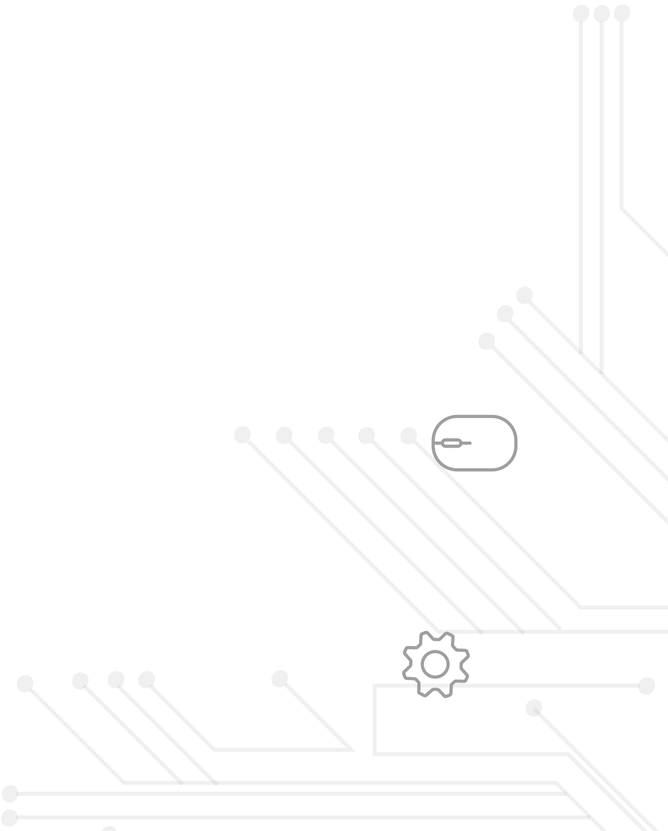
УНИВЕРСИТЕТ СИНЕРГИЯ

ПОВТОРЕНИЕ. СОРТИРОВКА

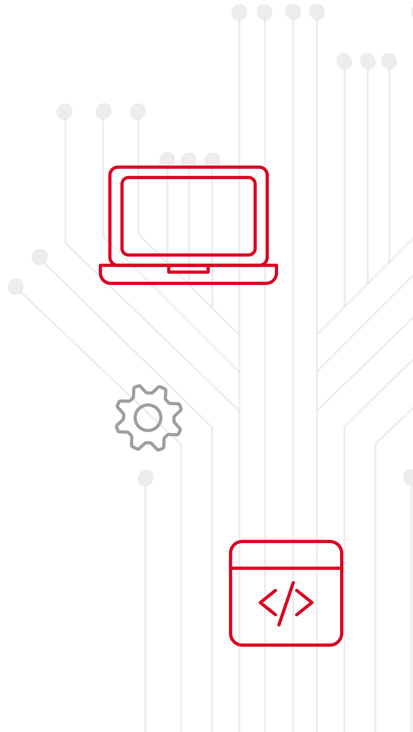
Введение

Сортировка является одним из базовых элементов построения многих алгоритмов и связана с захватывающими идеями. Понимание того, как алгоритмы сортировки работают в Python, является важным шагом при создании правильных и эффективных алгоритмов, которые решают реальные проблемы. Сортировка широко используется во многих областях, включая базы данных, поиск и машинное обучение.



Основные алгоритмы сортировки

- Пузырьковая сортировка (Bubble Sort)
- Сортировка выбором (Selection Sort)
- Сортировка вставками (Insertion Sort)
- Сортировка слиянием (Merge Sort)
- Быстрая сортировка (Quick Sort)



