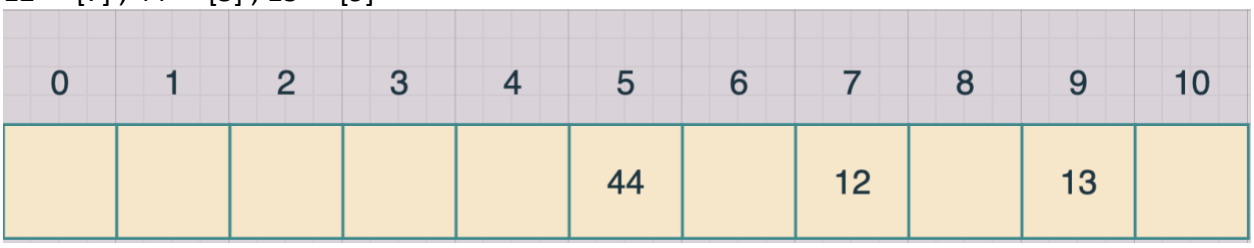
Task 1:
Desirable properties of a hash function:
- Easy to compute from k to h(k), but impossible to trace from h(k) back to k
- Minize the number of collisions caused by duplicate values

Task 2: h(k) = (2k + 5) mod 11

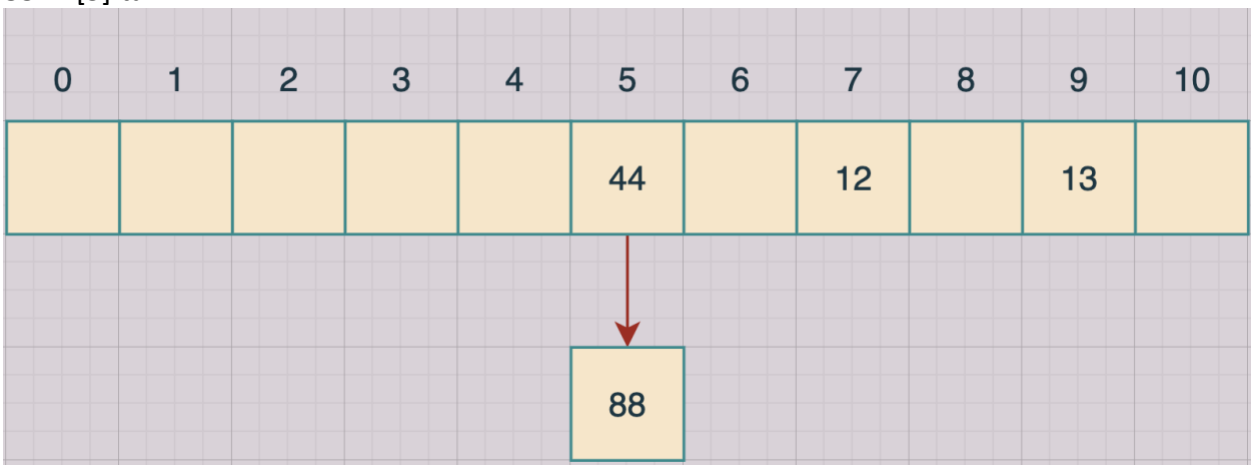
k	12	44	13	88	23	94	11	39	20	16	5
h(k)	7	5	9	5	7	6	5	6	1	4	4

a)

