DSA - Seminar 5

Iterator for a SortedMap represented on a hash table, collision resolution with separate chaining.

- Assume
  - We memorize only the keys from the Map
  - Keys are integer numbers

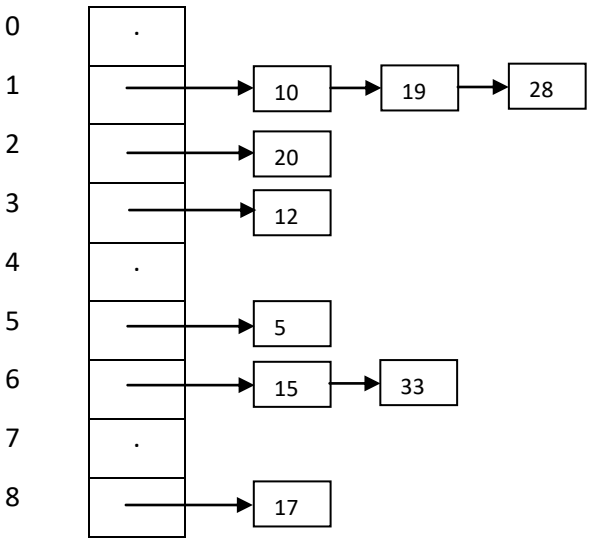
For ex:

- Keys from the map: 5, 28, 19, 15, 20, 33, 12, 17, 10 – Keys have to be unique!
- HT
  - m = 9
  - Hash function defined with the division method
    - h(k) = k mod m

k	5	28	19	15	20	33	12	17	10
h(k)	5	1	1	6	2	6	3	8	1

- h(k) can contain duplicates – they are called collisions

SM:



Iterator:

- If we iterate through the elements using the iterator, they should be visited in the following order: 5, 10, 12, 15, 17, 19, 20, 28, 33
- If we use the iterator -> complexity of the whole iteration to be Θ(n)

Representation:

TNode:
e: TElem // key, value
next: ↑TNode

SortedMap:
m: Integer
T : (↑TNode)[]
h: TFunction
R: relation

IteratorSortedMap:
sm: SortedMap
l: TList
currentNode: ↑TNode

```
subalgorithm init(it, sm):
      it.sm ← sm
      mergeLists (sm, it.l)
      it.currentNode ← it.l.head
end-subalgoritm
```

- mergeLists merges the separate linked lists:
  - first with the second, the result with the third, etc.
  - all lists using a binary heap
- Operations *valid, next, getCurrent* have a complexity of Θ(1)

Complexity of merging:

$$\left.\begin{matrix}\text{HT with m positions}\\ SortedMap\ with\ n\ elems\end{matrix}\right\} \Rightarrow average\ number\ of\ elems\ in\ a\ list: \frac{n}{m} = \alpha\ (load\ factor)$$

Merge the first list with the second, the result with the third, etc.

- list1 + list2 => list12 => α + α = 2α
- list12 + list3 => list123 => 2α + α = 3α
- list123 + list4 => list1234 => 3 α + α = 4 α
- …

Total merging: $2\alpha + 3\alpha + \cdots + m\alpha \approx \left.\begin{matrix}\frac{m*(m+1)}{2}\alpha\\ \alpha = \frac{n}{m}\end{matrix}\right\} \rightarrow \frac{m\,(m+1)}{2}\,\frac{n}{m} \Rightarrow \in\ \theta\ (n*m)$

All lists using a binary heap:

- Add from each list the first node to the heap
- Remove the minimum from the heap, and add to the heap the next of the node (if exists)
- The heap will contain at most k elements at any given time (k is the number of the listst, $1 \leq k \leq$ m) => height of the heap is $O(\log_2 k)$

- Merge complexity:
  - $O(n \log_2 k)$, if k > 1
  - Θ(n), if k = 1