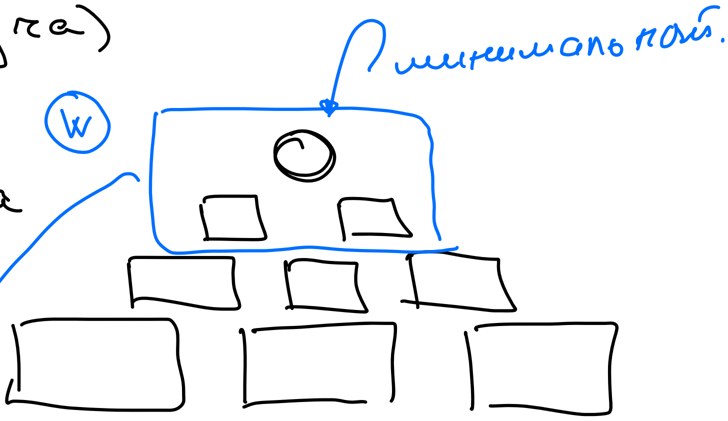
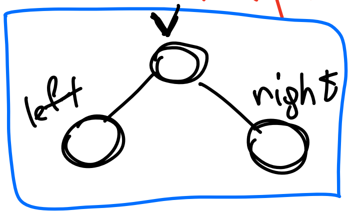
Heap (пирамида)
(куча)

Минимальная пирамида (W)

Максимальная пирамида



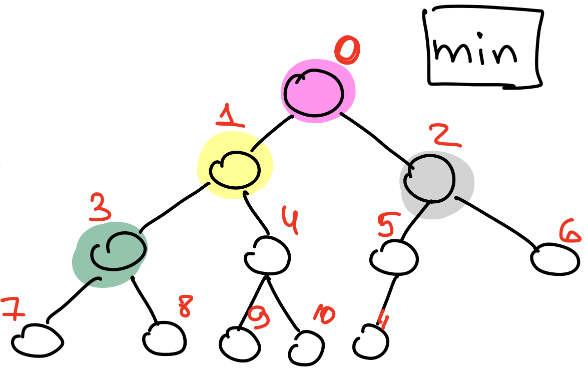
Свойство пирамида



$\text{left} \geq v$

$\text{right} \geq v$

left (v) right



v – номер вершины

$\text{left} = 2v+1$

$\text{right} = 2v+2$

$\text{child} \to \text{parent} = \left\lfloor \frac{\text{child}-1}{2} \right\rfloor$

округление вниз

$7 = 3+4 = 3+3+1 = 2v+1$

$8 = 3+3+2 = 2v+2$

$c = 2v+1$

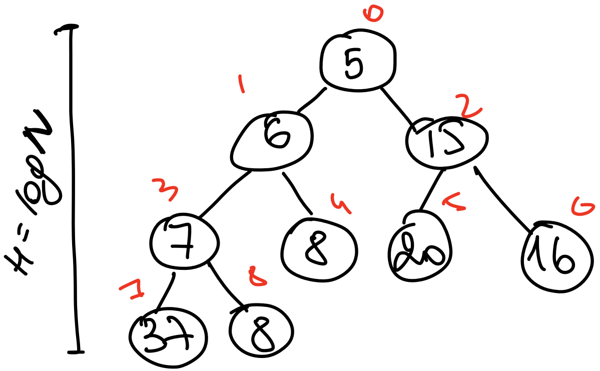
$c-1 = 2v$

$v = \frac{c-1}{2}$

$c = 2v+2$

$c-2 = 2v$

$v = \frac{c-2}{2}$



$0 \to 1$

$1 \to 2$

$2 \to 4$

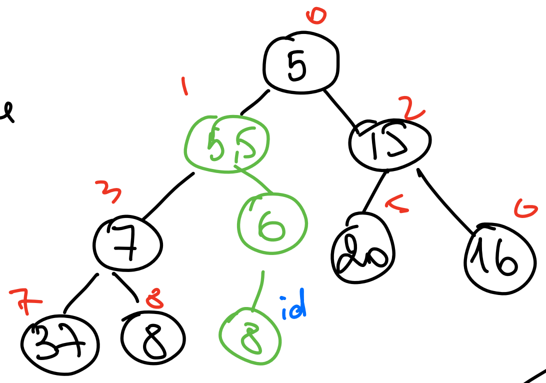
$H \to 2^H = N$

$\boxed{H = \log_2 N}$

	0	1	2	3	4	5	6	7	8
arr=	5	6	15	7	8	20	16	37	8

(а) Find Min()

↳ return arr[0]   Асимп: $\underline{O(1)}$



(б) Просеивание вверх

$H \leq \log_2 N$

Sift Up

1. Идем к папе. сравниваем с ним
2. Если папа больше
   ↳ меняемся с папой местами
3. Если папа меньше
   ↳ закончить.

