Competitive Algorithm Design and Practice
Longest Increasing Sub-sequence

2018/03/28

Lin Yun Wen
l40303k@gmail.com

Department of Computer Science and Information Engineering
National Cheng Kung University
Tainan, Taiwan

Made By mike199250

Longest Increasing Sub-sequence

Made By mike199250

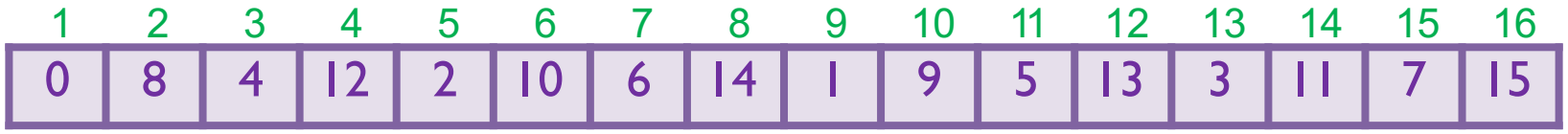
LIS

- Find a sub-sequence of a given sequence in which the sub-sequence`s elements are in sorted order, lowest to highest.

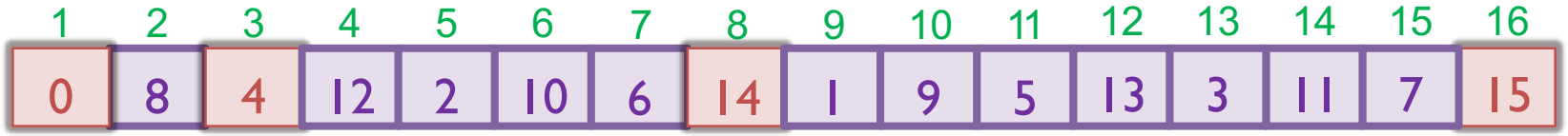
- And the sub-sequence is as long as possible.

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

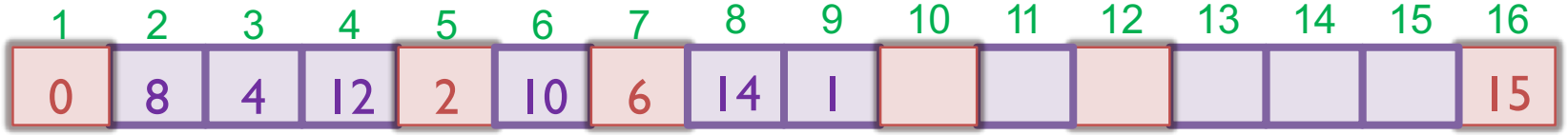
LIS



- An increasing sub-sequence:



- The longest one:

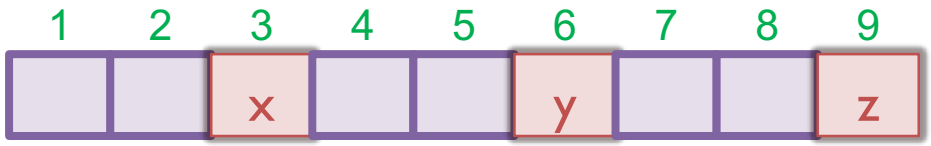


LIS

- Naïve solutions:
  - For every sub-sequence, check if it is LIS.
- Time-complexity:
  - Every sub-sequence, $O(2^N)$.
  - For each sub-sequence checking needs $O(N)$.
  - Total: $O(N*2^N)$

