Problème 1.

Consider the sequence of values $S = [3, 42, 39, 86, 49, 89, 99, 20, 88, 51, 64]$

1.1

Draw a left-leaning red-black tree obtained by adding the values in S in sequence. Show each step.

Let the following format as node: `[R|B]-<value>` where `[R|B]` denotes whether the node is red or black, followed by its value.

The left-leaning red-black tree obtained by adding the values in S in sequence is as follows:

1. $S = [3]$

```
B-3
```

2. $S = [3, 42]$ New node become red, 42

```
B-3
  \
   R-42
```

3. $S = [3, 42, 39]$ New root 39, rotate color for 3, 42

```
  B-39
 /     \
R-3    R-42
```

4. $S = [3, 42, 39, 86]$ Add 86, rotate color for tree

```
  B-39
 /     \
R-3    R-42
          \
           R-86
```

5. $S = [3, 42, 39, 86, 49]$ Add 49, rotate color for tree 49, 42

```
  B-39
 /    \
R-3   R-49
      /  \
    R-42   R-86
```

6. $S = [3, 42, 39, 86, 49, 89]$ Add 89, rotate color for 42, 86

```
  B-39
 /    \
R-3   R-49
      /  \
    B-42  B-86
            \
           R-89
```

7. $S = [3, 42, 39, 86, 49, 89, 99]$ Add 99, rotate color for 86, 89

```
  B-39
 /    \
R-3   R-49
      /  \
    B-42  B-89
          /  \
        R-86 R-89
```

8. $S = [3, 42, 39, 86, 49, 89, 99, 20]$ Add 20, right of 3, red. Rotate color of 3

```
      B-39
     /    \
    B-3   R-49
     |    |   \
    R-20 B-42 B-89
              /  \
            R-86 R-89
```

9. $S = [3, 42, 39, 86, 49, 89, 99, 20, 88]$ Add 88, correct root of 49, balance tree to L-39. R-89

```
      B-49
     /    \
    R-39   R-89
    /  |    |   \
```

```
  B-3 R-42 B-86 B-99
   |         |
  R-20      R-88
```

10. $S = [3, 42, 39, 86, 49, 89, 99, 20, 88, 51]$ Add 51, left of 86

```
      B-49
     /    \
    R-39   R-89
    /  |     |   \
  B-3 R-42 B-86 B-99
   |        /  \
R-20      R-51 R-88
```

11. $S = [3, 42, 39, 86, 49, 89, 99, 20, 88, 51, 64]$ Add 64, 86 comes red, 64 becomes right of 51, rotate color 51, 88, 86

```
      B-49
     /    \
    R-39   R-89
    /  |     |   \
  B-3 R-42 R-86 B-99
   |        /  \
R-20      B-51 B-88
            |
          R-64
```

1.2

Consider the hash function $h(x) = (x + 7) \bmod 13$ a hash-table of 13 table entries that uses hashing with separate chaining. Draw the hash-table obtained by adding the values in S in sequence. Show each step.

The hash-table obtained by adding the values in S in sequence is as follows:

1. $S = [3]$ $h(3) = 10 \bmod 13 = 10$

```
0:
1:
2:
3:
4:
5:
6:
```

```
7:
8:
9:
10: 3
11:
12:
```

2. $S = [3, 42]$ $h(42) = 49 \bmod 13 = 10$ Collision with 3, chaining with 3

```
0:
1:
2:
3:
4:
5:
6:
7:
8:
9:
10: 3 ⟶ 42
11:
12:
```

3. $S = [3, 42, 39]$ $h(39) = 46 \bmod 13 = 7$

```
0:
1:
2:
3:
4:
5:
6:
7: 39
8:
9:
10: 3 ⟶ 42
11:
12:
```

4. $S = [3, 42, 39, 86]$ $h(86) = 93 \bmod 13 = 2$

```
0:
1:
2: 86
3:
4:
5:
6:
7: 39
```

```
8:
9:
10: 3 → 42
11:
12:
```

5. $S = [3, 42, 39, 86, 49]$ $h(49) = 56 \bmod 13 = 4$

```
0:
1:
2: 86
3:
4: 49
5:
6:
7: 39
8:
9:
10: 3 → 42
11:
12:
```

6. $S = [3, 42, 39, 86, 49, 89]$ $h(89) = 96 \bmod 13 = 5$

```
0:
1:
2: 86
3:
4: 49
5: 89
6:
7: 39
8:
9:
10: 3 → 42
11:
12:
```

7. $S = [3, 42, 39, 86, 49, 89, 99]$ $h(99) = 106 \bmod 13 = 2$ Collide with 86, chaining with 86

```
0:
1:
2: 86 → 99
3:
4: 49
5: 89
6:
7: 39
8:
```

```
9:
10: 3 ⟶ 42
11:
12:
```

8. $S = [3, 42, 39, 86, 49, 89, 99, 20]$ $h(20) = 27 \text{ mod } 13 = 1$

```
0:
1: 27
2: 86 ⟶ 99
3:
4: 49
5: 89
6:
7: 39
8:
9:
10: 3 ⟶ 42
11:
12:
```

9. $S = [3, 42, 39, 86, 49, 89, 99, 20, 88]$ $h(88) = 95 \text{ mod } 13 = 4$ Collide with 49, chaining with 49

```
0:
1: 27
2: 86 ⟶ 99
3:
4: 49 ⟶ 88
5: 89
6:
7: 39
8:
9:
10: 3 ⟶ 42
11:
12:
```