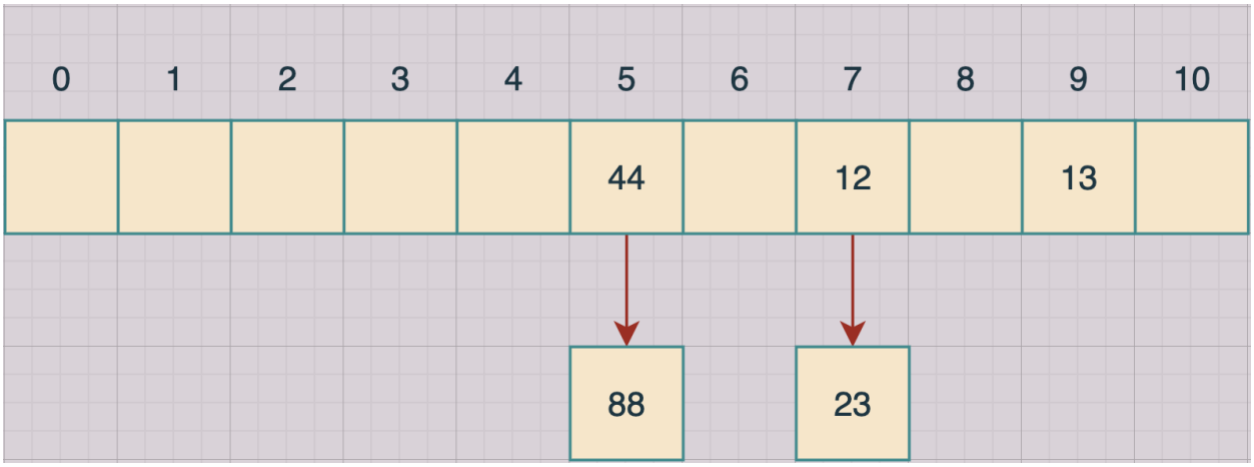
12 -> [7] ; 44 -> [5] ; 13 -> [9]