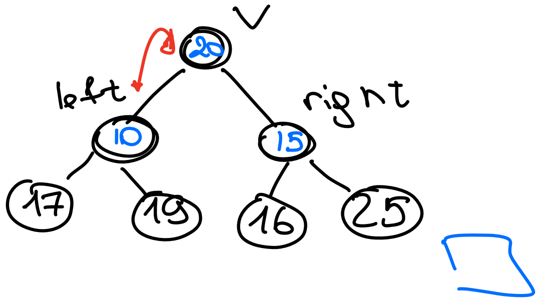
Асимптотика: $\underline{O(h)}$

$\underline{O}(\log_2 N)$

$M(N) = O(1)$

```
метод SiftUp(id)
1.      parent = (id-1)//2
3.      while id > 0 and
                 arr[parent] > arr[id]:
        {
2.          SWAP(arr[id], arr[parent])
            id = parent
1.          parent = (id-1)//2
        }
```

(в) Просеивание вниз (Sift Down)



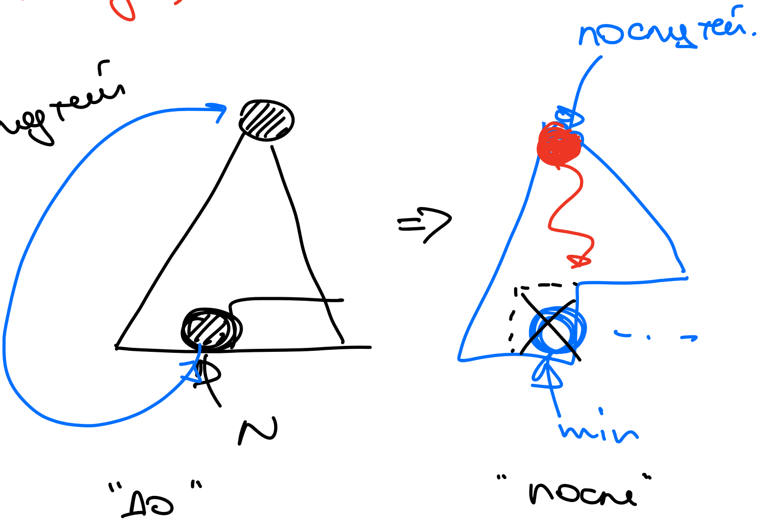
```
SiftDown(id) {
                        "N
    while id < arr.size()
    {
        left = id*2+1
        right = id*2+2
        id_min = id          // в id_min ← меньший
                                       индекс минимальный
        если (left < N) and (arr[left] < arr[id_min])
                ↳ id_min = left
        если (right < N) and (arr[right] < arr[id_min])
        ↳ id_min = right

    если   id_min = id
        ↳ break  // мы дошли до границы
    иначе
        ↳ SWAP (arr[id], arr[id_min])
          id = id_min
}
```

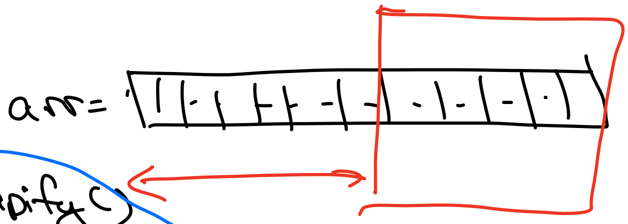
Асимпт $T(N) = \underline{O}(\log N)$

```
② PopMin () {
     min = arr[0]
     swap(arr[0], arr[N-1])
     arr.pop_back()
     SiftDown(0)
     return min
}
```

$T(N) = \underline{O}(\log N)$

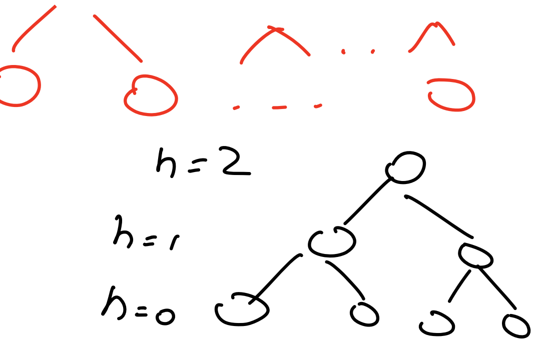


Ⅲ Построение пирамиды



sift Up
sift Down

```
метод Heapify()
    for i = N/2 ... 0
        SiftDown(i)   // O(log N)
```



Асимптотика: $T(N) = \underline{O}(N)$

Док-во:

$$T(N) = \sum_{h=0}^{\log_2 N} \frac{N}{2^{h+1}} \underline{O}(h) =$$

$$= \frac{N}{2} \sum_{h=0}^{\log_2 N} \underline{O}\left(\frac{h}{2^h}\right) = \frac{N}{2}\ \underline{O}\left(\sum_{h=0}^{\log_2 N} \frac{h}{2^h}\right) =$$

$$= \underline{O}\left(\frac{N}{2} \cdot 2\right) = \underline{O}(N)$$

(IV) Научились

$FindMin() \to \underline{O}(1)$

$PopMin() \to \underline{O}(\log_2 N)$

$SiftUp() \to \underline{O}(\log_2 N)$

$SiftDown() \to$

$Heapify \to \underline{O}(N)$

```
HeapSort(arr)
    Heapify(arr)  → построить heap   )O(N)
    for i=0... N:
        print(PopMin())   // O(logN)  )N
```

Асимптотика: $\underline{O}(N + N\log N) = \underline{O}(N \log N)$

(V) Пирамида → N элементов

↕

Куча → push

```
Push(elem) {
    arr.push_back(elem)
    SiftUp(arr[arr.size() - 1])    ] Куча
```

(VI) Сравнение

	Push	Min Max	PopMin PopMax	id
Linked List	1	N	N	1N
Array	1*	N	N	1
Sorted Array	N	1	N	1
Heap	logN	1	logN	—
Stack / Queue				