Санкт-Петербургский Научный Исследовательский Университет Информационных Технологий, Механики и Оптики

Задачи пятой недели по курсу «Алгоритмы и структуры данных» на Openedu

Выполнил: студент группы P3218 Артамонов Александр Владимирович

Куча ли?

Структуру данных «куча», или, более конкретно, «неубывающая пирамида», можно реализовать на основе массива.

Для этого должно выполнятся основное свойство неубывающей пирамиды, которое заключается в том, что для каждого $1 \le i \le n$ выполняются условия:

- если 2i≤n, то a[i]≤a[2i];
- если 2i+1≤n, то a[i]≤a[2i+1].

Дан массив целых чисел. Определите, является ли он неубывающей пирамидой.

```
#include "edx-io.hpp"
using namespace std;
//Структура хранит в себе введённое значение(приоритет) и номер строки, в которой оно
было введено
struct number {
      long value;
       long input_time;
};
void swop(number* a, number* b) {
      number temp = *a;
       *a = *b;
      *b = temp;
}
//Обход кучи снизу с і-того элемента, нахождение его места в куче
void heapifyUp(number* array, long i, long* array_of_positions) {
       //Позиция родительского для і-того элемента в куче
       long parent = i / 2 - (1 - i \% 2);
       //Пока не добрались до корня и пока родитель больше
      while (i != 0 && array[i].value < array[parent].value) {</pre>
              //Меняем местами ребёнка и родителя, сохраняя их позиции в массиве позиций
              array_of_positions[array[i].input_time] = parent;
              array_of_positions[array[parent].input_time] = i;
             swop(&array[i], &array[parent]);
              //Переопределяем местоположение элемента и его родителя
              i = parent;
             parent = i / 2 - (1 - i \% 2);
       }
}
//Восстановление кучи сверху
void heapifyDown(number* array, long tail, long* array_of_positions) {
       //Позиция текущего элемента
       long position = 0;
       //Позиции его детей
       long left = 2 * (position + 1) - 1;
      long right = 2 * (position + 1);
      //Пока не дошли до конца очереди, проверяем, что хотя бы один ребёнок меньше
родителя и при этом находится в очереди
      while (position != tail && ((array[position].value > array[left].value && left <=</pre>
tail) || (array[position].value > array[right].value && right <= tail))) {
             //Находим наименьшего, при условии, что правый находится в очереди
```

```
if (array[right].value < array[left].value && right <= tail) {</pre>
                     //Меняем местами ребёнка и родителя, сохраняя их позиции в массиве
позиций
                    array_of_positions[array[position].input_time] = right;
                    array_of_positions[array[right].input_time] = position;
                     swop(&array[right], &array[position]);
                     //Переопределяем позицию сортируемого элемента
                    position = right;
             else {
                     array of positions[array[position].input time] = left;
                     array of positions[array[left].input time] = position;
                     swop(&array[left], &array[position]);
                    position = left;
              //Переопределяем детей сортируемого элемента
             left = 2 * (position + 1) - 1;
             right = 2 * (position + 1);
      }
}
int main() {
       long N;
      io \gg N;
      //Очередь с приоритетами
      number* Queue = new number[N];
      long head = 0;
      long tail = -1;
      //Массив, индекс которого - номер строки, в которой ввели число, а значение -
положение с очереди
      long* array = new long[N];
      char action;
      long a, temp;
      for (long i = 0; i < N; i++) {
             io >> action;
             switch (action)
              {
             case 'A':
                    io >> a;
                     //Добавляем в очередь структуру (значение, номер строки)
                    Queue[++tail] = number{ a, i };
                     //Записываем начальную позицию введённого числа
                    array[i] = tail;
                     //Располагаем наш элемент в куче по приоритету
                    heapifyUp(Queue, tail, array);
                    break;
             case 'X':
                    if (head > tail) {
                           io << '*' << '\n';
                    else {
                            //Меняем местами первый и последний элемент, сохраняя при этом
целостность массива array
                            array[Queue[head].input_time] = tail;
                            array[Queue[tail].input_time] = head;
                            swop(&Queue[head], &Queue[tail]);
                            io << Queue[tail--].value << '\n';</pre>
                            //Восстанавливаем кучу
```

```
heapifyDown(Queue, tail, array);
                    }
                    break;
             case 'D':
                    io >> a;
                    io >> temp;
                    //Позиция элемента введённого в а-той строке сохранена в массиве
array
                    Queue[array[a - 1]].value = temp;
                    //Располагаем измененный элемент в куче
                    heapifyUp(Queue, array[a - 1], array);
                    break;
             default:
                    break;
             }
      }
      return 0;
}
```

