Модуль heapq, кучи в Python

Алгоритм очереди кучи

Модуль heapq обеспечивает реализацию алгоритма очереди кучи, также известного как алгоритм очереди приоритетов.

Кучи - это двоичные деревья, для которых каждый родительский узел имеет значение, меньшее или равное любому из его дочерних элементов. В этой реализации используются массивы, для которых heap[k] <= heap[2*k+1] и heap[k] <= heap[2*k+2] для всех k, считая элементы с нуля. Для сравнения, несуществующие элементы считаются бесконечными. Интересным свойством кучи является то, что ее наименьшим элементом всегда является корень heap[0].

Приведенный ниже API отличается от алгоритмов кучи, описанных в учебниках в двух аспектах:

1. Модуль heapq использует индексацию с нуля. Это делает связь между индексом для узла и индексами для его дочерних элементов несколько менее очевидной, но является более подходящей, поскольку Python использует индексацию с нуля.
2. Метод pop() модуля heapq возвращает наименьший элемент, а не самый большой. В учебниках он называется min heap. Элемент max heap чаще встречается в учебниках из-за его пригодности для сортировки на месте.

Эти два аспекта позволяют просматривать кучу как обычный список Python без сюрпризов: heap[0] - самый маленький элемент, а heap.sort() поддерживает инвариант кучи!

Чтобы создать кучу, используйте инициализацию списка [], или можно преобразовать заполненный список в кучу с помощью функции heapq.heapify().

Функция heappush() модуля heapq в Python

Функция `heappush()` модуля `heapq` добавляет значение элемента в кучу, сохраняя инвариант кучи.

Функция heappop() модуля heapq в Python

Функция `heappop()` модуля `heapq` возвращает и удаляет наименьший элемент из кучи `heap`, сохраняя инвариант кучи.

Функция heappushpop() модуля heapq в Python

Функция heappushpop() модуля heapq добавляет значение элемента item в кучу heap, затем возвращает и удаляет самый маленький элемент из кучи heap.

Функция heapify() модуля heapq в Python

Функция `heapify()` модуля `heapq` преобразовывает список `x` в кучу на месте за линейное время.

Функция heapreplace() модуля heapq в Python

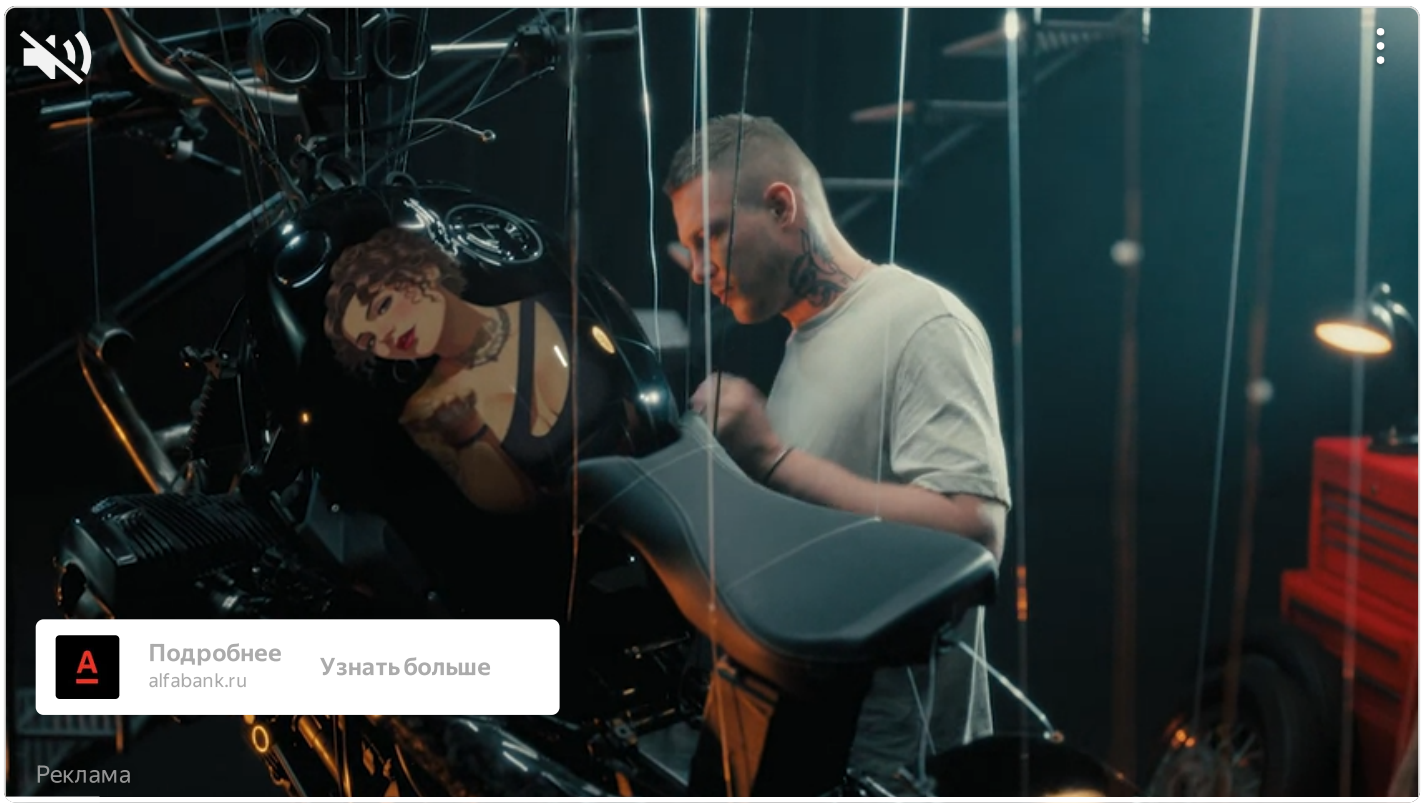
Функция `heapreplace()` модуля `heapq` сначала удаляет и возвращает наименьший элемент из кучи `heap`, а потом добавляет новый элемент `item`. Размер кучи `heap` не меняется. Если куча пуста, поднимается исключение `IndexError`.

Функция merge() модуля heapq в Python

Функция `merge()` модуля `heapq` объединяет несколько отсортированных последовательностей `*iterables` в один отсортированный итератор. Например, объединить записи с метками времени из нескольких файлов журнала.

Функция nlargest() модуля heapq в Python

Функция `nlargest()` модуля `heapq` возвращает список с `n` самыми большими элементами из набора данных, определенного с помощью итерации `iterable`.



Функция nsmallest() модуля heapq в Python

Функция `nsmallest()` модуля `heapq` возвращает список с `n` наименьшими элементами из набора данных, определенного с помощью итерируемой последовательности `iterable`.

Примеры использования heapq

Примеры: назначений приоритетов задач вместе с основной задачей, которую нужно выполнить и пирамидальная сортировка.

Реализация очереди приоритетов

Реализация очереди приоритетов.

Вверх