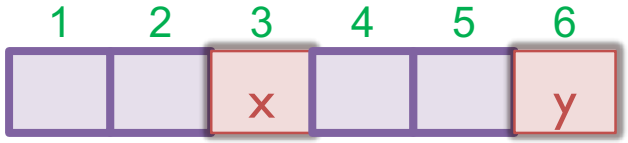
LIS

- Observation:
- If



- is LIS ended at z,

- then



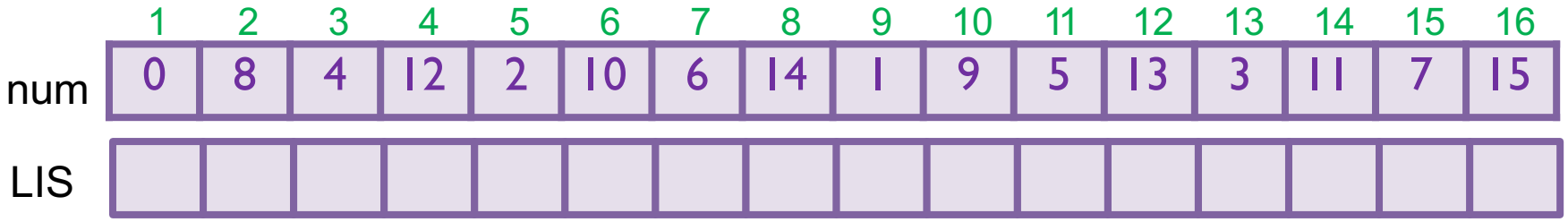
- must be LIS ended at y

LIS

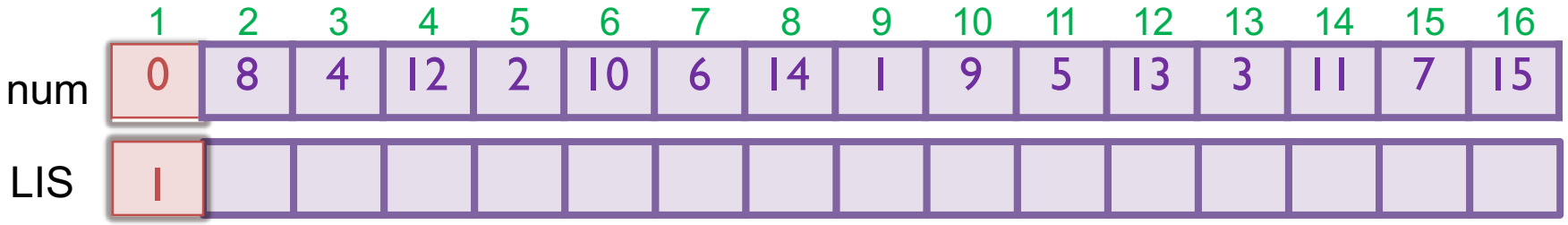
- What do we want to know?
  - The LIS ended at index k, i.e. LIS[k]
- How can we get that?
  - Find the previous number with longest LIS.
  - LIS[k] =

    max(1, LIS[i]+1) for i<k & num[i] < num[k]