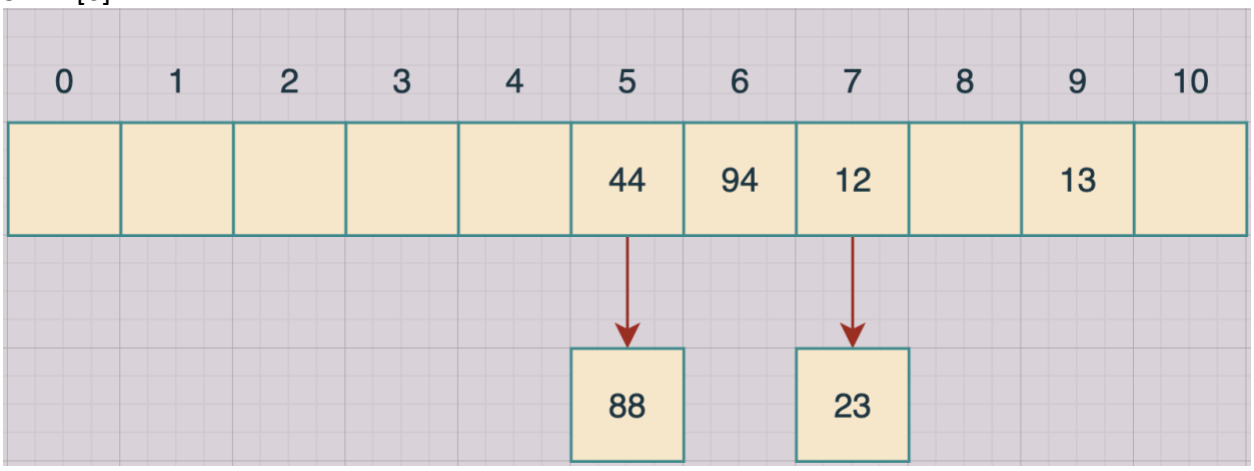
88 -> [5] => collision
88 -> [5].tail



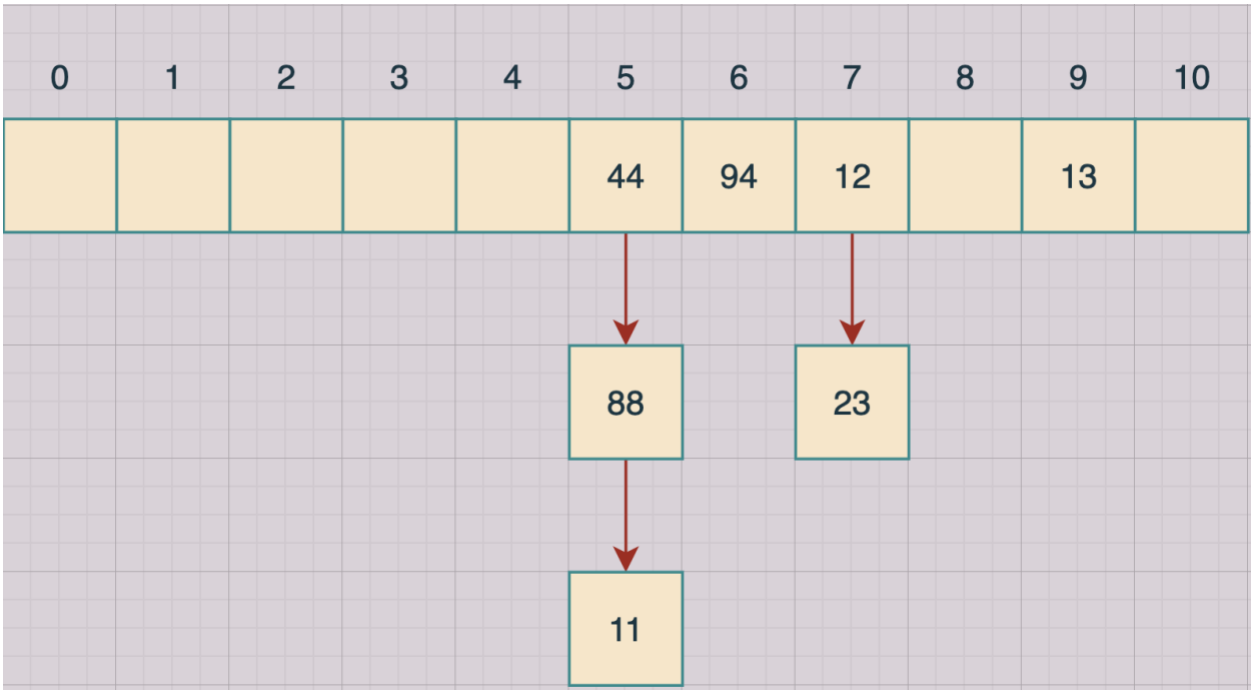