Max 0.062 16760832 10945420 3 1 OK 0.000 2224128 14 2 2 OK 0.000 2236416 14 3 3 OK 0.000 2240512 1092 3 4 OK 0.000 2234218 1099 2 5 OK 0.000 2236416 1100 3 7 OK 0.015 2224128 1098 3 8 OK 0.015 2224128 1098 3 9 OK 0.001 2236416 1105 2 10 OK 0.000 2224128 1093 3 9 OK 0.000 2224128 1095 2 11 OK 0.000 2224128 1095 2 11 OK 0.000 2232320 10931 3 12 OK 0.000 2228224 10934 3 <th>№ теста</th> <th>Результат</th> <th>Время, с</th> <th>Память</th> <th>Размер входного файла</th> <th>Размер выходного файла</th>	№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
2 OK 0.000 2236416 14 3 3 OK 0.000 2240512 1092 3 4 OK 0.000 2236416 889 3 5 OK 0.000 2236416 1100 3 6 OK 0.000 2236416 1100 3 7 OK 0.015 2224128 1098 3 8 OK 0.015 2224128 1093 3 9 OK 0.000 2236416 1105 2 10 OK 0.000 2236416 1105 2 10 OK 0.000 223428 1095 2 11 OK 0.000 2232320 10931 3 12 OK 0.000 2232320 10928 2 14 OK 0.000 2232320 10989 3 15 OK 0.015 2232320 10960 <	Max		0.062	16760832	10945420	3
3 OK 0.000 2240512 1092 3 4 OK 0.000 2236416 889 3 5 OK 0.000 2224128 1099 2 6 OK 0.000 2236416 1100 3 7 OK 0.015 2224128 1098 3 8 OK 0.015 2224128 1093 3 9 OK 0.000 2236416 1105 2 10 OK 0.000 2223220 10931 3 11 OK 0.000 2228224 8837 3 12 OK 0.000 2228224 10934 3 13 OK 0.000 2228224 10934 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10960	1	ОК	0.000	2224128	14	2
4 OK 0.000 2236416 889 3 5 OK 0.000 2224128 1099 2 6 OK 0.000 2236416 1100 3 7 OK 0.015 2224128 1098 3 8 OK 0.015 2224128 1093 3 9 OK 0.000 2236416 1105 2 10 OK 0.000 2232320 10931 3 11 OK 0.000 2232320 10931 3 12 OK 0.000 2232320 10928 2 13 OK 0.000 2228224 10934 3 14 OK 0.000 2228224 10934 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2282224 10934 3 17 OK 0.000 2232320 10989 3 18 OK 0.015 2232320 10978 2	2	ОК	0.000	2236416	14	3
5 OK 0.000 2224128 1099 2 6 OK 0.000 2236416 1100 3 7 OK 0.015 2224128 1098 3 8 OK 0.015 2224128 1093 3 9 OK 0.000 2236416 1105 2 10 OK 0.000 2224128 1095 2 11 OK 0.000 2232320 10931 3 12 OK 0.000 2228224 8837 3 13 OK 0.000 2232320 10928 2 14 OK 0.000 2228224 10934 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 <td>3</td> <td>ОК</td> <td>0.000</td> <td>2240512</td> <td>1092</td> <td>3</td>	3	ОК	0.000	2240512	1092	3
66 OK 0.000 2236416 1100 3 7 OK 0.015 2224128 1098 3 8 OK 0.015 2224128 1093 3 9 OK 0.000 2236416 1105 2 10 OK 0.000 2224128 1095 2 11 OK 0.000 2232320 10931 3 12 OK 0.000 2228224 8837 3 13 OK 0.000 2232320 10928 2 14 OK 0.000 2228224 10934 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474<	4	ОК	0.000	2236416	889	3
7 OK 0.015 2224128 1098 3 8 OK 0.015 2224128 1093 3 9 OK 0.000 2236416 1105 2 10 OK 0.000 2224128 1095 2 11 OK 0.000 2232320 10931 3 12 OK 0.000 2232320 10931 3 13 OK 0.000 2232320 10928 2 14 OK 0.000 2232320 10928 2 15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10969 3 18 OK 0.015 2232320 10978 2 18 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 1094	5	ОК	0.000	2224128	1099	2
8 OK 0.015 2224128 1093 3 9 OK 0.000 2236416 1105 2 110 OK 0.000 2224128 1095 2 111 OK 0.000 2232320 10931 3 12 OK 0.000 2228224 8837 3 13 OK 0.000 2232320 10928 2 14 OK 0.000 2228224 10934 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10989 3 18 OK 0.015 2232320 10978 2 18 OK 0.015 2232320 10978 2 20 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 1	6	ОК	0.000	2236416	1100	3