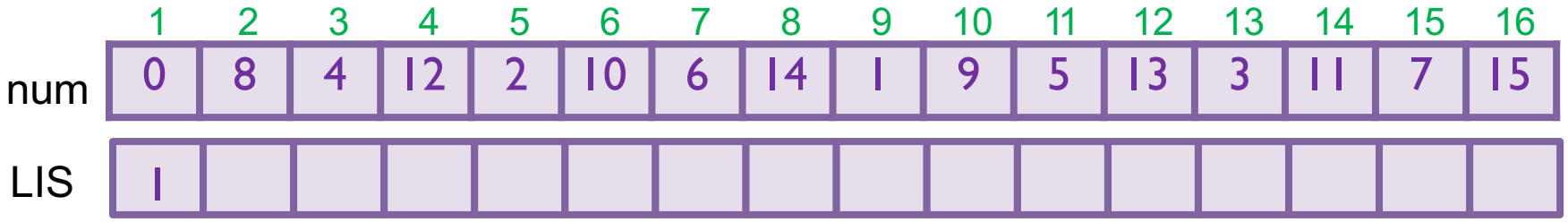
LIS


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

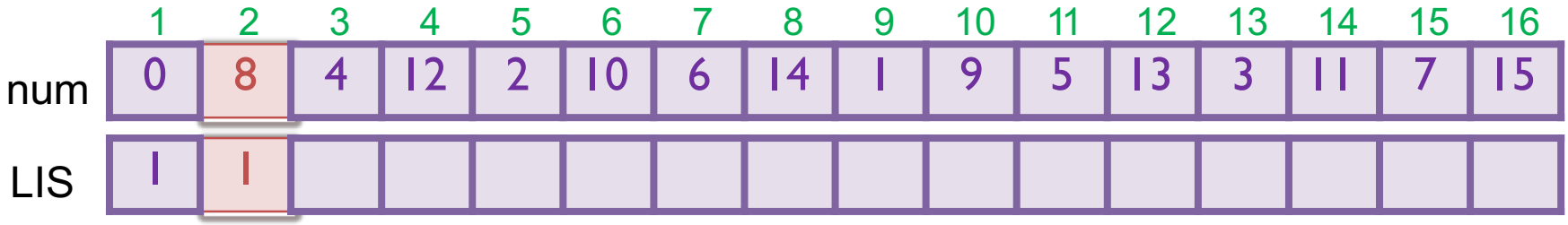
LIS


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

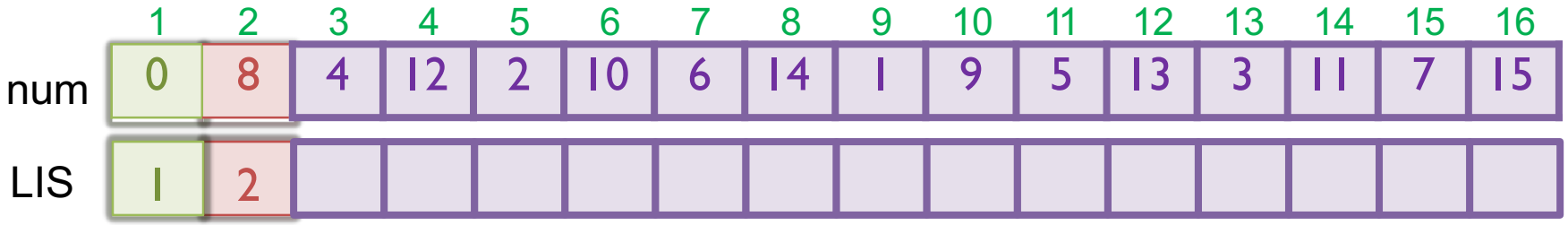
LIS


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

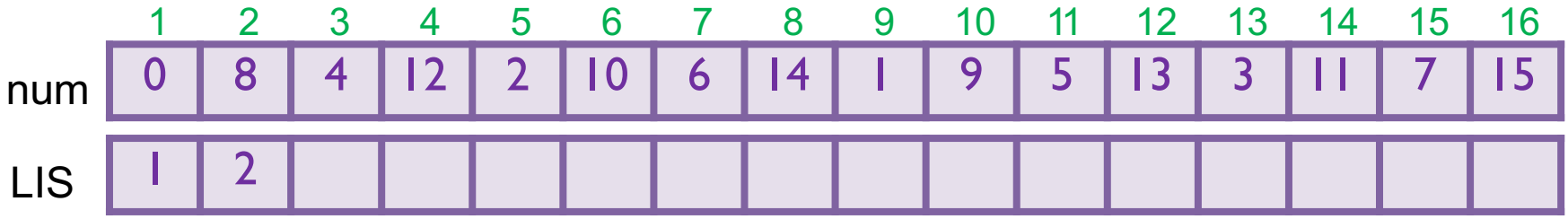
LIS


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

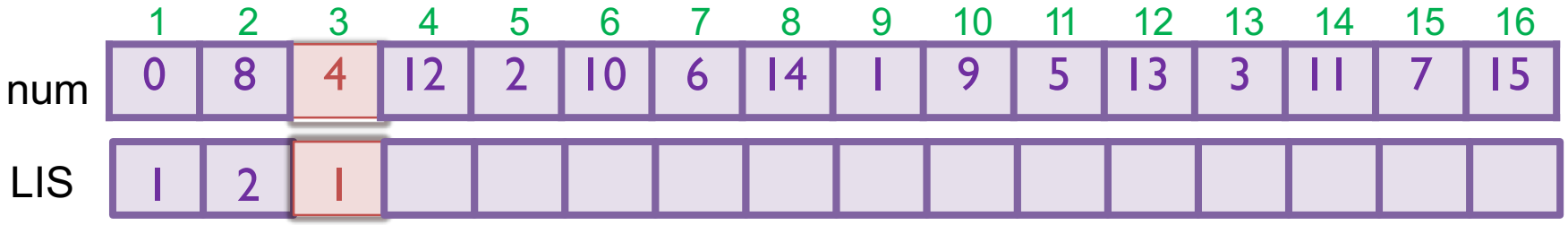
LIS


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

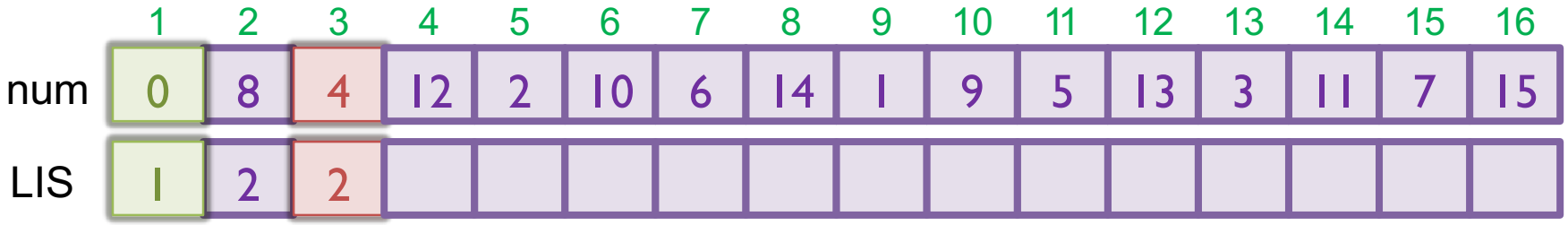
LIS


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

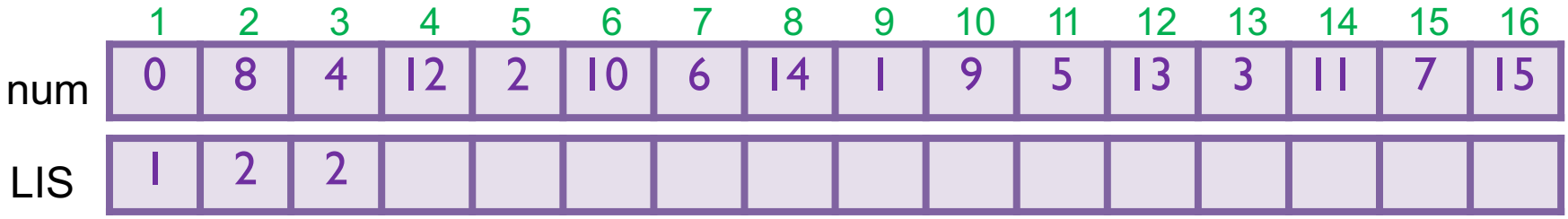
LIS


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

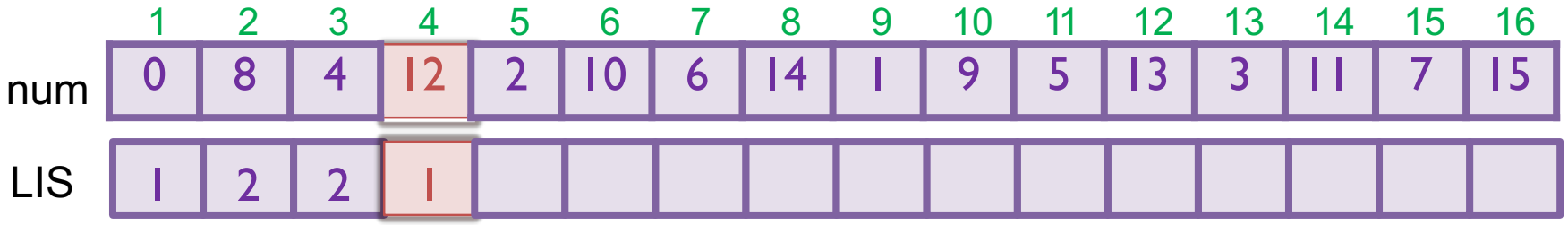
LIS


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