23 -> [7] => collision
23 -> [7].tail



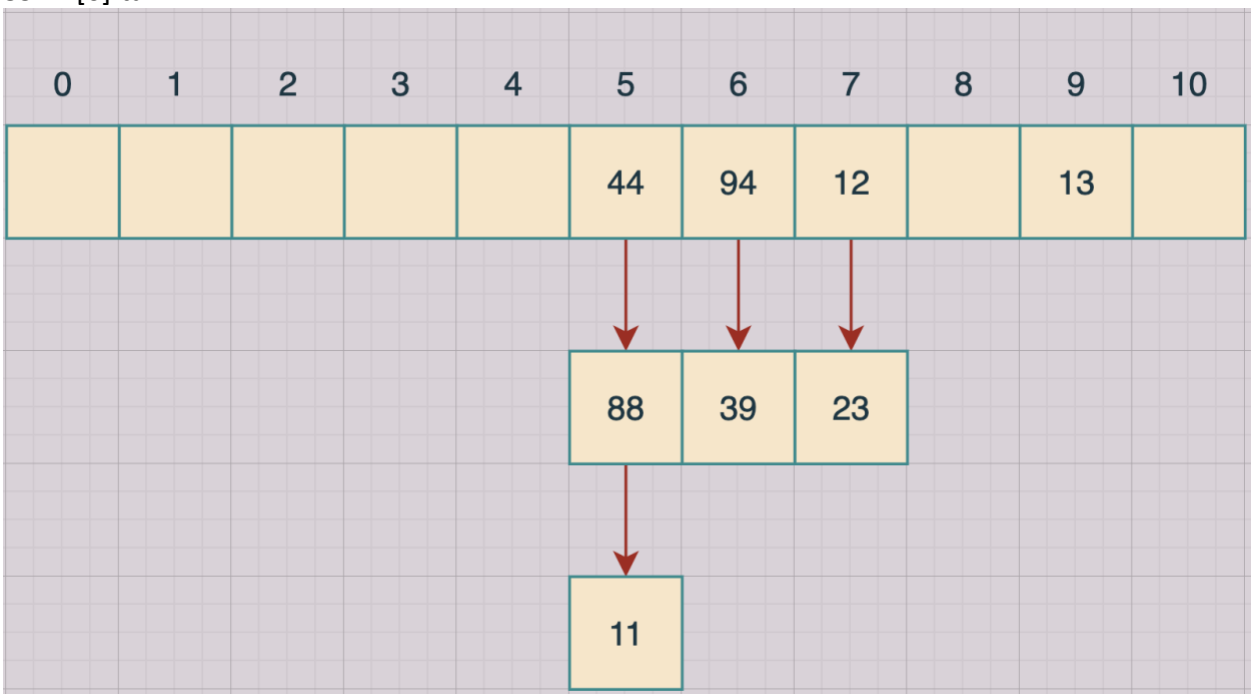
94 -> [6]