9 OK 0.000 2236416 1105 2 110 OK 0.000 2224128 1095 2 111 OK 0.000 2232320 10931 3 112 OK 0.000 2228224 8837 3 13 OK 0.000 2232320 10928 2 14 OK 0.000 2228224 10934 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 109474 3 20 OK 0.000 2265088 109362 2 21 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 10943 2 26 OK 0.000 2265088 10943 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 30 OK 0.015 3309568 1094726 2 30 OK 0.031 3309568 1094726 2 30 OK 0.031 3309568 109408 3 31 OK 0.031 3309568 109408 3	7	ОК	0.015	2224128	1098	3
10 OK 0.000 2224128 1095 2 111 OK 0.000 2232320 10931 3 112 OK 0.000 2228224 8837 3 113 OK 0.000 2232320 10928 2 114 OK 0.000 2228224 10934 3 115 OK 0.015 2232320 10989 3 116 OK 0.000 2228224 10934 3 117 OK 0.000 2228224 10934 3 118 OK 0.000 2232320 10978 2 118 OK 0.015 2232320 10978 2 118 OK 0.015 2232320 10960 2 119 OK 0.015 2265088 109474 3 120 OK 0.000 2265088 109474 3 120 OK 0.000 2265088 109362 2 11 OK 0.000 2265088 109479 3 121 OK 0.000 2265088 109479 3 122 OK 0.000 2277376 109486 3 123 OK 0.000 2277376 109486 3 124 OK 0.031 2265088 109443 2 125 OK 0.000 2256508 10943 2 126 OK 0.000 2256508 10943 2 127 OK 0.000 2256896 109493 2 128 OK 0.000 3309568 109437 3 129 OK 0.015 3309568 1094726 2 100 OK 0.015 3309568 109408 3 10 OK 0.015 3309568 109408 3	8	ОК	0.015	2224128	1093	3
11 OK 0.000 2232320 10931 3 12 OK 0.000 2228224 8837 3 13 OK 0.000 2232320 10928 2 14 OK 0.000 2232320 10928 2 14 OK 0.0015 2232320 10989 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 109474 3 20 OK 0.000 2265088 109474 3 21 OK 0.000 2265088 109479 3 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 109443 2 25 OK 0.000 2265088 109443 2 26 OK 0.000 2265088 109443 2 27 OK 0.000 3309568 109493 2 28 OK 0.015 3100672 886879 3 30 OK 0.015 3309568 1094726 2 30 OK 0.031 3309568 1094726 2 30 OK 0.031 3309568 1094117 3 31 OK 0.031 3309568 1094308 3	9	ОК	0.000	2236416	1105	2
12 OK 0.000 2228224 8837 3 13 OK 0.000 2232320 10928 2 14 OK 0.000 2228224 10934 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 89095 3 21 OK 0.000 2265088 109362 2 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 10943 2 25 OK 0.000 2265088 109493 2 26 OK 0.000 3309568 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3309568 1094726 2 30 OK 0.031 3309568 1094726 2 30 OK 0.031 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.001 3309568 1094215 3	10	ОК	0.000	2224128	1095	2
13 OK 0.000 2232320 10928 2 14 OK 0.000 2228224 10934 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 109474 3 21 OK 0.000 2265088 109362 2 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 109565 2 26 OK 0.000 2265088 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 31 OK 0.031 3309568 1094726 2 30 OK 0.031 3309568 1094308 3 31 OK 0.031 3309568 1094308 3	11	ОК	0.000	2232320	10931	3
14 OK 0.000 2228224 10934 3 15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 109474 3 21 OK 0.000 2265088 109362 2 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2256896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3309568	12	ОК	0.000	2228224	8837	3
15 OK 0.015 2232320 10989 3 16 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 89095 3 21 OK 0.000 2265088 109362 2 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 109565 2 26 OK 0.000 2266896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.031 3309568 1094726 3 31 OK 0.031 3309568 1094308 3 32 OK 0.031 3309568 1094308 3	13	ОК	0.000	2232320	10928	2
