

Пирамидальная сортировка HeapSort



Пирамида Хеопса

Пирамидальная сортировка или метод Вильямса – Флойда (Williams, Floyd, 1964)

Пирамидальная сортировка основана на алгоритме построения пирамиды.

Определение

Последовательность a_L, a_{L+1}, \dots, a_R называется **пирамидой**, если неравенство

$$a_i \leq \min(a_{2i}, a_{2i+1})$$

выполняется **для всех** i , для которых хотя бы один из элементов a_{2i} и a_{2i+1} существует

Пример

2	3	4	5	6	7	8
3	2	6	3	4	5	7
1	2	3	4	5	6	7

Свойства пирамиды



1. Двустороннее усечение:

Если последовательность $a_L, a_{L+1}, \dots, a_{R-1}$,

a_R – пирамида, то a_{L+1}, \dots, a_{R-1} тоже пирамида.

2. Если a_1, a_2, \dots, a_n – пирамида,
то a_1 – минимальный элемент пирамиды.

3. Если a_1, \dots, a_n – произвольная

Построение пирамиды

Пусть a_{L+1}, \dots, a_R - пирамида,
необходимо добавить элемент X ,
чтобы получить новую пирамиду a_L, \dots, a_R .

Новый элемент **добавляем в начало**,
расширяя последовательность влево.

Если a_L удовлетворяет **условию пирамиды**,
то пирамида построена.

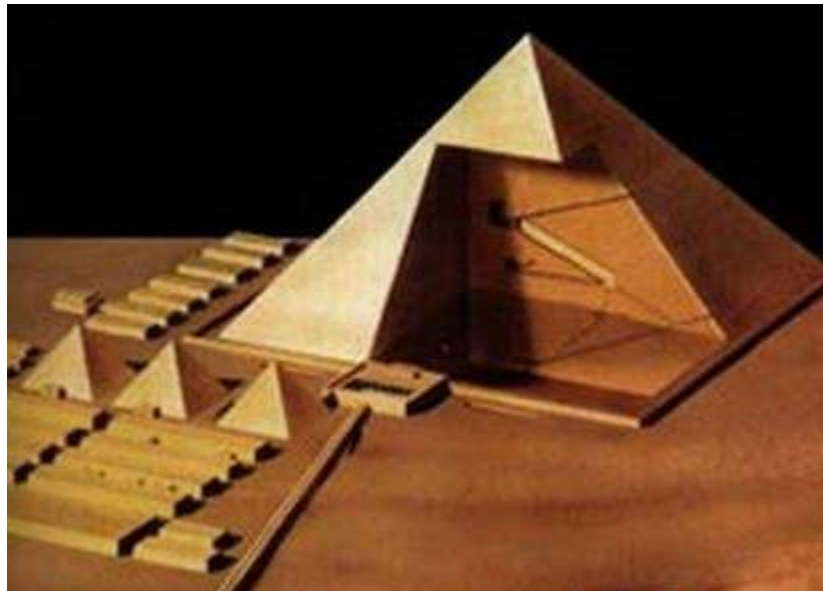


Построение пирамиды

Иначе найдутся такие a_{2L} или a_{2L+1} , что **не будут удовлетворять** условию пирамиды.

Возьмем минимальный элемент из a_{2L} и a_{2L+1} , обозначим его за a_j и обменяем с a_L .

