VE281 Writing Assignment Three

Liu Yihao 515370910207

Ex. 1

Let u_i be the number of elements in the i^{th} slot of the hash table generated by the hash function h, then

$$|U| = \sum_{i=0}^{n-1} u_i$$

$$|U|^2 = \left(\sum_{i=0}^{n-1} u_i\right)^2 = \sum_{i=0}^{n-1} u_i^2 + 2\sum_{i=0}^{n-2}\sum_{j=i+1}^{n-1} u_i u_j < n\sum_{i=0}^{n-1} u_i^2$$

$$\epsilon \geqslant Pr(h(k) = h(l)) = \frac{\sum_{i=0}^{n-1} u_i(u_i - 1)}{|U|^2} = \frac{\sum_{i=0}^{n-1} u_i^2 - |U|}{|U|^2} > \frac{\frac{|U|^2}{n} - |U|}{|U|^2} = \frac{1}{n} - \frac{1}{|U|}$$

$$\epsilon > \frac{1}{n} - \frac{1}{|U|}$$

Ex. 2

From the lecture, we know the H as set of all functions that map from U to $\{0, 1, 2, \ldots, n-1\}$ is universal, so

$$\underset{h\in H}{Pr}(h(k) = h(l)) \leqslant \frac{1}{n}$$

Now, we can take away n functions that map U to $\{0\}, \{1\}, \ldots, \{n-1\}$ from H to form H', since these function always collide for all $k \neq l$, taking away them can decrease the average probability of collision. Then we can find that

$$\underset{h\in H'}{Pr}(h(k) = h(l)) < \frac{1}{n}$$

Here is a simple example:

Let $n = 2$, $|U| = 3$, $H = \{h_i(x) | x \in U, i = 0, 1, 2, 3, 4, 5\}$, define $h_i(x)$ as following table:

x	$h_0(x)$	$h_1(x)$	$h_2(x)$	$h_3(x)$	$h_4(x)$	$h_5(x)$
0	1	0	0	1	1	0
1	0	1	0	1	0	1
2	0	0	1	0	1	1

$$Pr(h(0) = h(1)) = Pr(h(1) = h(2)) = Pr(h(0) = h(2)) = \frac{1}{3} < \frac{1}{2}$$

Ex. 3

$$U(L) = \frac{1}{2}\left[1 + \left(\frac{1}{1-L}\right)^2\right] \leqslant 8.5 \Longrightarrow L \leqslant 0.75$$

$$S(L) = \frac{1}{2}\left(1 + \frac{1}{1-L}\right) \leqslant 3 \Longrightarrow L \leqslant 0.8$$

So $L = 0.75$ should be chosen, the hash table size should be $600/0.75 = 800$.

Ex. 4

We want to prove that if the number of full nodes is n, then the number of leaves in a non-empty binary tree is $n+1$. Mathematical induction is used to prove it.

First, when $n = 0$, there is no full node, so the number of leaves is obviously 1.

Then, when $n = k$, suppose the statement is true. When $n = k+1$, one more full node is added now. We know each node have three status: full node, not full node and leaf. We can't add more nodes to a full node. When we add a node to a not full node, it becomes a full node, and there is one more leaf. When we add a node to a leaf, the leaf becomes a not full node, so the number of leaves doesn't change. So we can concluded that when $n = k+1$, the number of leaves is $k+2$, the statement is proved.

Ex. 5

(a) A→B→C→D→E→F→G→H→I

(b) D→C→F→G→E→B→I→H→A

(c) C→D→B→F→E→G→A→I→H

(d) A→B→H→C→E→I→D→F→G

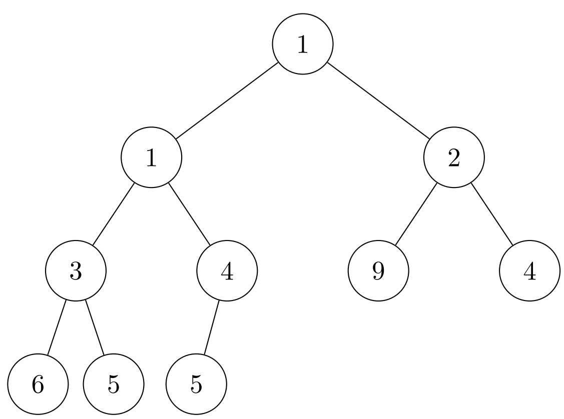
Ex. 6

We can add a bool attribute "visited" to the struct.