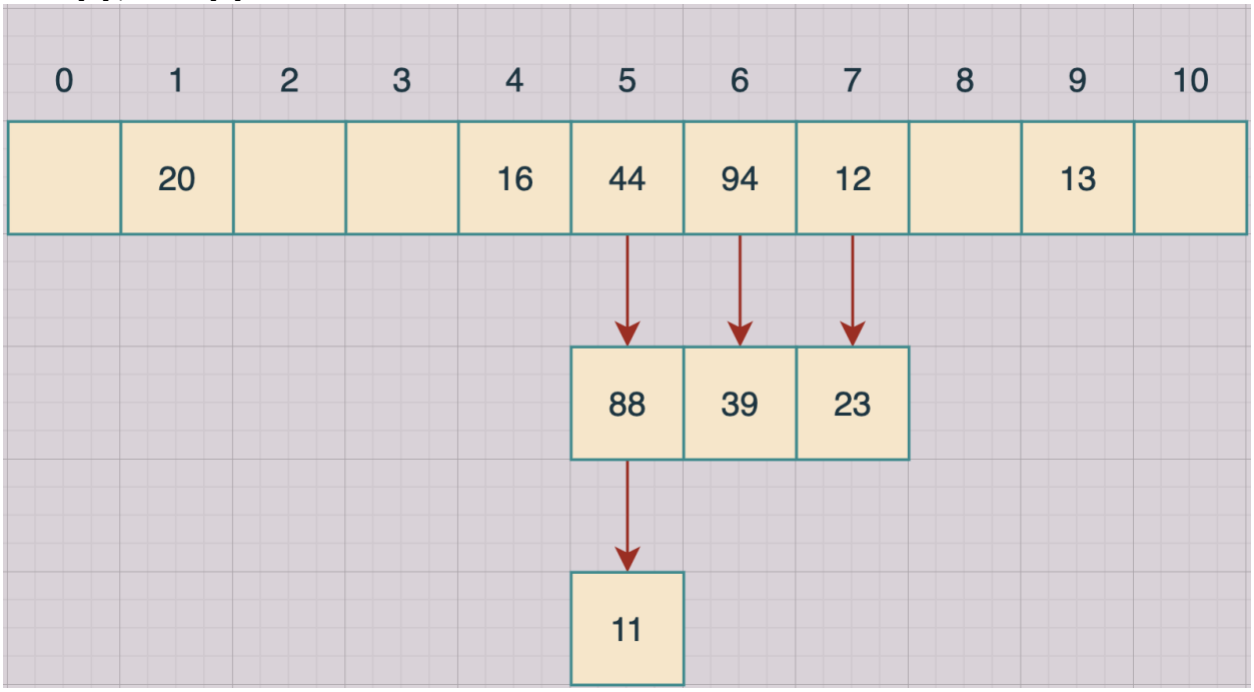
11 -> [5] => collision
11 -> [5].tail



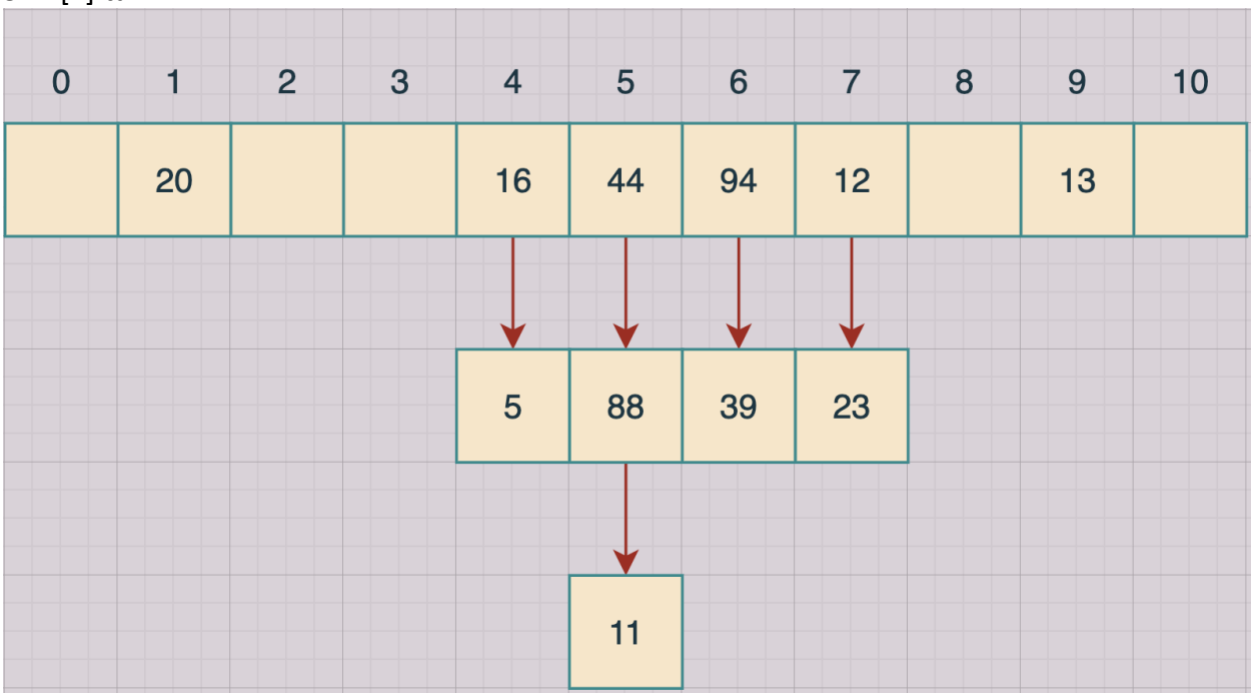
39 -> [6] => collision
39 -> [6].tail



20 -> [1] ; 16 -> [4]



5 -> [4] => collision
5 -> [4].tail



b)

12 -> [7] ; 44 -> [5] ; 13 -> [9]

0	1	2	3	4	5	6	7	8	9	10
					44		12		13	

88 -> [5] => collision

88 -> [6]

0	1	2	3	4	5	6	7	8	9	10
					44	88	12		13	

23 -> [7] => collision

23 -> [8]

0	1	2	3	4	5	6	7	8	9	10
					44	88	12	23	13	

94 -> [6] => collision

94 -> [7] => collision

94 -> [8] => collision

94 -> [9] => collision

94 -> [10]

0	1	2	3	4	5	6	7	8	9	10
					44	88	12	23	13	94

11 -> [5] => collision

11 -> [6] => collision

11 -> [7] => collision

11 -> [8] => collision

11 -> [9] => collision

11 -> [10] => collision

11 -> [0]

0	1	2	3	4	5	6	7	8	9	10
11					44	88	12	23	13	94

39 -> [6] => collision

39 -> [7] => collision

39 -> [8] => collision

39 -> [9] => collision

39 -> [10] => collision

39 -> [0] => collision

39 -> [1]

0	1	2	3	4	5	6	7	8	9	10
11	39				44	88	12	23	13	94

20 -> [1] => collison
20 -> [2]

0	1	2	3	4	5	6	7	8	9	10
11	39	20			44	88	12	23	13	94

16 -> [4]

0	1	2	3	4	5	6	7	8	9	10
11	39	20		16	44	88	12	23	13	94

5 -> [4] => collision
5 -> [5] => collision
5 -> [6] => collision
5 -> [7] => collision
5 -> [8] => collision
5 -> [9] => collision
5 -> [10] => collision
5 -> [0] => collision
5 -> [1] => collision
5 -> [2] => collision
5 -> [3]