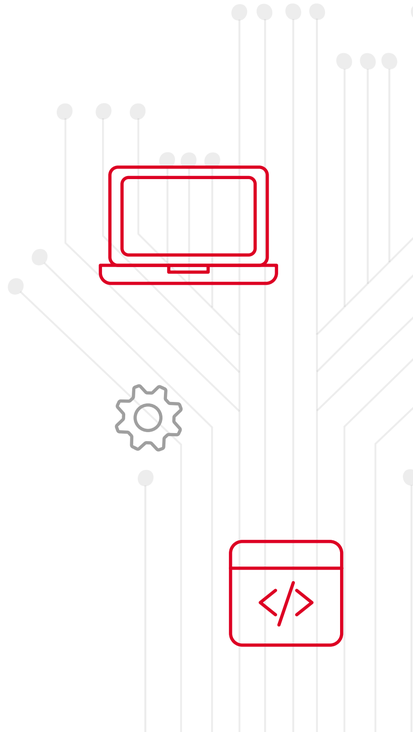
Сортировка встроенными методами в Python

Функция **sorted()** возвращает отсортированный список из итерируемого объекта. Она не изменяет исходный список, а возвращает новый отсортированный список.

```
arr = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5]
sorted_arr = sorted(arr)
```

Метод **sort()** сортирует список на месте, то есть изменяет исходный список.

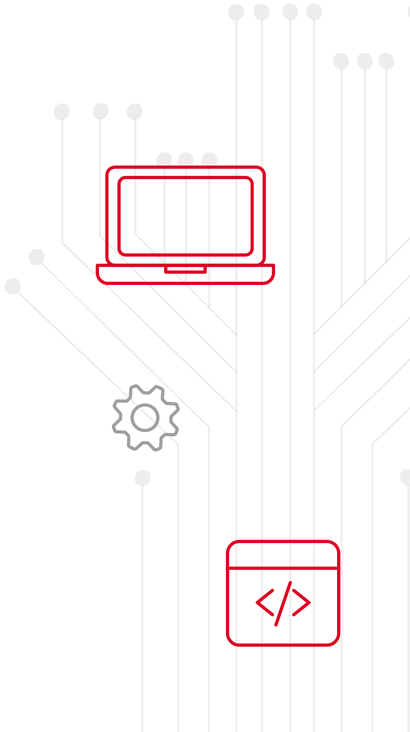
```
arr = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5]
arr.sort()
```



Сортировка словарей

Словари в Python не упорядочены, но можно отсортировать их по ключам или значениям. Для сортировки словаря по ключам можно использовать функцию sorted() или метод items().

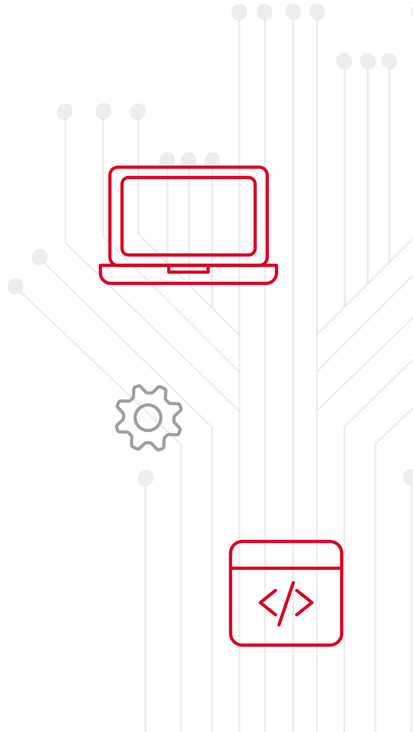
```
# Сортировка словаря по ключам
d = {'apple': 10, 'orange': 20, 'banana': 5, 'peach': 15}
sorted_d = dict(sorted(d.items()))

# Сортировка словаря по значениям
sorted_d = dict(sorted(d.items(), key=lambda x: x[1]))
```



Сортировка множеств

Множества в Python также можно сортировать. Для этого можно использовать функцию **sorted()**.

```
# Сортировка множества
s = {3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5}
sorted_s = sorted(s)
```



Подведем итоги

- У разных алгоритмов сортировки свои плюсы и минусы
- Коллекции можно сортировать с помощью sort(), sorted()