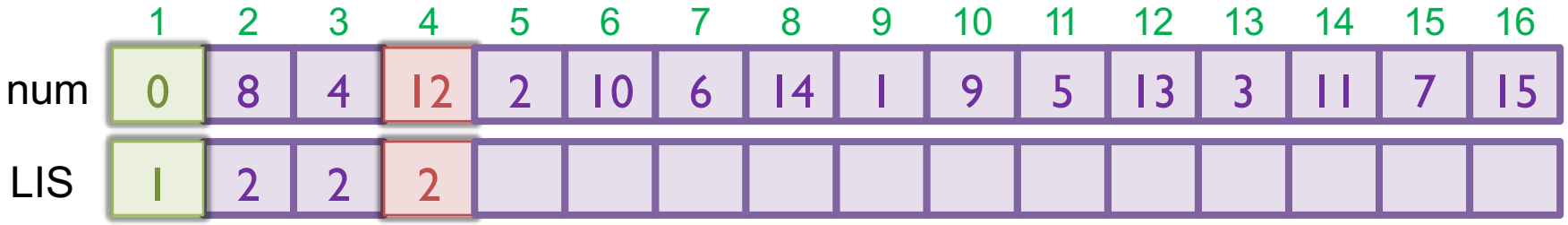
LIS


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

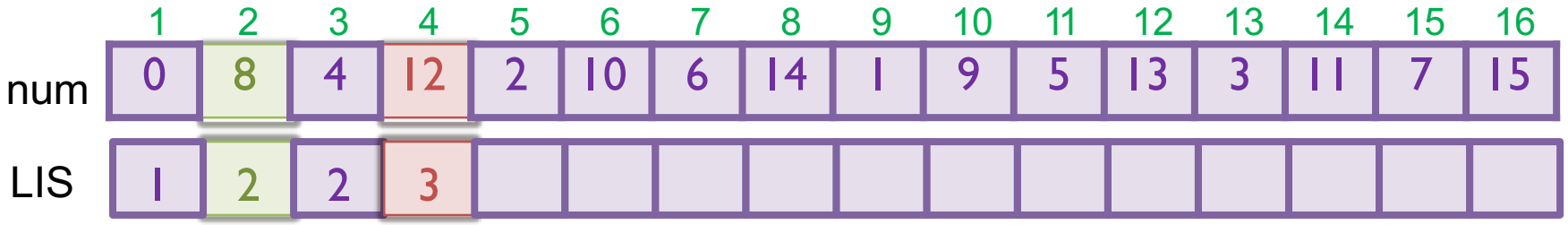
LIS


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

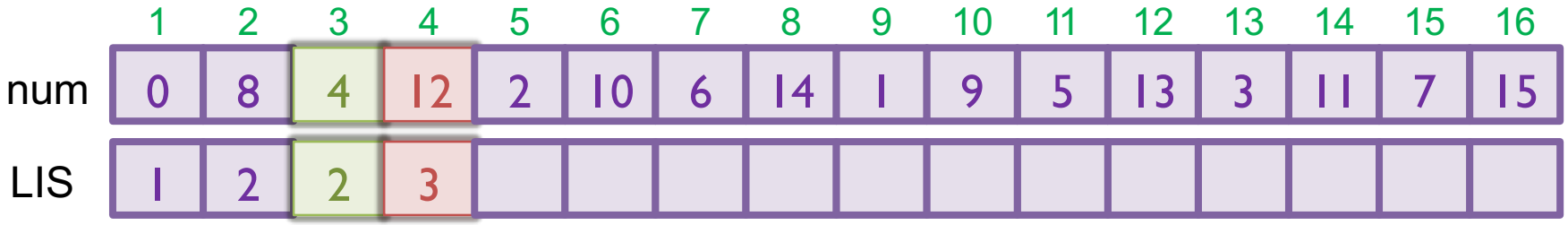
LIS


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

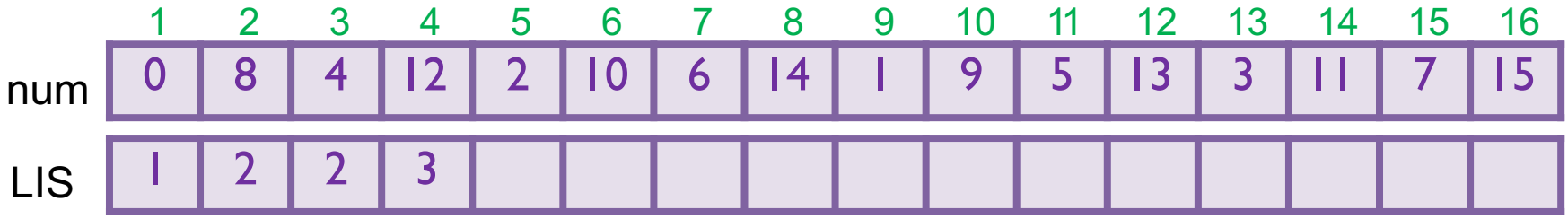
LIS


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

LIS


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

LIS


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

LIS

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

LIS

- Time-complexity:
  - For each element find the previous longest LIS.
  - Every element, O(N)
  - Each element, check previous elements, O(N)
  - Total: O(N*N) => $O(N^2)$
- Space-complexity:
  - Additional LIS array, O(N)

