HashCode Spatial Complexity

Type	Variable	Size of 1 atomic value	Number of atomic values
Input	K	16	1
Auxiliar	keyString	16	1
Auxiliar	i	32	1
Auxiliar	kLength	32	1
Output	hash	32	1

HashCode Time Complexity

```
public int hashCode(K key) {
    String keyString = key.toString();              1
    int kLenght = toString().length() - 1;          1
    int hash = 0;                                   1
    for (int i = 0; i < keyString.length(); i++) {   n
        hash += keyString.charAt(i) * Math.pow(128, kLenght);  n-1
        kLenght--;                                             n-1
    }

    hash = (int) Math.floor(limit * ((hash * 0.75) % 1));      1

    return hash;                                               1
}
```

HeapSort Spatial Complexity			
Type	Variable	Size of 1 atomic value	Number of atomic values
Auxiliar	i	32	1
Auxiliar	Pair	32	n

HeapSort Temporal Complexity

```
public Pair<K,V>[] heapSort() {
    buildHeap(this.array);                          1
    for (int i = this.heapSize; i >= 0; i--){       n
        Pair<K,V> pair = this.array[0];             n-1
        this.array[0] = this.array[i];              n-1
        this.array[i] = pair;                       n-1
        this.heapSize--;                            n-1
        maxHeapIfy(this.array, index: 0);           n-1
    }
    return this.array;                              1
}
```