Lab 1 Report Template

1. **GivImplement algorithms with MinHeap for job scheduling, assuming job with a lower key (expected running time) has higher priority.**

1. add a time keeper, show the running status evolution of the jobs

2. add job scheduler, to add a job into the MinHeap.

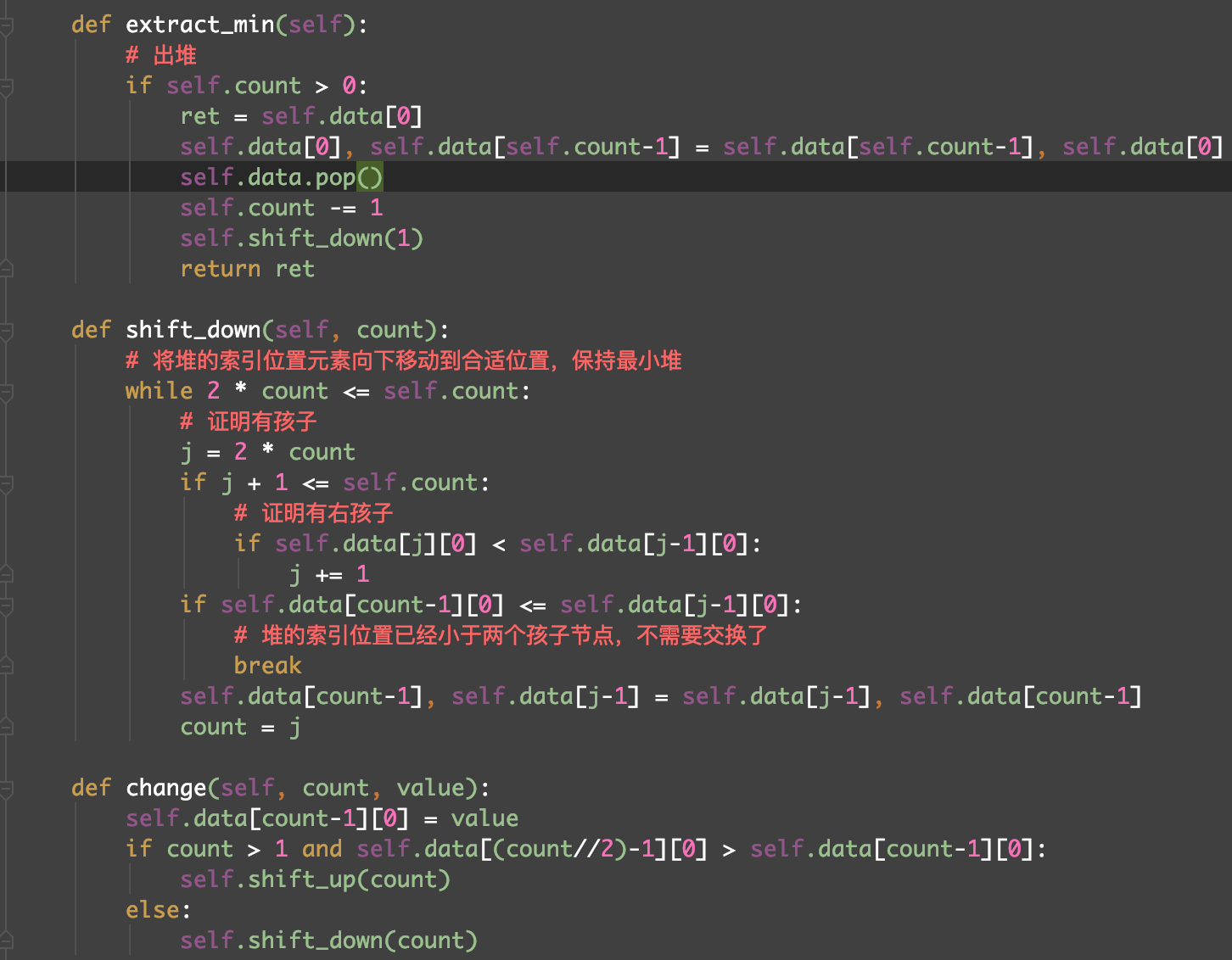
3. add a trouble-maker, to randomly pick an element in the MinHelp and add a random value to its key.

