华东师范大学数据科学与工程学院上机实践报告

课程名称：算法设计与分析	**年级：**19 级	**上机实践成绩：**
指导教师：金澈清	**姓名：**龚敬洋	
上机实践名称：优先级队列	**学号：**10195501436	**上机实践日期：**
上机实践编号：No.3	**组号：**1-436	

一、目的

1. 熟悉算法设计的基本思想
2. 掌握优先级队列的方法

二、内容与设计思想

1. 利用堆实现优先级队列；
2. 按照顺序插入 1, 3, 5, 7, 9, 2, 4, 6, 8, 10, 11, 13, 15, 12, 14，构建优先级队列，打印出整个数组的内容；
3. 按照顺序插入 9, 7, 10, 12, 5, 4, 2, 1, 15, 14, 3, 7, 8, 6, 11, 13，构建优先级队列，打印出整个数组的内容，并且体会不同输入顺序的情况之下数组内元素排序的差异；
4. 随机生成 1000、10000、100000、1000000 个数，分别构建优先级队列，画图描述不同情况下的运行时间差异。

三、使用环境

推荐使用 C/C++集成编译环境。

四、实验过程

1. *写出算法的源代码；*

随机数生成器

```cpp
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <ctime>
using namespace std;
int main(){
    ofstream fout("data.txt");
    int n;
    srand(time(0));
    cin>>n;
    for(int i = 0; i < n; i++) fout<<rand()<<" ";
    fout.close();
    return 0;
}
```

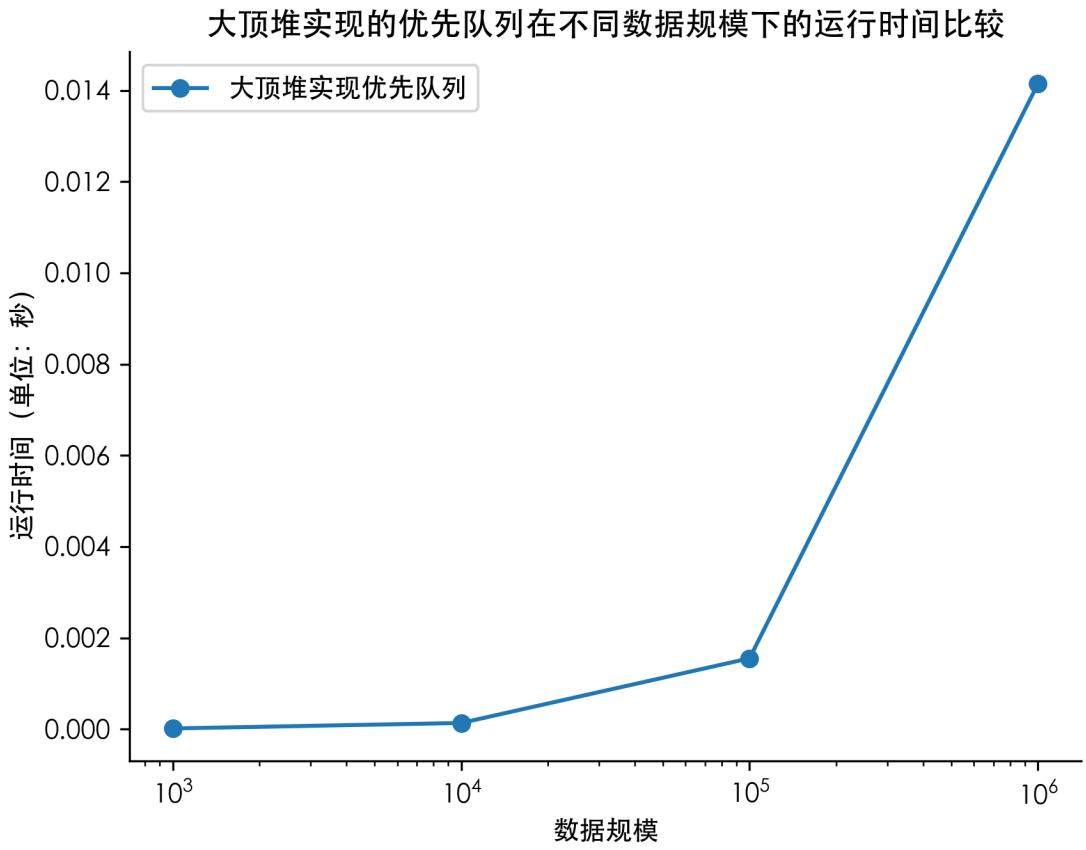
堆实现优先级队列

```cpp
#include <iostream>
#include <fstream>
using namespace std;
int a[1000005], cur;
void siftUp(int index){
    int parent, tmp;
    parent = (index - 1) / 2;
    while(index != 0 && a[index] > a[parent]){
        tmp = a[parent];
        a[parent] = a[index];
        a[index] = tmp;
        index = parent;
        parent = (index - 1) / 2;
    }
}
void siftDown(int index){
    int lchild, rchild, tmp, tcur;
    lchild = index * 2 + 1;
    rchild = (index + 1) * 2;
    while(rchild <= cur && (a[index] < a[lchild] || a[index] < a[rchild])){
        tcur = a[lchild] > a[rchild] ? lchild : rchild;
        tmp = a[tcur];
        a[tcur] = a[index];
        a[index] = tmp;
        index = tcur;
        lchild = tcur * 2 + 1;
        rchild = (tcur + 1) * 2;
    }
    if(lchild <= cur && a[index] < a[lchild]){
        tmp = a[lchild];
        a[lchild] = a[index];
        a[index] = tmp;
    }
}
void add(int data){
    cur++;
    a[cur] = data;
    siftUp(cur);
}
int popMax(){
    int r = a[0];
    a[0] = a[cur];
    cur--;
    siftDown(0);
    return r;
}
void heapify(){
    int tcur;
    tcur = (cur - 1) / 2;
    while(tcur >= 0){
        siftDown(tcur);
        tcur--;
    }
}
int main(){
    ifstream fin("data.txt");
    cur = 0;
    while(!fin.eof()){
        fin>>a[cur];
        cur++;
    }
    cur--;
    heapify();
    for(int i = 0; i < cur; i++) cout<<a[i]<<" ";
    fin.close();
    return 0;
```

```
67. }
```

2. 分别画出各个实验结果的折线图

时间记录使用了 C++自带的 clock()函数，通过在程序开头和结尾分别调用 clock()函数并将两值相减，即可得到程序运行时间。结果如下：



五、总结

对上机实践结果进行分析，问题回答，上机的心得体会及改进意见。

对 1, 3, 5, 7, 9, 2, 4, 6, 8, 10, 11, 13, 15, 12, 14 构建优先级队列的结果为：
15, 11, 14, 8, 10, 13, 12, 6, 7, 3, 9, 5, 2, 1

对 9, 7, 10, 12, 5, 4, 2, 1, 15, 14, 3, 7, 8, 6, 11, 13 构建优先级队列的结果为：
15, 14, 11, 13, 9, 8, 10, 7, 12, 5, 3, 7, 4, 6, 2, 1

若将顺序改为 10, 5, 6, 7, 12, 4, 2, 14, 15, 1, 3, 7, 13, 9, 11, 8，则构建优先级队列的结果为：
15, 14, 13, 10, 12, 7, 11, 8, 7, 1, 3, 6, 4, 9, 2, 5

经实测表明，随着数据规模的增大，运行时间逐步增加，但在数据规模为10^6以内时运行时间均小于 0.1 秒，这与堆实现优先级队列的插入时间复杂度为$O(lgn)$基本相符。