Input: Give List = [50, 130, 110 40, 30, 70, 0, 90 ,60 ,120 ,100, 10, 80 ,20]

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

Expected Output: result = [3, 5, 7, 9]

1/ Map is used to track the value and the index inside the given list

2/ sort the value inside the map in increasing order

Map = [

6: 0,

11: 10,

13: 20,

4: 30,

3: 40,

0: 50,

8: 60,

5: 70,

12: 80,

7: 90,

10: 100,

2: 110,

9: 120,

1: 130]

3/ Convert the map into a new list

newList X = **[6, 11, 13, 4, 3, 0, 8, 5, 12, 7, 10, 2, 9, 1]**

4/ max = 13

Let result = [];

Begin: X[i] = 6

tempArr = [X[i]];

From 6 to 13: 13 - 6 = 7 => max_dif = 7

Check max_diff = 0. If yes, continue.

- Check diff = 1:

- Check longest: 6 + (0-1) * 1 < 13 -> Need to check existence.
- Check existence: 6 + 1 = 7 from i = 6 -> pass -> tempArr = [6,7] and update arr.length = 2, update the cheking position
- Check existence: 6 + 2 = 8 from i =7 -> not pass -> stop checking
- Check whether result.length < tempArr.length. If yes, you update the result.length and result = tempArr. If no, stop, move to new diff.

newList X = **[6, 11, 13, 4, 3, 0, 8, 5, 12, 7, 10, 2, 9, 1]**

- Check diff = 2:
  - Check longest: 6 + (2-1)*2 < 13 -> Need to check existence
  - Check existence: 6 + 1*2 = 8 from i = 6 -> pass -> tempArr = [6,8] and update arr.length = 2, update the cheking position
  - Check existence: 6 + 2*2 = 10 from i = 8 -> pass -> tempArr = [6,8,10] and update arr.length = 3, update the cheking position
  - Check existence: 6 + 2*3 = 12 from i = 10 -> not pass -> stop checking
  - Check whether result.length < tempArr.length. If yes, you update the result.length and result = tempArr. If no, stop, move to new diff.

- Check diff = 3:
  - Check longest: 6 + 2*3 = 12 < 13 -> Need to check existence
  - Check existence: 6 + 1*3 = 9 from i = 6 -> pass -> tempArr = [6,9] and update arr.length = 2, update the cheking position
  - Check existence: 6 + 2*3 = 11 from i = 9 -> not pass -> stop checking
  - Check whether result.length < tempArr.length. If yes, you update the result.length and result = tempArr. If no, stop, move to new diff.

- Check diff = 4:
  - Check longest: 6 + 2*4 = 14 > 13 -> Don't need to check existence
  - Break the loop,
  - Check existence: 6 + 1*4 = 10 from i = 6 -> pass -> tempArr = [6,10] and update arr.length = 2, update the cheking position
  - Check existence: 6 + 2*4 = 14 from i = 10 -> not pass -> stop checking
  - Check whether result.length < tempArr.length. If yes, you update the result.length and result = tempArr. If no, stop, move to new diff.

- Check diff = 5: the same
  - Check longest: 6 + 2 * 7 > 13 -> Don't need to check existence
- Check diff = 6: the same
- Check diff = 7: the same

Begin: X[i] = 11

tempArr = [11]

From 11 to 13: 13 - 11 = 2 => max_dif = 2

Check max_diff = 0. If yes, continue

newList X = **[6, 11, 13, 4, 3, 0, 8, 5, 12, 7, 10, 2, 9, 1]**

- Check diff = 1:
  - Check longest: 11 + (3-1) * 1 = 13 -> Need to check existence
  - Check existence: 11 + 1 = 12 from i = 11 -> pass -> tempArr = [11,12] and update arr.length = 2, update the cheking position
  - Check existence: 11 + 1*2 = 13 from i = 12 -> not pass -> stop checking
  - Check whether result.length < tempArr.length. If yes, you update the result.length and result = tempArr. If no, stop, move to new diff.
- Check diff = 2;
  - Check longest: 11 + (3-1) * 2 >13 -> Don't Need to check existence
  - Break

Begin: X[i] = 13

tempArr = [13]

Check max_diff = 0. If yes, continue

Begin: X[i] = 4

tempArr = [X[i]] =[4]

max_Stride = 9

Check max_diff = 0. If yes, continue

- Check diff = 1:
  - Check longest: 4 + (3-1)*1 < 13-> Need to check existence
  - ……
- Check diff = 3 -> tempArr = [4,7,10] = [6,8,10] Same length, same increasing order.
  - Check whether result.length < tempArr.length. If yes, you update the result.length and result = tempArr. If no, stop, move to new diff.
  - ⇨ Result won't updated from [6,8,10] to [4,7,10]
- Check diff = 5:
  - Check longest: 4 + 2*5 = 14 > 13
  - Break

Begin: X[i] = 3

tempArr = [X[i]] =[3]

max_Stride = 10

Check max_diff = 0. If yes, continue

newList X = **[6, 11, 13, 4, 3, 0, 8, 5, 12, 7, 10, 2, 9, 1]**

- Check diff = 1 -> Skip
- Check dif = 2 -> [3,5,7,9].
    - Check longest: 3 + (3-1) * 2 < 13 -> Need to check existence
    - Check existence: tempArr =[3,5,7,9]
    - Check whether result.length < tempArr.length. If yes, you update the result.length and result = tempArr. If no, stop, move to new diff.
        - Result will be updated from [6,8,10] to [3,5,7,9]
        - Result.length = 4
- Check dif = 3
    - Check longest: 3 + (4-1)*3 =12 < 13 -> Need to check existence
    - Skip
- Check dif = 4
    - Check longest: 3 + (4-1)*4 > 13 -> Don't Need to check existence
    - Break

Check longest: Beginning checking point + (result.length – 1)* diff < max