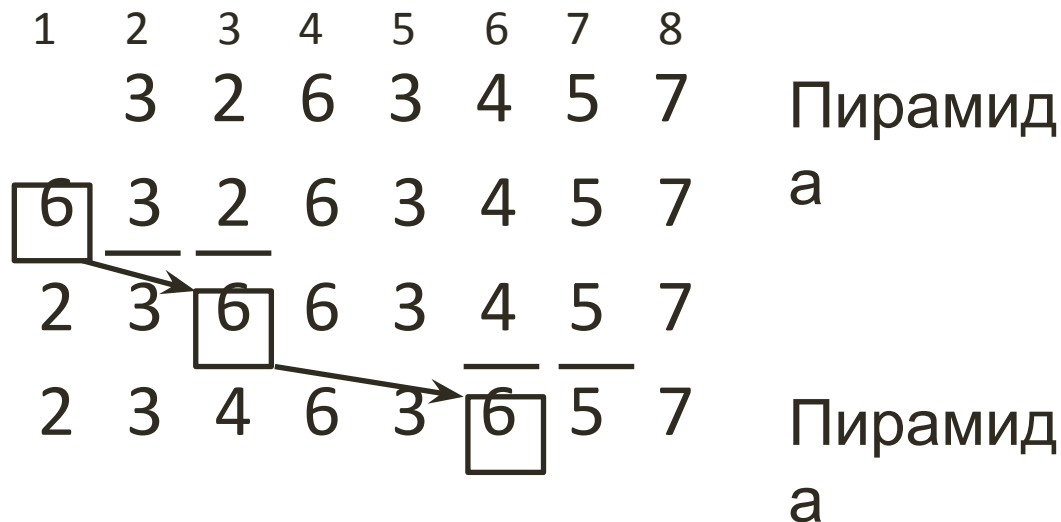
В результате получим $a'_L \leq a_{2L}$ и $a'_L \leq a_{2L+1}$, что удовлетворяет **условию пирамиды**.



Построение пирамиды

Теперь элемент **X** попал на место a_j и для него необходимо проверить **условие пирамиды**, и так до конца массива.

Пример:



Построение пирамиды (L, R)

Алгоритм на псевдокоде

a_{L+1}, \dots, a_R – на входе пирамида (L+1, R)
 a_L – НОВЫЙ ЭЛЕМЕНТ

$x := a_L, i := L$

DO

$j := 2i$

IF ($j > R$) OD FI

IF ($j < R$ и $a_{j+1} \leq a_j$) $j = j + 1$ FI

IF ($x \leq a_j$) OD FI

$a_i = a_j$

$i := j$

OD

$a_i := x$

Трудоёмкость алгоритма построения пирамиды

Определим **верхнюю границу** трудоёмкости алгоритма. На каждой **итерации цикла** выполняется максимум **два** сравнения и **одна** пересылка. Найдем *наибольшее количество итераций* (k). Наихудший случай, когда в перестановках участвуют элементы $a_L, a_{2L}, a_{4L} \dots$

$$a_L \dots a_{2L} a_{2L+1} \dots a_{4L} a_{4L+1} \dots a_m a_{mL+1} \dots a_R$$

$2^1 \qquad \qquad \qquad 2^2 \qquad \qquad \qquad 2^k$

$$mL \leq R \quad 2^k L \leq R \quad 2^k \leq \frac{R}{L} \quad k \leq \log_2 \frac{R}{L}$$

$$C = 2k \quad M = k+2$$

$$C = 2 \log_2 \frac{R}{L} \quad M = \log_2 \frac{R}{L} + 2$$

Пирамидальная сортировка (HeapSort)

Первый этап. Построение пирамиды из элементов массива.

В соответствии со **свойством 3** правая часть массива уже пирамида. Будем добавлять по одному элементу слева, расширяя пирамиду, пока в нее не войдут все элементы массива.

Второй этап. Собственно сортировка.

По **свойству 2** в пирамиде первый элемент минимальный. Производим двустороннее усечение пирамиды: уберем элементы a_1 и a_n .

По **свойству 1** a_2, \dots, a_{n-1} – пирамида. Поставим элемент a_1 на последнее место, а элемент a_n добавим к пирамиде a_2, \dots, a_{n-1} . Отсекаем последний элемент и повторяем действия, пока пирамида не исчезнет.