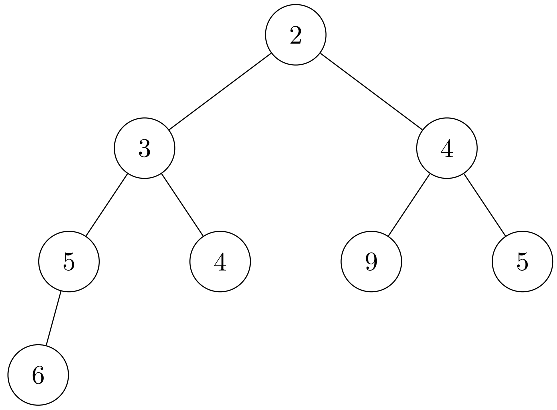
Input: The root node *root*
  $root.visited \leftarrow false$
  push *root* into *stack*
  while *stack* is not empty **do**
    node ← pop a element from *stack*
    if *node.visited* **then**
      if *node.right* exists **then**
        $node.right.visited \leftarrow false$
        push *node.right* into *stack*
      end if
      $node.visited \leftarrow true$
      push *node* into *stack*
      if $node.left$ exists **then**
        $node.left.visited \leftarrow false$
        push $node.left$ into *stack*
      end if
    else
      do something with *node*
    end if
  end while

Ex. 7

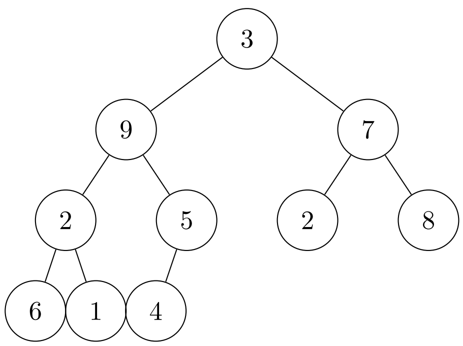
(a)

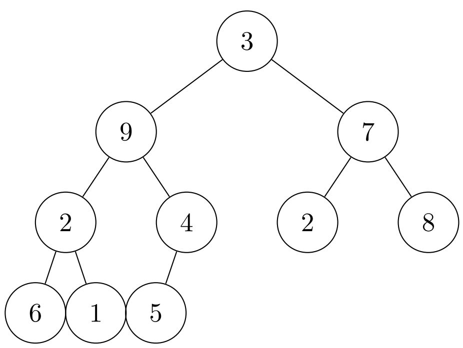


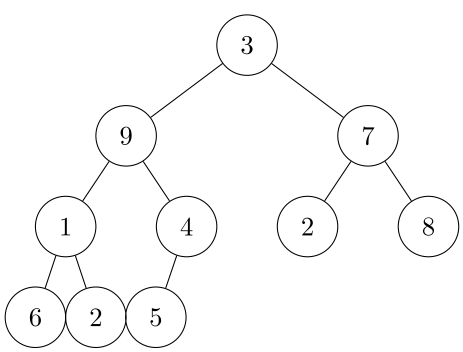
(b)

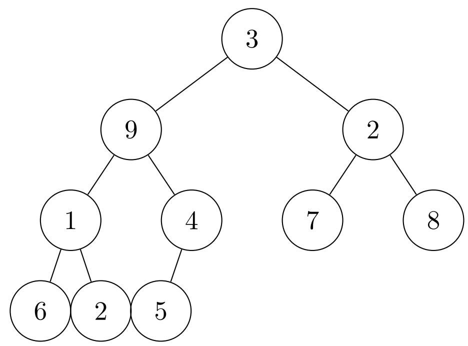

2
3 4
5 4 9 5
6

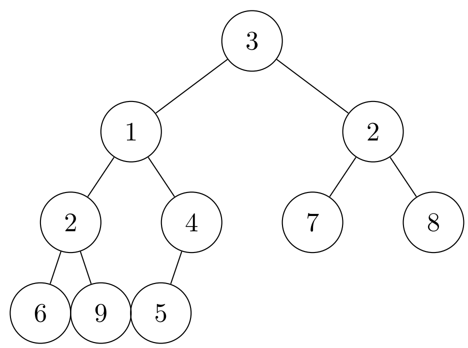
Ex. 8

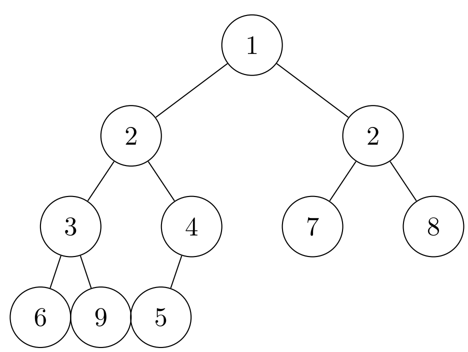

3
9 7
2 5 2 8
6 1 4
1.


3
9 7
2 4 2 8
6 1 5
2.


3
9 7
1 4 2 8
6 2 5
3.


3
9 2
1 4 7 8
6 2 5
4.


3
1 2
2 4 7 8
6 9 5
5.


1
2 2
3 4 7 8
6 9 5
6.