16 OK 0.000 2228224 10934 3 17 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 89095 3 21 OK 0.000 2265088 109362 2 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 225688 109493 2 26 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094308 3 31 OK 0.031 3309568 10943	14	ОК	0.000	2228224	10934	3
17 OK 0.000 2232320 10978 2 18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 89095 3 21 OK 0.000 2265088 109362 2 22 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 109965 2 26 OK 0.000 2265088 109443 2 27 OK 0.000 3309568 109493 2 28 OK 0.015 3100672 886879 3 30 OK 0.015 3309568 1094717 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094308 3	15	ОК	0.015	2232320	10989	3
18 OK 0.015 2232320 10960 2 19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 89095 3 21 OK 0.000 2265088 109362 2 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 225688 109493 2 26 OK 0.000 3309568 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.031 3309568 1094308 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 10	16	ОК	0.000	2228224	10934	3
19 OK 0.015 2265088 109474 3 20 OK 0.000 2265088 89095 3 21 OK 0.000 2265088 109362 2 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 109965 2 26 OK 0.000 2256896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.031 3309568 1094308 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	17	ОК	0.000	2232320	10978	2
20 OK 0.000 2265088 89095 3 21 OK 0.000 2265088 109362 2 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2256886 109565 2 26 OK 0.000 2256896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	18	ОК	0.015	2232320	10960	2
21 OK 0.000 2265088 109362 2 22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 109565 2 26 OK 0.000 2256896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094215 3 32 OK 0.000 3309568 1094215 3	19	ОК	0.015	2265088	109474	3
22 OK 0.000 2265088 109479 3 23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 109565 2 26 OK 0.000 2256896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	20	ОК	0.000	2265088	89095	3
23 OK 0.000 2277376 109486 3 24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 109565 2 26 OK 0.000 2256896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	21	ОК	0.000	2265088	109362	2
24 OK 0.031 2265088 109443 2 25 OK 0.000 2265088 109565 2 26 OK 0.000 2256896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	22	ОК	0.000	2265088	109479	3
25 OK 0.000 2265088 109565 2 26 OK 0.000 2256896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	23	ОК	0.000	2277376	109486	3
26 OK 0.000 2256896 109493 2 27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	24	ОК	0.031	2265088	109443	2
27 OK 0.000 3309568 1094387 3 28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	25	ОК	0.000	2265088	109565	2
28 OK 0.015 3100672 886879 3 29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	26	ОК	0.000	2256896	109493	2
29 OK 0.015 3309568 1094726 2 30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	27	ОК	0.000	3309568	1094387	3
30 OK 0.015 3309568 1094117 3 31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	28	OK	0.015	3100672	886879	3
31 OK 0.031 3309568 1094308 3 32 OK 0.000 3309568 1094215 3	29	ОК	0.015	3309568	1094726	2
32 OK 0.000 3309568 1094215 3	30	ОК	0.015	3309568	1094117	3
	31	ОК	0.031	3309568	1094308	3
33 OK 0.015 3309568 1094084 2	32	ОК	0.000	3309568	1094215	3
	33	ОК	0.015	3309568	1094084	2
34 OK 0.015 3309568 1094403 2	34	ОК	0.015	3309568	1094403	2
35 OK 0.046 16756736 10944156 3	35	ОК	0.046	16756736	10944156	3