LIS

```c
/* file name: LIS.c */
#include <stdio.h>

int num[17]={0,0,8,4,12,2,10,6,14,1,9,5,13,3,11,7,15};
int LIS[17];
void Find_LIS()
{
    int i,j;
    for(i=1;i<=16;i++)
    {
        LIS[i]=1;
        for(j=1;j<i;j++)
            if(num[j]<num[i] && LIS[j]+1>LIS[i])
                LIS[i]=LIS[j]+1;
    }
}
int main()
{
    int i;
    Find_LIS();
    printf("num:");
    for(i=1;i<=16;i++)printf("%3d",num[i]);
    printf("\nLIS:");
    for(i=1;i<=16;i++)printf("%3d",LIS[i]);
    putchar('\n');
    return 0;
}
```

```
num:  0  8  4 12  2 10  6 14  1  9  5 13  3 11  7 15
LIS:  1  2  2  3  2  3  3  4  2  4  3  5  3  5  4  6

Process returned 0 (0x0)   execution time : 0.049 s
Press any key to continue.
```

Made By mike199250

POJ 2533

Made By mike199250

Learn more!

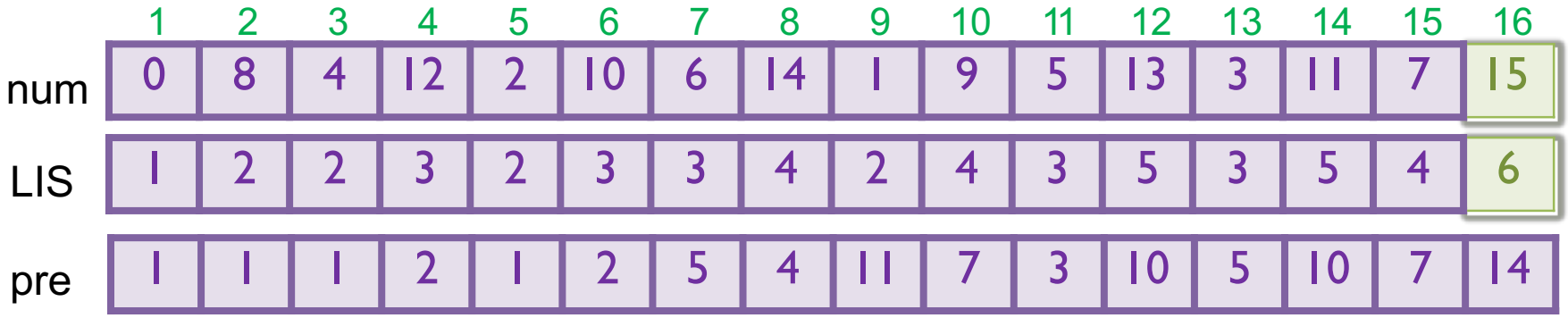
- Sometimes we need to output a solution too, but how?

- 1. Find the number backwards.
  - Assume ans=length of LIS
  - Find k such that LIS[k]=ans.
  - Then from index 1~k find i such that LIS[i] = LIS[k]-1 && num[i] < num[k], and so on
- 2. Additional array pre[].
  - pre[k] = i such that LIS[i] = LIS[k]-1 && num[i] < num[k]

Learn more!

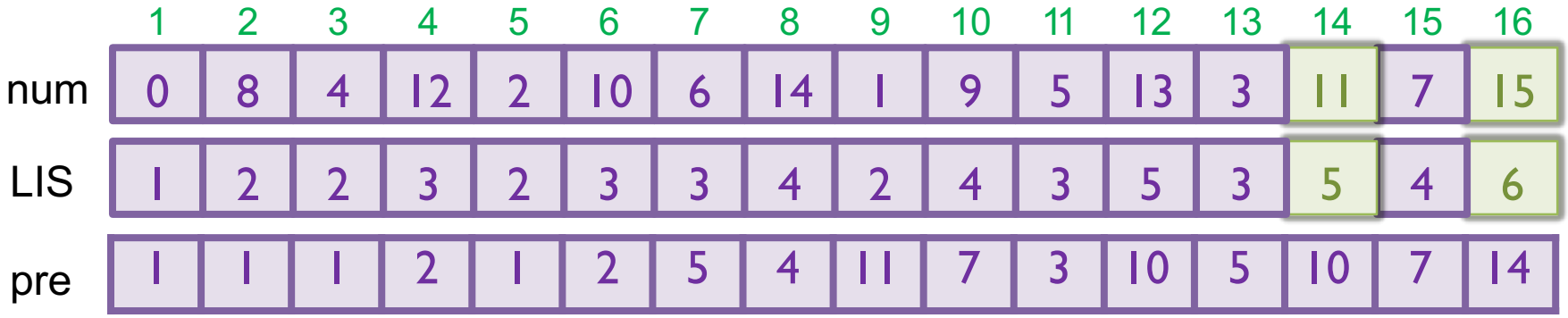
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
num	0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15
LIS	1	2	2	3	2	3	3	4	2	4	3	5	3	5	4	6
pre	1	1	1	2	1	2	5	4	11	7	3	10	5	10	7	14

Learn more!