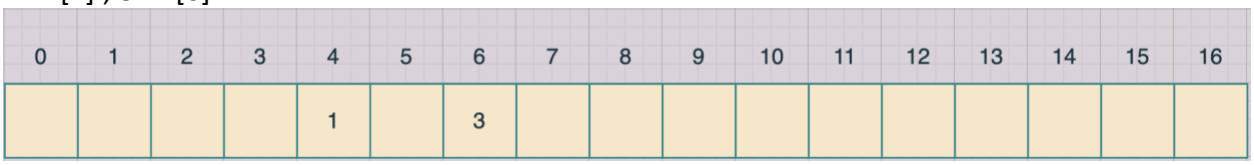
0	1	2	3	4	5	6	7	8	9	10
11	39	20	5	16	44	88	12	23	13	94

Task 3: h(k) = (k + 3) mod 17

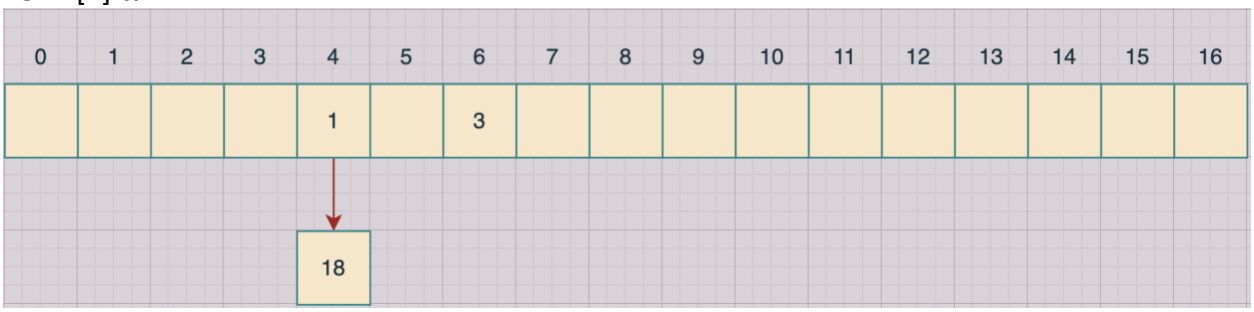
k	1	3	18	8	23	35	11	36	20	16
h(k)	4	6	4	11	9	4	14	5	6	2

a)

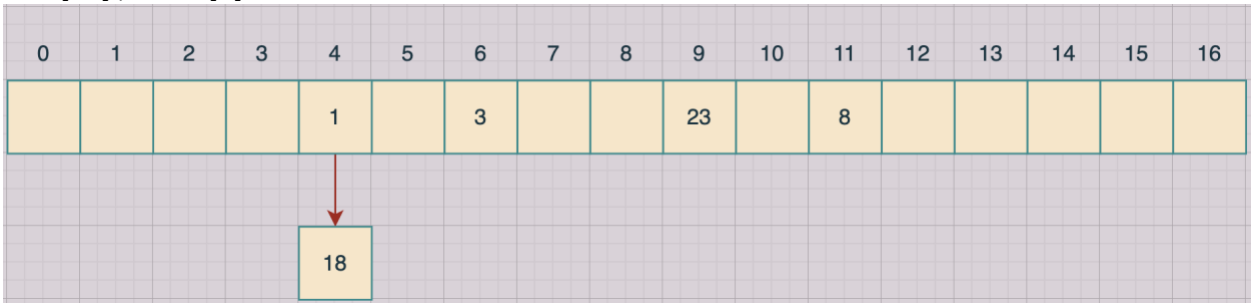
1 -> [4] ; 3 -> [6]



