Lab 3 - Heap

> Zifeng Jiang
>
> 9 Oct 2019

Introduction

This lab is designed to help you get familiar with **heap** , a useful data structure . Your task in this lab is to implement a heap **individually** , and solve a problem with the heap .

Specification

Every day , cows of ***AD Farm*** go and eat grass on the grassland of ***CT Farm*** . In order not to be noticed , only **K** cows can eat grass at the same time . Each cow has its **arrival time** and **eating time** . The input stream consists of ordered pairs of **arrival time** and **eating time** for each cow, sorted by arrival time.

Cow data structure

To model a cow with arrival time and eating time, create a data type Cow with the following API:

```
public class Cow{
  //create a cow
  public Cow (int arrival, int eating);
  //get the arrival time
  public int getArrival();
  //get the eating time
  public int getEating();
  //set the arrival time
  public void setArrival(int arrival);
  //set the eating time
  public void setEating(int eating);
}
```

Binary Heap data structure

To model a binary heap, create a data type BinaryHeap with following API:

```
public class BinaryHeap{
  //create a BinaryHeap
  public BinaryHeap(int capacity);
  public BinaryHeap(int[] items);
  //insert a element into the heap
  public void insert(int x);
  //return the minimum element
  public int findMin() throws EmptyException;
  //remove and return the minimum element
  public int deleteMin() throws EmptyException;
  //judge whether the heap is empty
  public boolean isEmpty();
  //remove all elements
  public void makeEmpty();
}
```

Event Simulator data structure

To model an event simulator, create a data type EventSimulator with the following API:

```
public class EventSimulator{
  //create an event simulator
  //k determine how many cows can eat grass at the same time
  //arrivalLine is the input stream, sorted by arrival time
  public EventSimulator(int k, ArrayList<Cow> arrivalLine);
  //set the arrival line
  public void setArrivalLine(ArrayList<Cow> arrivalLine);
  //start simulation
  public void simulate();
}
```

You can use `java.util.Queue` when you need. But you should implement the binary heap data structure by your own.

Test case

The following method is an example to test the data type you implement.

```
public static void main(String args[]){
  ArrayList<Cow> arrivalLine = new ArrayList<Cow>();
  arrivalLine.add(new Cow(0, 8));
  arrivalLine.add(new Cow(1, 5));
  arrivalLine.add(new Cow(2, 3));
  arrivalLine.add(new Cow(4, 1));
  arrivalLine.add(new Cow(4, 5));
  arrivalLine.add(new Cow(4, 4));
  arrivalLine.add(new Cow(5, 4));
  arrivalLine.add(new Cow(6, 1));
  arrivalLine.add(new Cow(7, 3));
  EventSimulator es = new EventSimulator(3, arrivalLine);
  es.simulate();
}
```

The output is as follow:

```
0 o'clock: cow which arrived at 0 o'clock starts eating grass and will leave
at 8 o'clock
1 o'clock: cow which arrived at 1 o'clock starts eating grass and will leave
at 6 o'clock
2 o'clock: cow which arrived at 2 o'clock starts eating grass and will leave
at 5 o'clock
5 o'clock: cow which arrived at 4 o'clock starts eating grass and will leave
at 6 o'clock
6 o'clock: cow which arrived at 4 o'clock starts eating grass and will leave
at 11 o'clock
6 o'clock: cow which arrived at 4 o'clock starts eating grass and will leave
at 10 o'clock
8 o'clock: cow which arrived at 5 o'clock starts eating grass and will leave
at 12 o'clock
10 o'clock: cow which arrived at 6 o'clock starts eating grass and will leave
at 11 o'clock
11 o'clock: cow which arrival at 7 o'clock starts eating grass and will leave
at 14 o'clock
```

Submission

Create a zip file named **YourStudentID.zip** that contains your code project and upload your zip file to FTP server.

Deadline

9 Oct 2019 24:00 **TODAY**