Data Structures and Algorithms

HUS HKI, 23 - 24

Assignment 6

Lecturer: Nguyễn Thị Hồng Minh
Trần Bá Tuấn - Đặng Trung Du

§ Priority Queue and Heap §

Phần 1: THÔNG TIN CHUNG

(1) Mục tiêu

- Nắm được các khái niệm, kiến thức liên quan đến cấu trúc dữ liệu hàng đợi ưu tiên và đống.
- Sinh viên cần phân biệt được sự khác nhau giữa hàng đợi ưu tiên và hàng đợi bình thường.
- Sinh viên cài đặt được hàng đợi ưu tiên bằng mảng và cấu trúc liên kết, cài đặt đống bằng mảng.
- Nắm rõ thuật toán sắp xếp vun đống và sử dụng đống biểu diễn hàng đợi ưu tiên.

(2) Tài liệu đọc thêm

- Hàng đợi ưu tiên - https://www.geeksforgeeks.org/priority-queue-set-1-introduction/
- Heap - https://www.geeksforgeeks.org/heap-data-structure/
- Thuật toán sắp xếp vun đống - https://www.geeksforgeeks.org/heap-sort/

(3) Quy cách nộp bài

- Mỗi sinh viên hoàn thành bài tập trong package có tên Hw6_<idSinhvien>_<Hovaten>
- Trong đó, <idSinhvien> là mã sinh viên.
- Sinh viên nộp bài làm trên tài khoản của mình bao gồm:
  1. File nén .zip của thư mục chứa package (Hw6_<idSinhvien>_<Hovaten>.zip)
  2. Tất cả các file nguồn *.java.
- Khi hết hạn nộp bài, các bài làm sẽ công bố để thực hiện chấm chéo bài tập.
- Sinh viên không nộp bài sẽ nhận điểm 0 bài tập tuần.
- Sinh viên CÓ GIAN LẬN trong nộp bài tập sẽ bị ĐÌNH CHỈ môn học (điểm 0 cho tất cả các điểm thành phần).

Phần 2: Thực hành

(1) Bài tập

CHÚ Ý

1. Lựa chọn các gói bài tập để thực hiện:
   - Combo 1: Bài 1, Bài 2, Bài 3 (cơ bản) được 80đ.
   - Combo 2: Bài 1, Bài 4, Bài 5 (nâng cao 1) được 85đ.
   - Combo 3: Bài 2, Bài 3, Bài 6 (nâng cao 2) được 90đ.
   - Combo 4: Bài 5, Bài 6 (nâng cao 3) được 100đ.
2. Trong bài nộp có file .doc hoặc .txt thuyết minh về gói bài tập thực hiện và những nội dung cần giải thích về bài làm: thuật toán được chọn, tài liệu tham khảo, định dạng input, yêu cầu hệ thống...

Bài tập 1. ***Tạo giao diện các phần tử và hàng đợi ưu tiên PriotityQueueInterface như sau:***

```
public interface Entry <K,E> {
    K getKey();
    E getValue();
}
public interface PriorityQueueInterface<K , E> {
    public int size();
    public boolean isEmpty();
    public void insert(Entry<K, E> entry);
    public void insert(K k, E e);
    public Entry<k, E> removeMin();
    public Entry<k, E> min();
}
```

a) ***Xây dựng kiểu dữ liệu UnsortedArrayPriorityQueue sử dụng mảng, cài đặt giao diện PriorityQueueInterface đã xây dựng ở trên với gợi ý như sau:***

```
public class UnsortedArrayPriorityQueue<K extends Comparable, E> implements
                                                        PriorityQueueInterface {
    protected class ArrEntry<K, E> implements Entry<K, E>{
        K key;
        E element;
        public ArrEntry (K k, E e){

        }
    }
    ArrEntry<K, E> [] array;
    int n = 0;
    int defaultsize = 1000;
}
```

b) ***Xây dựng kiểu dữ liệu SortedArrayPriorityQueue sử dụng mảng, cài đặt giao diện PriorityQueueInterface đã xây dựng ở trên với gợi ý như trong ý a)***

```
public class SortedArrayPriorityQueue<K extends Comparable, E> implements
                                                        PriorityQueueInterface {

}
```

c) ***Xây dựng kiểu dữ liệu UnsortedLinkedPriorityQueue sử dụng cấu trúc liên kết, cài đặt giao diện PriorityQueueInterface đã xây dựng ở trên với gợi ý như sau:***

```
public class UnsortedLinkedPriorityQueue<K extends Comparable, E> implements
                                                        PriorityQueueInterface {
    protected class NodeEntry<K, E> implements Entry<K, E>{
```

```
        private K key;
        private E element;
        private NodeEntry<K, E> next;
        public ArrEntry (K k, E e){

        }
    }
    private NodeEntry<K,E> head;
    private NodeEntry<K,E> tail;
    int n = 0;
}
```