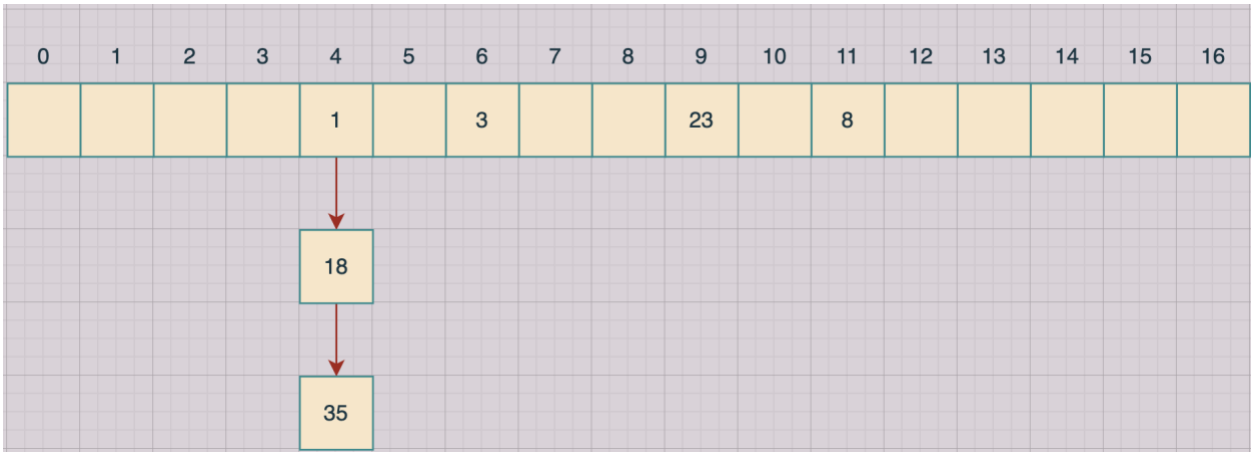
18 -> [4] => collision
18 -> [4].tail



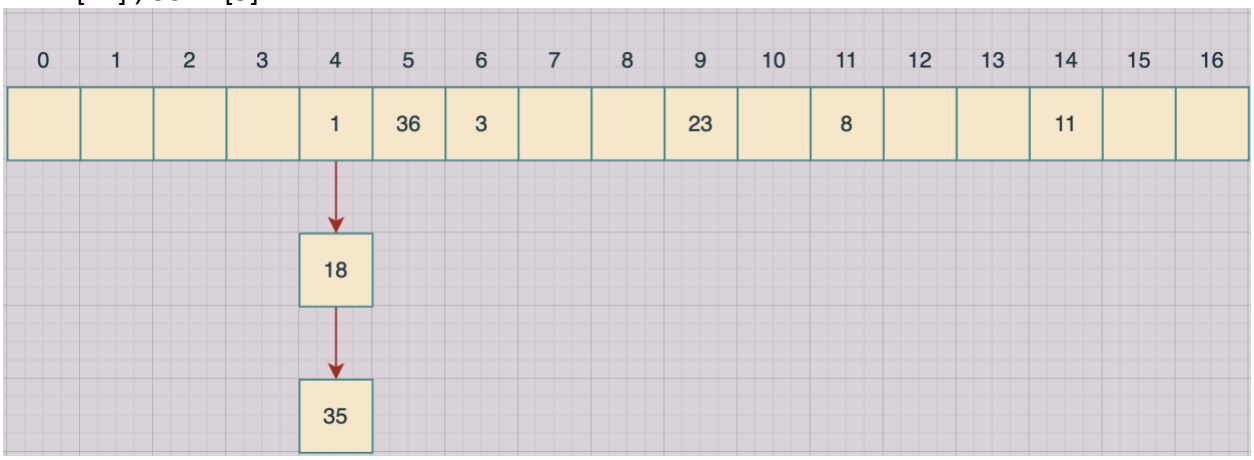
8 -> [11] ; 23 -> [9]