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

Learn more!

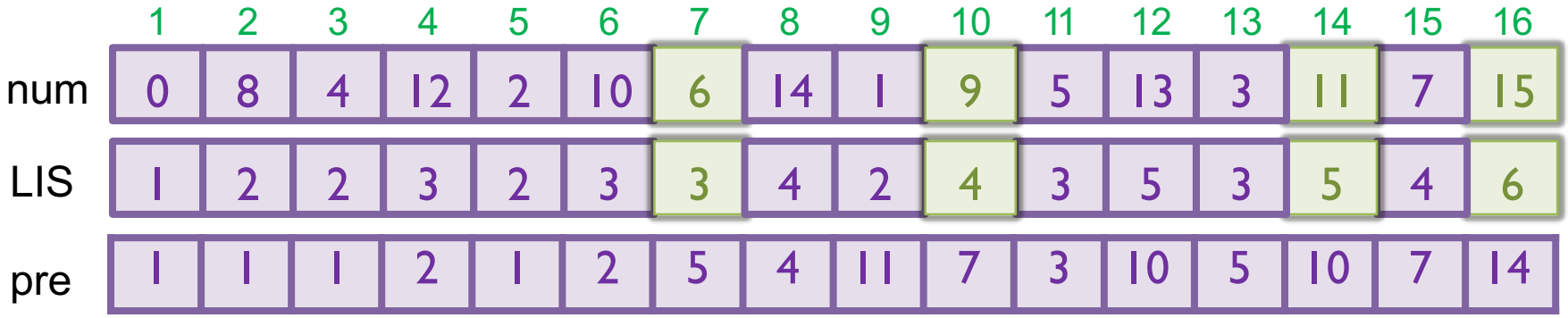


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
num	0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15
LIS	1	2	2	3	2	3	3	4	2	4	3	5	3	5	4	6
pre	1	1	1	2	1	2	5	4	11	7	3	10	5	10	7	14

Learn more!

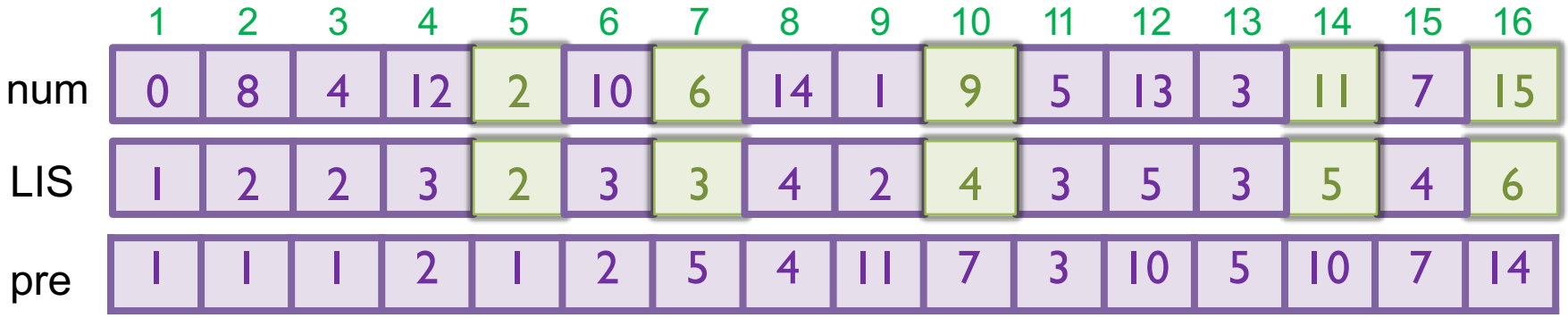
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
num	0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15
LIS	1	2	2	3	2	3	3	4	2	4	3	5	3	5	4	6
pre	1	1	1	2	1	2	5	4	11	7	3	10	5	10	7	14

Learn more!



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

Learn more!



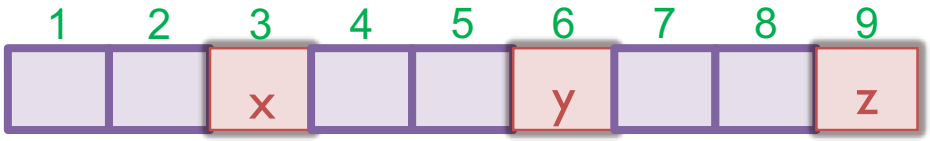
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
num	0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15
LIS	1	2	2	3	2	3	3	4	2	4	3	5	3	5	4	6
pre	1	1	1	2	1	2	5	4	11	7	3	10	5	10	7	14

Learn more!