要求：

* 1. 每3秒增加一个任务（包括完成时间key值/名称等）
  2. 完成任务的时间为1-10的随机数
  3. 开始执行任务时，删除该任务；完成该任务后，可执行下一个任务
  4. 每5秒调整第k个任务，k为1-n的随机数
  5. 每次删除任务前，每次调整任务前/后，按顺序打印当前Heap中的所有节点（包括名称/key值等）
  6. 增/删/改任务的执行时间忽略
  7. 指定第N秒时可以停止（如N=200）

A. Implementation of the MinHeap：





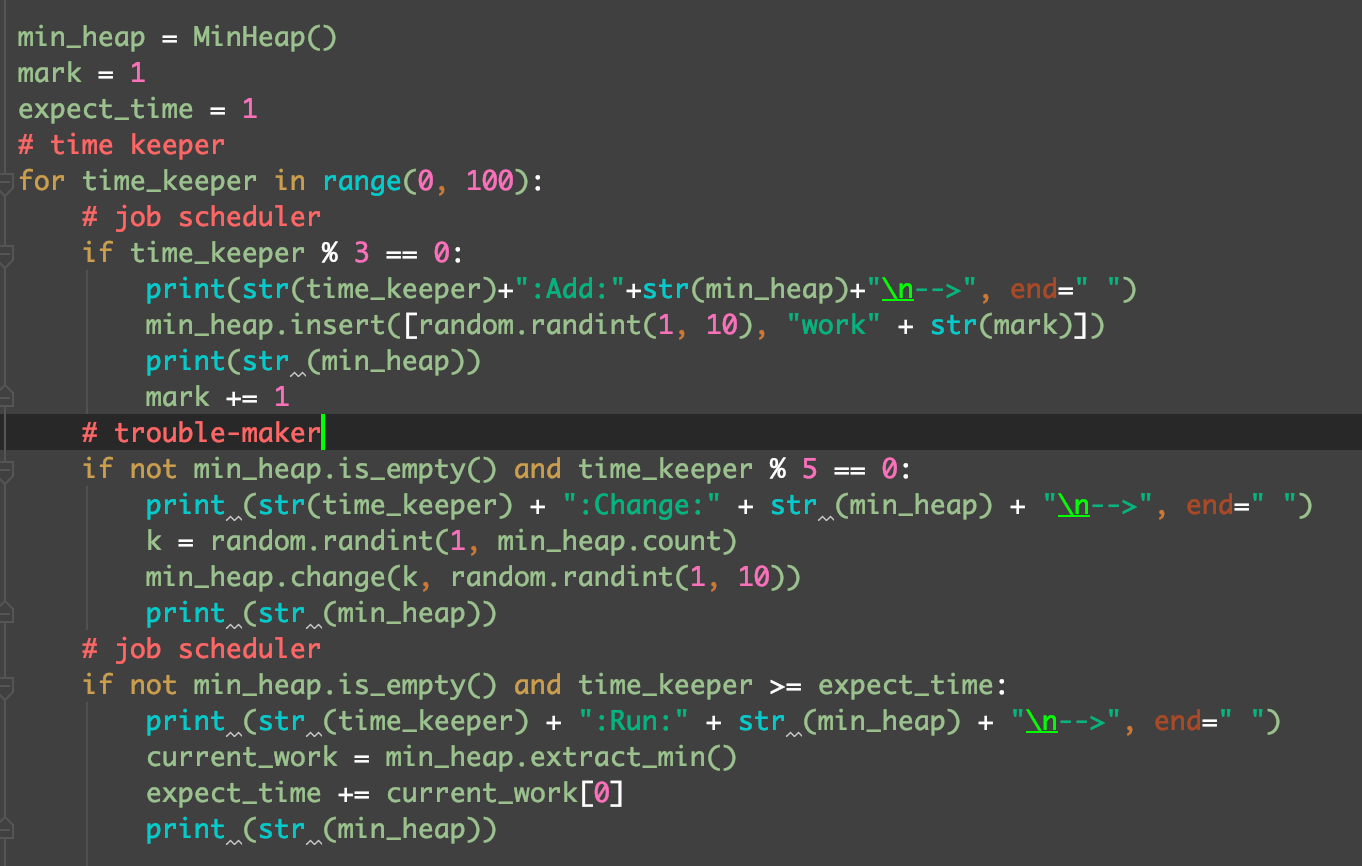
The ***insert*** will insert a new element into the MinHeap and shift it.

The ***extract\_min*** will remove the root of the MinHeap and shift it.

The ***change*** will change the key value of specific node and shift the MinHeap.

The ***shift\_up*** and ***shift\_down*** could be used to shift the MinHeap form a specific node with different direction.