Очередь с приоритетами

Реализуйте очередь с приоритетами. Ваша очередь должна поддерживать следующие операции: добавить элемент, извлечь минимальный элемент, уменьшить элемент, добавленный во время одной из операций.

Формат входного файла

В первой строке входного файла содержится число $n (1 \le n \le 10^6)$ - число операций с очередью.

Следующие п строк содержат описание операций с очередью, по одному описанию в строке. Операции могут быть следующими:

- A x требуется добавить элемент x в очередь.
- X требуется удалить из очереди минимальный элемент и вывести его в выходной файл. Если очередь пуста, в выходной файл требуется вывести звездочку «*».
- D x y требуется заменить значение элемента, добавленного в очередь операцией A в строке входного файла номер x+1, на y. Гарантируется, что в строке x+1 действительно находится операция A, что этот элемент не был ранее удален операцией X, и что у меньше, чем предыдущее значение этого элемента.

В очередь помещаются и извлекаются только целые числа, не превышающие по модулю 10^9 .

```
#include "edx-io.hpp"
using namespace std;
int main() {
      long N;
       io >> N;
      long* array = new long[N];
      for (long i = 0; i < N; i++) {
             io >> array[i];
      bool isHeap = true;
       //Для каждого узла проверяем, что его дети больше него
       //Если хоть в одном случае это не так, значит данный массив - не куча
      for (long i = 1; i < N / 2 + N % 2; i++) {
             if (array[i - 1] > array[2 * i - 1] || array[i - 1] > array[2 * i]) {
                     isHeap = false;
                     break;
              }
       if (isHeap) {
             io << "YES";</pre>
       else {
              io << "NO";
       return 0;
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.375	22732800	12083657	5694235
1	ОК	0.015	2220032	37	12
2	ОК	0.015	2236416	6	3
3	ОК	0.000	2224128	11	3
4	ОК	0.000	2220032	22	4
5	ОК	0.031	2224128	19	6
6	ОК	0.015	2224128	19	6
7	ОК	0.015	2224128	19	6
8	ОК	0.000	2224128	48	19
9	ОК	0.015	2224128	58	29
10	ОК	0.000	2236416	57	28
11	ОК	0.000	2224128	48	19
12	ОК	0.000	2224128	58	29
13	ОК	0.015	2240512	57	28
14	ОК	0.000	2224128	828	573
15	ОК	0.015	2220032	1037	369
16	ОК	0.015	2232320	828	573
17	ОК	0.000	2220032	988	404
18	ОК	0.015	2236416	1082	300
19	ОК	0.000	2224128	1139	240
20	ОК	0.000	2236416	930	377
21	ОК	0.000	2220032	1190	280
22	ОК	0.000	2240512	8184	5678
23	ОК	0.000	2240512	10768	3637
24	ОК	0.015	2252800	8206	5700
25	ОК	0.000	2240512	9903	3928
26	ОК	0.000	2240512	10814	3000
27	ОК	0.015	2240512	11338	2400
28	ОК	0.015	2236416	11138	3582
29	ОК	0.000	2252800	10904	3851
30	ОК	0.000	2326528	81951	56944
31	ОК	0.015	2314240	110901	36274
32	ОК	0.015	2306048	81971	56964
33	ОК	0.000	2330624	99351	39719
34	ОК	0.000	2322432	107882	30000
25	OK	0.015	2330624	112101	24000