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

Learn more!

- Sometimes $O(N^2)$ is too slow……



- If LIS[x] = LIS[y](mean-while x>=y)
- For some z that LIS[z] = LIS[x]+1
  We can append z after either x or y to form LIS, by the way y might be smaller!

- That is, for LIS[] = k we can memorize just one number, the smaller one, y.

Learn more!

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

Learn more!

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

LIS =1 | 0 |

Learn more!

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

LIS =2 | 8

LIS =1 | 0

Made By mike199250

Learn more!

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

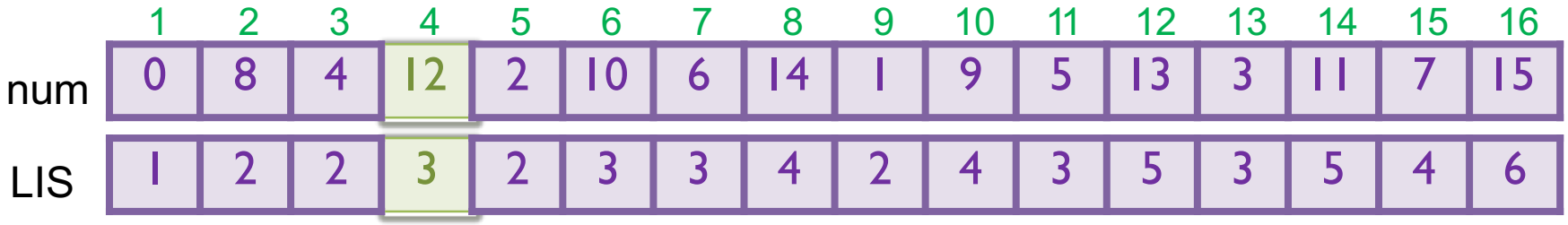
LIS =2

LIS =1

4
0

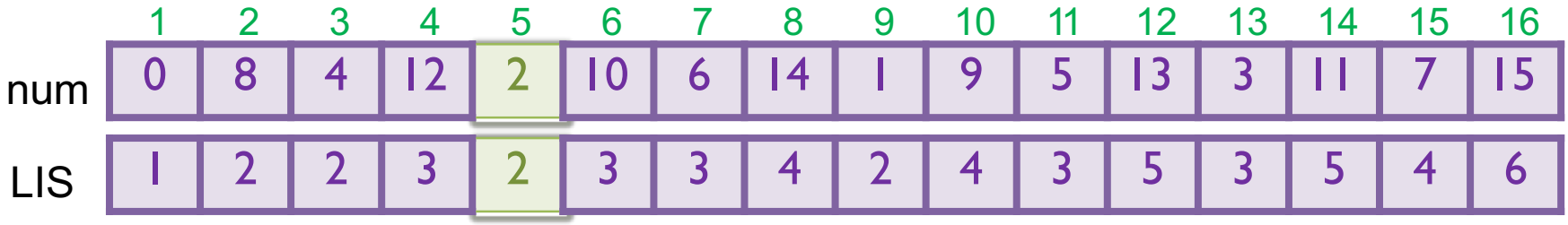
Made By mike199250

Learn more!



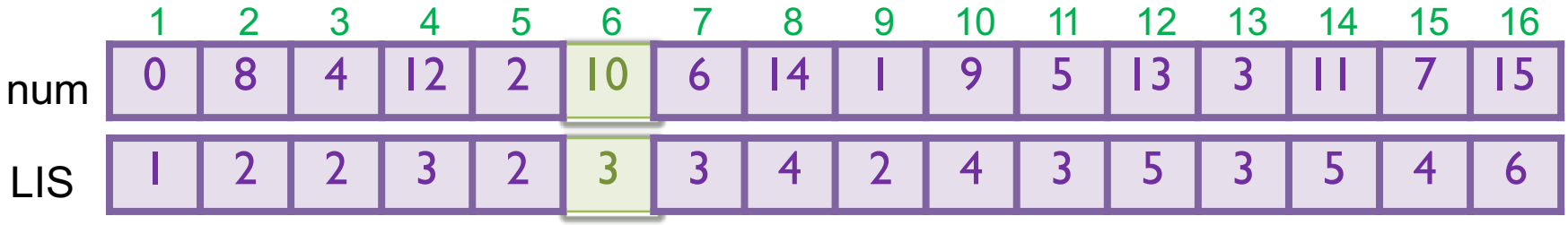
LIS =3	12
LIS =2	4
LIS =1	0

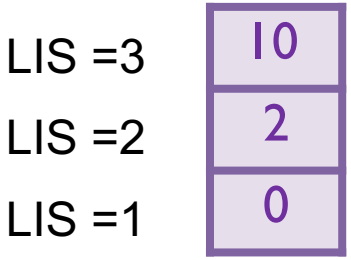
Learn more!