Пирамидальная сортировка (HeapSort)

Алгоритм на псевдокоде

```
| L := ⌊n/2⌋  
| DO ( L > 0 )  
1 |     Построение пирамиды (L, n)  
|     L := L - 1  
| OD  
|     R := n  
|     DO ( R > 1 )  
2 |         a1 ↔ aR  
|         R := R - 1  
|         Построение пирамиды (1, R)  
|     OD
```

1 2 3 4 5 6 7 8
К У Р А П О В А

П О В А Пирамид

А П О В А Пирамид^а

Р А П О В А^а

В А П О Р А Пирамид

У В А П О Р А^а

А В У П О Р А

А В А П О Р У Пирамид

К А В А П О Р У^а

А К В А П О Р У

А А В К П О Р У

— Пирамид
а

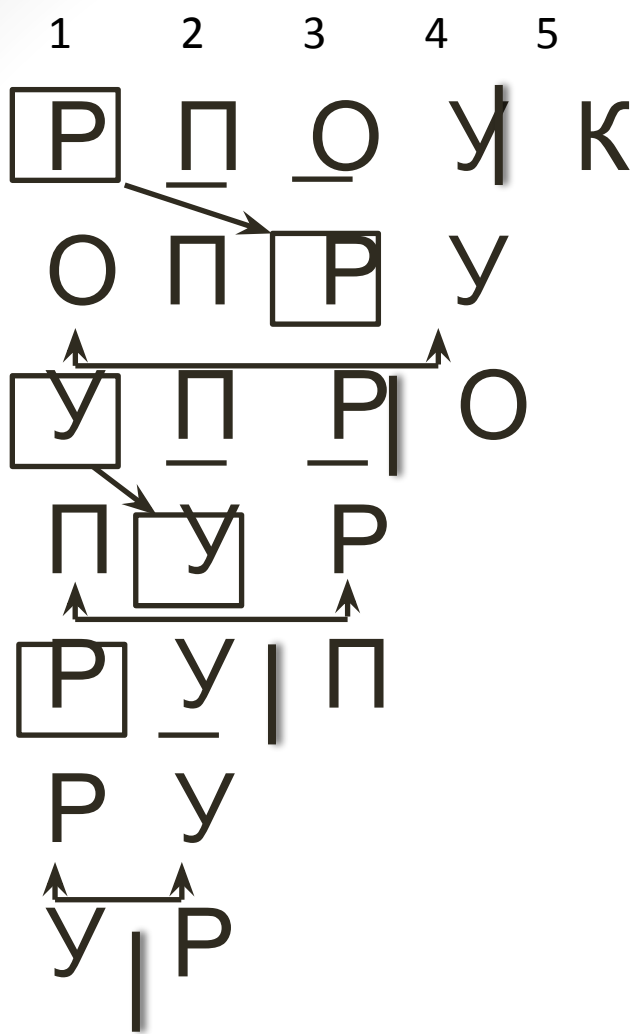


Пирамида

Пирамид
а

Пирамид
а

Пирамид
а



Пирамид
а

Пирамид
а
Пирамид

а
Пирамид
а

У Р П О К В А А

- # Трудоёмкость пирамидальной сортировки (HeapSort)

Оценим трудоёмкость сортировки, используя уже известную оценку трудоёмкости построения пирамиды:

$$C = 2 \log_2 \frac{R}{L} \quad M = \log_2 \frac{R}{L} + 2$$

На первом этапе построение пирамиды производится $n/2$ раз, на втором этапе – $n-1$ раз.

Очевидно, трудоёмкость пирамидальной сортировки имеет порядок $O(n \log_2 n)$, , $n \rightarrow \infty$.

Количество операций сравнения и пересылки оценивается следующими неравенствами:

$$C < 2 n \log_2 n + n + 2 \quad M < n \log_2 n + 6.5n - 4$$

Пирамидальная сортировка **не устойчива**.

Метод **практически не зависит** от исходной упорядоченности массива.

Метод	Трудоемкость	Устойчивость	Зависимость от упорядоченности
ShellSort	$O(n^{1,2})$	Не устойчив	Зависит
HeapSort	$O(n \log_2 n)$	Не устойчив	Практически не зависит