|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm and Data Structure Experimental Assignment | | | |
| Class | 17 CS 01-02 | Teacher Name | Mano and Hong Zhang |
| Student Name | 张钧豪 | Submitted Time | 2019.11.14 |
| Assignment Topic | Searching ‐‐ Practical Exercises | | |
| Assignment Requirement:  Read the array from the text file input.txt. Find the maximum number in the given  list by the priority queue sort with binary heap implementation. The final array  representations of the binary trees should be outputted in the text file output.txt.  For example:  Input‐1.txt is      9, 15, 11, 1, 20, 4, 22  Output‐1.txt is:    22    22, 15, 20, 1, 9, 4, 11 | | | |
| Code of Main Functions:  #include<stdio.h>  #define MAX\_LINE 1024  #define MAX\_NUM 1024  void HeapAdjust(int numbers[], int s, int n)  {  int i, temp;  temp = numbers[s];  for(i=2\*s; i<=n; i\*=2){  /\* i永远指向最大的子节点 \*/  if(i<n && numbers[i] < numbers[i+1]) i++; //右子树大于左子节点 i指向右子节点  if(temp >= numbers[i]) break; //双亲已经大于了最大子节点  numbers[s] = numbers[i];  s = i;  }  numbers[s] = temp;  }  void priority\_q(int numbers[], int n)  {  int i;  for(i=n/2; i>0; i--){ //从下至上，从右到左构建大顶堆  HeapAdjust(numbers, i, n);  }    }  int main()  {  int i, numbers[MAX\_LINE] = {0}, high; // high is down flag of the last numbers[last]  high = read(numbers) - 1;  priority\_q(numbers,high);  write(numbers, high+1);  printf("\n\nAfter HeapSorting the priority queue sort is :\n");  printf("%d ",numbers[1]);  for(i=1;i<high+1;i++)  printf("%d, ",numbers[i]);  printf("\n");  return 0;  } | | | |
| Experimental Results: | | | |