d) ***Xây dựng kiểu dữ liệu SortedLinkPriorityQueue sử dụng cấu trúc liên kết, cài đặt giao diện PriorityQueueInterface đã xây dựng ở trên với gợi ý như trong ý c)***

```
public class SortedLinkedPriorityQueue<K extends Comparable, E> implements
                                                        PriorityQueueInterface {
}
```

e) ***Viết hàm test các kiểu dữ liệu PriorityQueue đã triển khai với:***

- *Danh sách các số nguyên, giá trị phần tử dùng làm khóa.*
- *Danh sách các đối tượng có khóa và giá trị khác nhau. Ví dụ: đối tượng hàng hóa bao gồm tên hàng hóa (giá trị), giá tiền (khóa).*

Bài tập 2. ***Xây dựng cấu trúc dữ liệu hàng đợi ưu tiên sử dụng heap cài đặt bằng mảng với lược đồ như sau:***

```
public class MinHeapPriorityQueue<K extends Comparable, E> extends
                                                        SortedArrayPriorityQueue {
    ArrEntry<K,E> heapPQ[];
    //Update methods
    protected void upHeap()
    protected void downHeap()
}
```

Bài tập 3.

- *Sử dụng cấu trúc dữ liệu HeapPriorityQueue để viết hàm HeapSort sắp xếp dãy số theo thứ tự tăng dần.*
- *Chạy và so sánh thời gian thực hiện của các thuật toán sắp xếp: BubbleSort, InsertionSort, SelectionSort, HeapSort, QuickSort và MergeSort.*
- *Vẽ biểu đồ thể hiện việc so sánh đó.*

Bài tập 4. *Sử dụng 4 kiểu dữ liệu PriorityQueue đã xây dựng ở bài 1 để thực hiện các yêu cầu sau:*

- *Lập danh sách n đối tượng (d, n). Để đơn giản đối tượng d có giá trị nguyên và sử dụng giá trị làm khóa.*
- *Thực hiện các thao tác thêm phần tử vào danh sách (insert), lấy ra phần tử nhỏ nhất (removeMin) với các danh sách có độ dài n khác nhau, với các kiểu PriorityQueue đã được cài đặt. Lập bảng so sánh thời gian thực hiện (milisecond) các thao tác có dạng như sau:*

Methods & PriorityQueue \ n		10^3	10^4	10^5	10^6	10^7	10^8
insert	UnsortedArray						
	SortedArray						
	UnsortedLinked						
	SortedLinked						
removeMin	UnsortedArray						
	SortedArray						
	UnsortedLinked						
	SortedLinked						

Bài tập 5. *Sử dụng kiểu PriorityQueue đã xây dựng ở bài 1 viết chương trình mô phỏng hệ thống điều hành không lưu đơn giản, quản lí các sự kiện máy bay cất cánh và hạ cánh tại sân bay. (Xem bài tập R-9.4 sách M.Goodrich, trang 395).*

Hệ thống được mô tả như sau: *Mỗi một máy bay đăng kí sự kiện cất cánh và hạ cánh với một nhãn thời gian, là thời gian mà sự kiện đó dự kiến sẽ xảy ra. Chương trình mô phỏng cần thực hiện một cách hiệu quả 2 chức năng sau:*

- *Thêm vào hệ thống một đăng kí sự kiện cất/hạ cánh, là sự kiện sẽ xảy ra với một nhãn thời gian xác định.*
- *Lấy ra sự kiện cất/hạ cánh chuẩn bị cho công tác điều hành, là sự kiện sắp xảy ra với nhãn thời gian nhỏ nhất.*

Bài tập 6. *Sử dụng HeapPriorityQueue ở bài tập 2 viết chương trình mô phỏng hệ thống điều khiển giao dịch chứng khoán đơn giản, quản lí các lệnh mua và bán. (Xem bài tập C-9.48, P-9.54 sách M.Goodrich, trang 399).*

(2) Bài tập thêm

Sinh viên tự tìm hiểu và luyện tập

Các link dưới đây có thể có rất nhiều bài tập.

- Heap - https://www.geeksforgeeks.org/tag/heap/
- Heap - https://practice.geeksforgeeks.org/heap
- Reorganize String - https://leetcode.com/problems/reorganize-string/