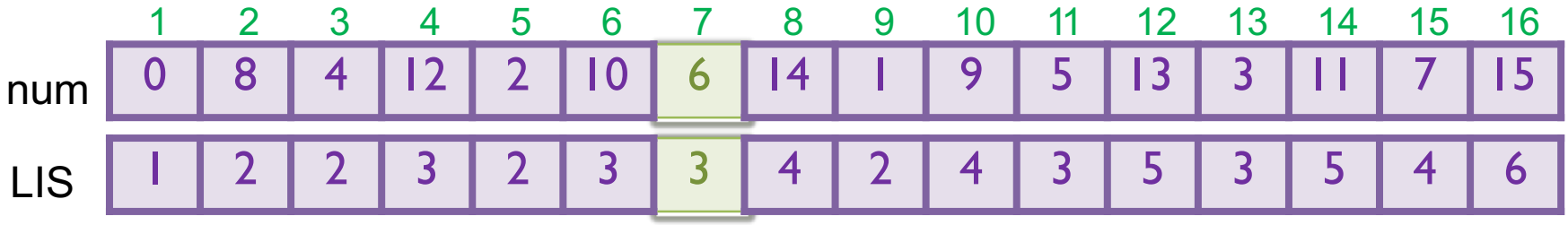
LIS =3	12
LIS =2	2
LIS =1	0

Learn more!


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


LIS =3
LIS =2
LIS =1
10
2
0

Learn more!



LIS =3	6
LIS =2	2
LIS =1	0

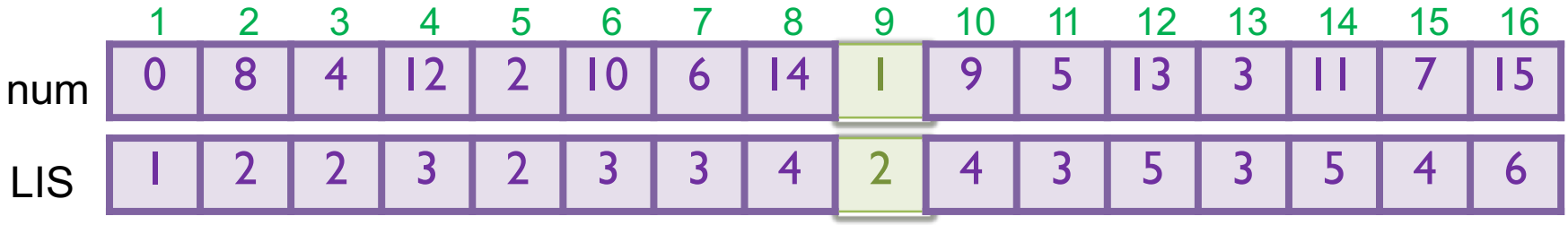
Learn more!

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

LIS =4	14
LIS =3	6
LIS =2	2
LIS =1	0

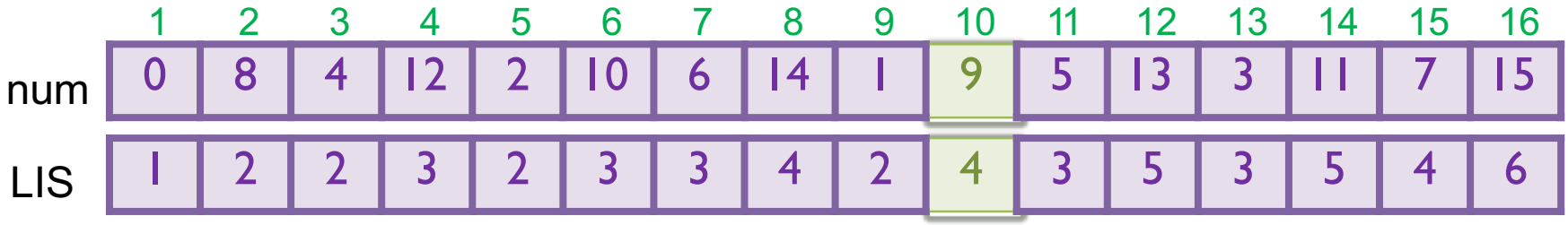
Made By mike199250

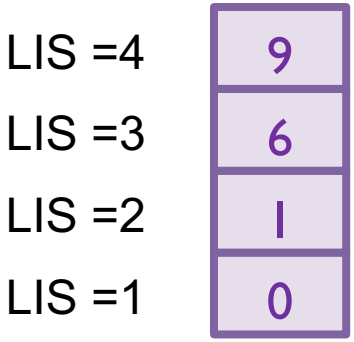
Learn more!



LIS =4	14
LIS =3	6
LIS =2	1
LIS =1	0

Learn more!





Made By mike199250

Learn more!

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

LIS =4	9
LIS =3	5
LIS =2	1
LIS =1	0

Made By mike199250

Learn more!

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

LIS =5	13
LIS =4	9
LIS =3	5
LIS =2	1
LIS =1	0

Learn more!

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

LIS =5	13
LIS =4	9
LIS =3	3
LIS =2	1
LIS =1	0

Learn more!

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

LIS =5	11
LIS =4	9
LIS =3	3
LIS =2	1
LIS =1	0

Made By mike199250

Learn more!

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

LIS =5	11
LIS =4	7
LIS =3	3
LIS =2	1
LIS =1	0

Made By mike199250

Learn more!

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

LIS =6	15
LIS =5	11
LIS =4	7
LIS =3	3
LIS =2	1
LIS =1	0

Learn more!

- There will be some ordering, so we can use binary-search to find LIS[k], then update the array.

- Find the best place it can be, and update
- It`s greedy!