Очередь с приоритетом: инвариант минимальной кучи

1. После heapq.heapify(a) для непустого списка что находится в a[0]?

1. Максимальный элемент
2. Последний добавленный обязательно
3. Медиана обязательно
4. Минимальный элемент

Правильный ответ: Минимальный элемент

2. Гарантирует ли heapify полную сортировку списка по возрастанию?

1. Да
2. Нет
3. Только для целых чисел всегда
4. Только для строк всегда

Правильный ответ: Нет

3. Что делает heapq.heappop(h) с непустой минимальной кучей?

1. Возвращает максимум без удаления
2. Удаляет и возвращает минимум
3. Удаляет случайный элемент
4. Только сортирует список

Правильный ответ: Удаляет и возвращает минимум

4. В минимальную кучу со значениями 4,7,9 добавили heappush(h,2). Новый корень?

1. 7
2. 2
3. 4
4. 9

Правильный ответ: 2

5. Из кучи со значениями 5,1,3 последовательно извлекли все элементы через heappop. Порядок?

1. 1,3,5
2. 3,1,5
3. 5,1,3
4. 5,3,1

Правильный ответ: 1,3,5

6. Каков обычный способ представления минимальной кучи в heapq?

1. Список Python
2. Только словарь
3. Только множество
4. Отдельная SQL-таблица обязательно

Правильный ответ: Список Python

7. Какое исключение вызовет heappop([])?

1. IndexError
2. ValueError
3. KeyError
4. Не будет ошибки, вернётся 0

Правильный ответ: IndexError

8. Что вернёт heapq.nsmallest(2,[8,2,6,1])?

1. [2,1,6,8]
2. [8,6]
3. [2,6]
4. [1,2]

Правильный ответ: [1,2]

9. Задания хранятся как (приоритет,порядковый_номер,объект). Зачем уникальный номер при одинаковом приоритете?

1. Чтобы удалить все дубли автоматически
2. Чтобы увеличить приоритет всех заданий
3. Чтобы очередь стала FIFO независимо от приоритета
4. Чтобы разрешить ничью без сравнения несравнимых объектов

Правильный ответ: Чтобы разрешить ничью без сравнения несравнимых объектов

10. Приоритет элемента внутри кучи изменили прямым присваиванием без восстановления. Что может случиться?

1. Список обязательно очистится
2. Все значения станут строками
3. Нарушится инвариант кучи
4. Куча всегда сама исправится

Правильный ответ: Нарушится инвариант кучи