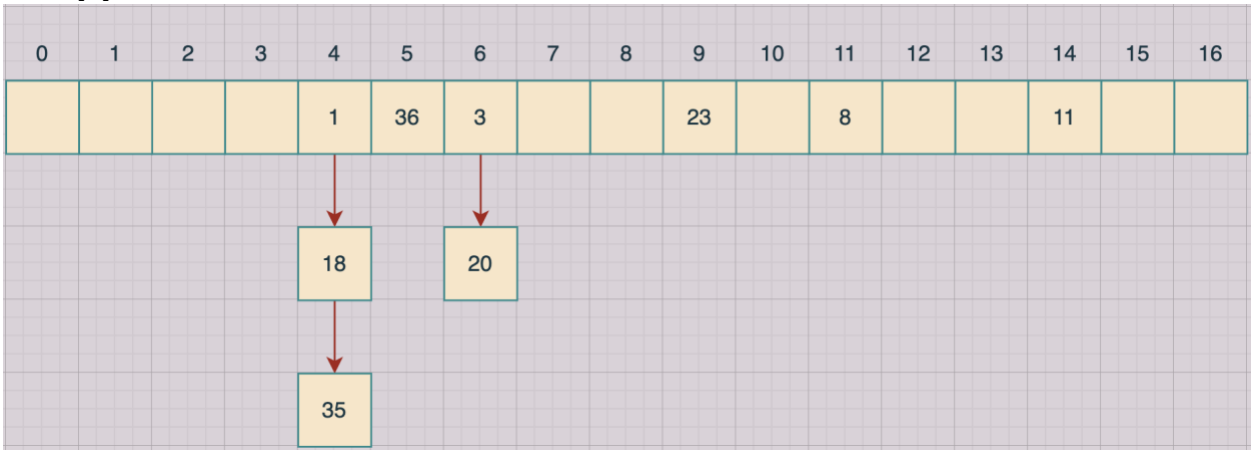
35 -> [4] => collision
35 -> [4].tail



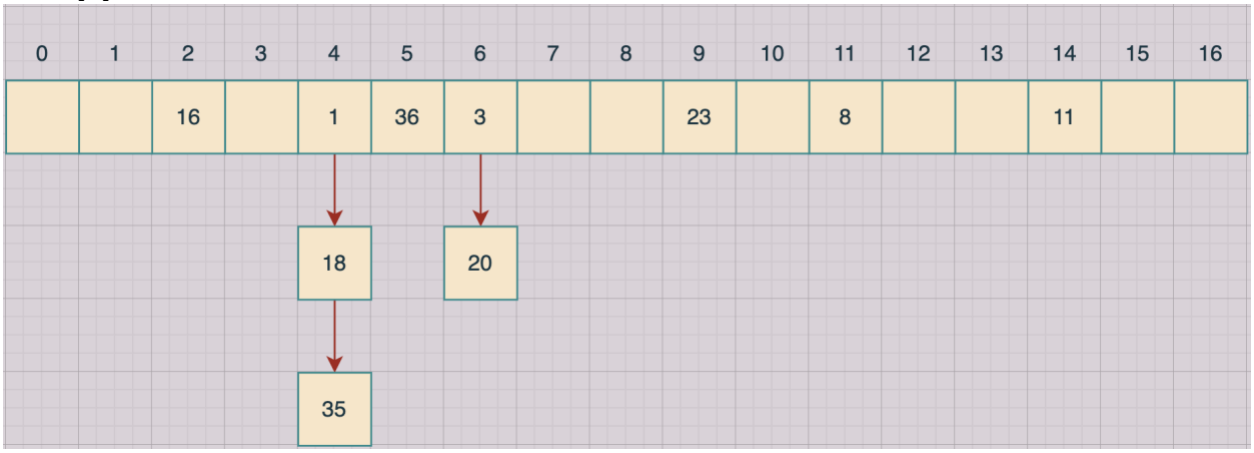
11 -> [14] ; 36 -> [5]



20 -> [6] => collision
20 -> [6].tail



16 -> [2]



b)

1 -> [4] ; 3 -> [6]

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
				1		3										

18 -> [4] => collision

18 -> [5]

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
				1	18	3										

8 -> [11] ; 23 -> [9]

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
				1	18	3			23		8					

35 -> [4] => collision

35 -> [5] => collision

35 -> [6] => collision

35 -> [7]

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
				1	18	3	35		23		8					

11 -> [14]

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
				1	18	3	35		23		8			11		

36 -> [5] => collision

36 -> [6] => collision

36 -> [7] => collision

36 -> [8]

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
				1	18	3	35	36	23		8			11		

20 -> [6] => collision
20 -> [7] => collision
20 -> [8] => collision
20 -> [9] => collision
20 -> [10]

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
				1	18	3	35	36	23	20	8			11		

16 -> [2]

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
		16		1	18	3	35	36	23	20	8			11		