B. time keeper, job scheduler and trouble-maker:

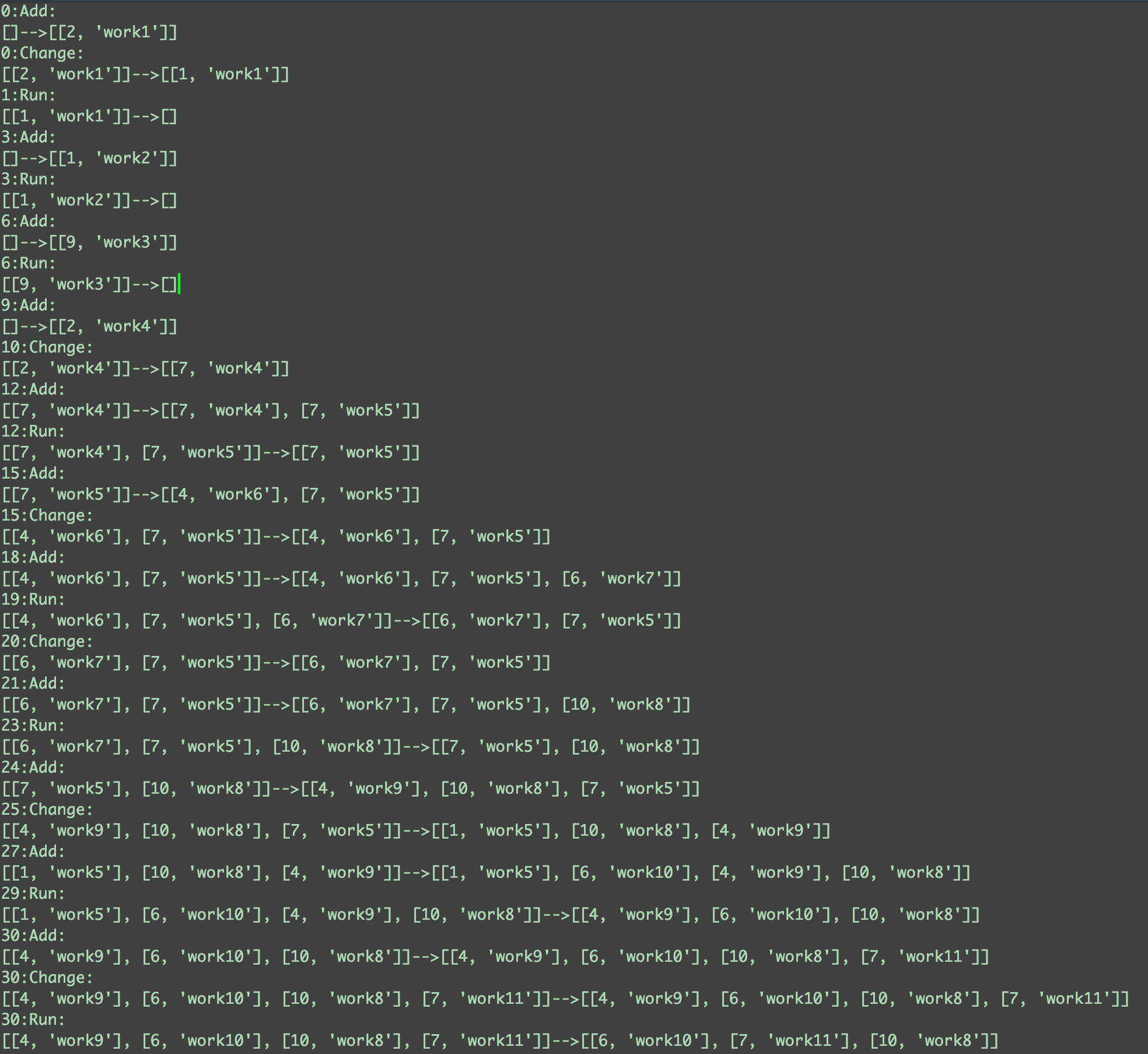


The time keeper increase from 0 to 100.

Since while the time keeper equal to 100, the length of the MinHeap will be too large to show in this document, I simply use all of the output before time keeper become larger than 30 as the output.

In first line of every turn of the output, the number show the value of time keeper and the word after the “:” is the operation will be operated on the MinHeap.

In second line of every turn of the output, the MinHeap without option at that second in front of the “-->”. The changed MinHeap is showed after the “-->” .

The output: