PJ1：高铁候补排队系统

代码逻辑：

数据结构部分：

堆：采用100个元素的顺序存储

队列：用堆实现

元素：包含（int）id，(char\*)name，(int)priority三个元素

主函数部分：

在while（1）循环中，每次输入一个字符串命令，用分支判断应当进入哪个功能函数并进行相应的输入。用end退出

各功能分析：

1. enqueue：

在堆的尾部添加一个新的元素，然后再堆中逐层向上筛选，直到其上的元素优先级大于当前新元素优先级，则停止移动

1. dequeue：

取堆顶元素，将堆底元素放置在堆顶，然后将堆顶元素逐层向下筛选，直至放置到合适的位置（即比下层最大优先级更大）

1. print：

按照顺序堆的顺序逐个输出

1. quit：

遍历整个堆，找到与id对应的元素，存储到temp元素，将堆底元素放在id元素所在位置，然后逐层向下筛选，直至放到合适的位置。最后返回temp。

1. update：

先将对应id的元素quit，存储至temp，然后将temp的优先级进行修改，再enqueue进堆。

堆操作函数：

Shiftup：从堆底开始逐层向上遍历，若其根节点优先级大于此节点，则停止遍历，并将堆底节点放置于此位置；否则将根节点放置于此节点位置。并向上一层，继续遍历。

Shiftdown：从堆顶开始遍历，找到子节点中较高优先级进行比较，如果子节点优先级低于顶节点，则停止遍历，并将顶节点放置于当前遍历位置。否则将子节点放置于当前遍历位置，并遍历下一层，直至堆底。

复杂度分析：

1. enqueue：最坏情况是从底部层层遍历至顶部，即o（logn）复杂度，最好情况o（1）复杂度，平均复杂度是o(logn)但是比最坏情况小
2. dequeue：最坏情况是从顶部层层遍历至底部，o（logn）复杂度，通常情况下和最坏情况相似，即o（logn）
3. print：遍历整个堆，复杂度o（n）
4. quit：分为两个部分，寻找id和出堆，寻找id复杂度为o（n），因此复杂度为o（n）
5. update：分为两部分，quit和enqueue，因为enqueue复杂度为o（logn），quit复杂度为o（n），因此整体复杂度为o（n）

与生成四个优先级队列的方法，堆排序的优缺点：

1. 堆排序的方法，在时间复杂度上，针对enqueue，dequeue两个操作，其时间复杂度较高。针对print，quit，update三个操作，因为均有遍历整个队列的操作，因此和四个优先级队列的方法区别不大。
2. 堆排序的方法，对于某固定人数（例如n人），其总的内存空间为n，不需要额外的空间，而生成四个优先级队列的方法，总内存空间为4n，要明显高于堆排序方法，因此对于大数据的实际情况，堆排序要更合适。
3. 堆排序的优先级是直接通过整型比大小得到的，也就是其优先级数量不固定，可以在后期随时更改，但是生成四个队列的方法就确定了其只有四个优先级，后期修改困难。