10. $S = [3, 42, 39, 86, 49, 89, 99, 20, 88, 51]$ $h(51) = 58 \text{ mod } 13 = 6$

```
0:
1: 27
2: 86 ⟶ 99
3:
4: 49 ⟶ 88
5: 89
6: 51
7: 39
8:
```

```
9:
10: 3 ⟶ 42
11:
12:
```

11. $S = [3, 42, 39, 86, 49, 89, 99, 20, 88, 51, 64]$ $h(64) = 71 \bmod 13 = 6$ Collide with 51, chaining with 51

```
0:
1: 27
2: 86 ⟶ 99
3:
4: 49 ⟶ 88
5: 89
6: 51 ⟶ 64
7: 39
8:
9:
10: 3 ⟶ 42
11:
12:
```

> ⓘ 1.3
>
> Consider the hash function $h(x) = (x + 7) \bmod 13$ a hash-table of 13 table entries that uses hashing with linear probing. Draw the hash-table obtained by adding the values in S in sequence. Show each step.

1. $S = [3]$ $h(3) = 10 \bmod 13 = 10$

```
0:
1:
2:
3:
4:
5:
6:
7:
8:
9:
10: 3
11:
12:
```

2. $S = [3, 42]$ $h(42) = 49 \bmod 13 = 10$ Collision with 3, increment index to 11

```
0:
1:
2:
3:
4:
5:
6:
7:
8:
9:
10: 3
11: 42
12:
```

3. $S = [3, 42, 39]$ $h(39) = 46 \bmod 13 = 7$

```
0:
1:
2:
3:
4:
5:
6:
7: 39
8:
9:
10: 3
11: 42
12:
```

4. $S = [3, 42, 39, 86]$ $h(86) = 93 \bmod 13 = 2$

```
0:
1:
2: 86
3:
4:
5:
6:
7: 39
8:
9:
10: 3
11: 42
12:
```

5. $S = [3, 42, 39, 86, 49]$ $h(49) = 56 \bmod 13 = 4$

```
0:
1:
2: 86
3:
4: 49
5:
6:
7: 39
8:
9:
10: 3
11: 42
12:
```

6. $S = [3, 42, 39, 86, 49, 89]$ $h(89) = 96 \bmod 13 = 5$

```
0:
1:
2: 86
3:
4: 49
5: 89
6:
7: 39
8:
9:
10: 3
11: 42
12:
```

7. $S = [3, 42, 39, 86, 49, 89, 99]$ $h(99) = 106 \bmod 13 = 2$ Collide with 86, increment index to 3

```
0:
1:
2: 86
3: 99
4: 49
5: 89
6:
7: 39
8:
9:
10: 3
11: 42
12:
```

8. $S = [3, 42, 39, 86, 49, 89, 99, 20]$ $h(20) = 27 \bmod 13 = 1$

```
0:
1: 27
2: 86
3: 99
4: 49
5: 89
6:
7: 39
8:
9:
10: 3
11: 42
12:
```

9. $S = [3, 42, 39, 86, 49, 89, 99, 20, 88]$ $h(88) = 95 \bmod 13 = 4$ Collide with 49, index to 5 at 5, collide with 89, index to 6

```
0:
1: 27
2: 86
3: 99
4: 49
5: 89
6: 88
7: 39
8:
9:
10: 3
11: 42
12:
```

10. $S = [3, 42, 39, 86, 49, 89, 99, 20, 88, 51]$ $h(51) = 58 \bmod 13 = 6$ Collide with 88, index to 7 at 7, collide with 39, index to 8

```
0:
1: 27
2: 86
3: 99
4: 49
5: 89
6: 88
7: 39
8: 51
9:
10: 3
11: 42
12:
```

11. $S = [3, 42, 39, 86, 49, 89, 99, 20, 88, 51, 64]$ $h(64) = 71 \text{ mod } 13 = 6$ Collide with 88, index to 7 at 7, collide with 39, index to 8 at 8, collide with 51, index to 9

```
0:
1: 27
2: 86
3: 99
4: 49
5: 89
6: 88
7: 39
8: 51
9: 64
10: 3
11: 42
12:
```

Problème 2.

Consider a list L of N sorted values. Show how to construct a valid left-leaning red-black tree holding the values in L in $\Theta(N)$

Solution

The following depicts the pseudocode implementation of the program

Algorithm BuildLLRBTree(L, start, end)

```
if start > end then
    return NULL
end if
mid ← (start + end)/2
left ← BuildLLRBTree(L, start, mid − 1)
right ← BuildLLRBTree(L, mid + 1, end)
node ← new Node(L[mid])
node.left ← left
node.right ← right
node.color ← BLACK
return node
```

Problème 3.

Consider a set of strings S. We want to figure out whether S has duplicates efficiently. We do not want to do so by sorting S and then checking for duplicates: comparing strings can be a lot of work (e.g., they might differ in only a single character). Assume that you have a hash function h that can compute a suitable hash code for any string $s \in S$ in $\mathcal{O}(|s|)$. Show how one can use hashing to find whether S has duplicates without performing many comparisons between strings. Your algorithm should have an expected runtime of $\mathcal{O}(|S|)$ in which $|S| = \sum_{s \in S} |s|$ represents the total length of all strings in S.

Solution

Algorithm Check for Duplicates Using Hashing

Require: A set of strings S

Ensure: True if there are duplicates in S, False otherwise

```
Initialize an empty hash table H
for each string s ∈ S do
    Compute h(s) using the hash function
    if h(s) is in H then
        for each string s′ ∈ H[h(s)] do
            if s = s′ then
                return True
            end if
        end for
        Append s to H[h(s)]
    else
        Insert s into H at h(s)
    end if
end for
return False
```