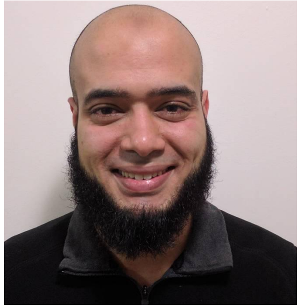
Python Programming

Practice

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

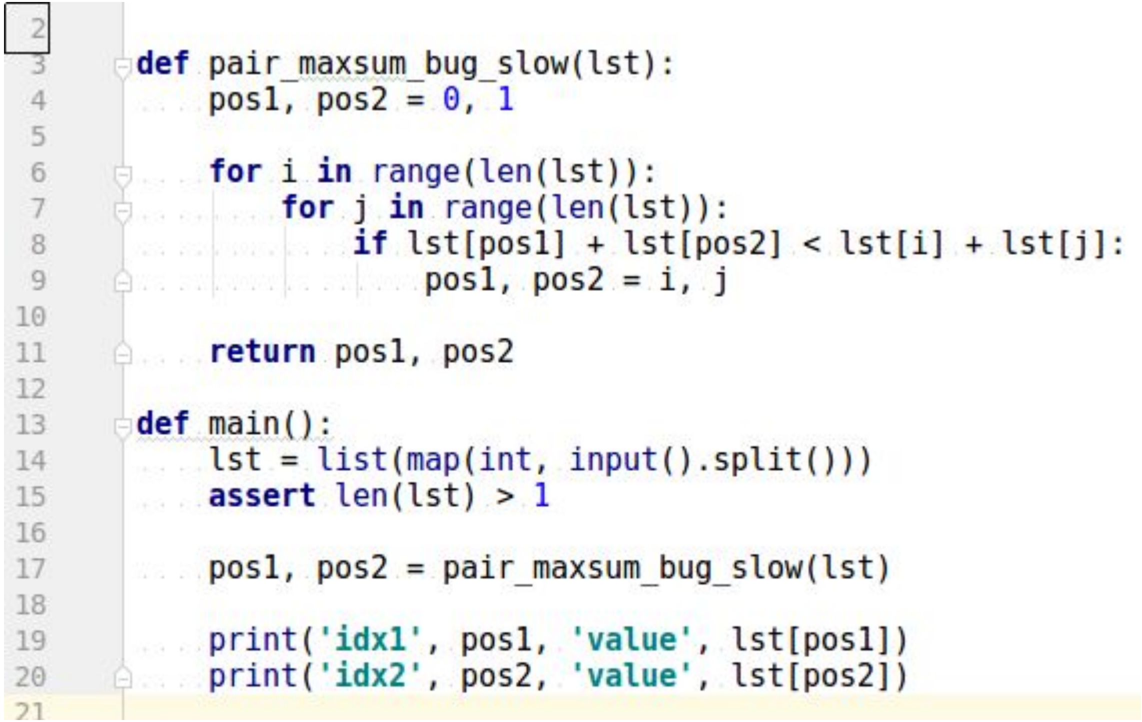


Practice: Find pair values of maximum sum

- Read a line of N integers (N > 1)
- Find a pair of indices whose **values’ sum is maximum**
- Input ⇒ output
  - 2 15 10 3 50 ⇒ 65 (from 50 + 15)
- Stop the video and code it
  - Do it with nested loops
  - Can you do with a linear loop?
    - e.g. not nested but can be several different 1 loop

Nested loop with bug

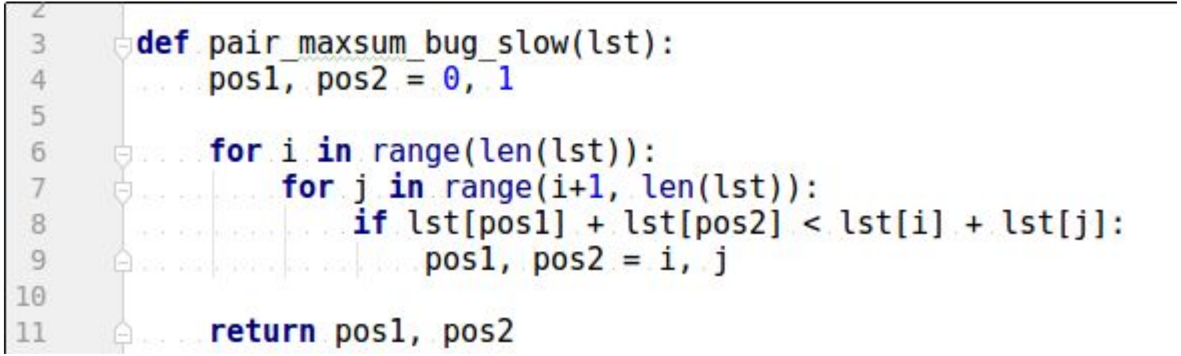
- I and J can be equal, this is buggy
- Solving the bug will also save us half the processing time!



```
def pair_maxsum_bug_slow(lst):
    pos1, pos2 = 0, 1

    for i in range(len(lst)):
        for j in range(len(lst)):
            if lst[pos1] + lst[pos2] < lst[i] + lst[j]:
                pos1, pos2 = i, j

    return pos1, pos2

def main():
    lst = list(map(int, input().split()))
    assert len(lst) > 1

    pos1, pos2 = pair_maxsum_bug_slow(lst)

    print('idx1', pos1, 'value', lst[pos1])
    print('idx2', pos2, 'value', lst[pos2])
```

Nested loop

- To fix, all what we need, the 2nd loop starts from i+1
    - Then we avoid duplicate bug
    - We also saves half of the processing
- Observe: for n = 10000, this codes perform around (10000^2)/2 operation
- Can we do it in a single loop style? Nothing nested
    - Even multiple separate 1 loop



```
def pair_maxsum_bug_slow(lst):
    pos1, pos2 = 0, 1

    for i in range(len(lst)):
        for j in range(i+1, len(lst)):
            if lst[pos1] + lst[pos2] < lst[i] + lst[j]:
                pos1, pos2 = i, j

    return pos1, pos2
```

Single loop

- The idea is based on simple observation!
- The maximum pair must come from the largest 2 values in the array
- So find the the first and 2nd maximum value
- Their sum is the answer
- This can be done trivially in a single loop style
- There is a seperate practice session for the code

Time Complexity

- You will study that in algorithms course
  - Just an informal note for now (not so accurate)
- When we have nested loop: we say time complexity is O(n^2)
  - It means for N=100, we need C * 100^2 operation, where C is some constant: e.g. 7
- When we have single loop: we say time complexity is O(n)
  - It means C * N operations. E.g. for N=10000, we need like 7 * 10000 operations
  - Imagine 3 seperate single loops, overall will be like 3-6 N operations
- Similarly: 3 nested loops have O(N^3)
  - This notation gives us a sense how fast is our algorithm
  - For N up to 100: O(N^3) is ok
  - For N up to 1000: O(N^2) is ok.
  - For larger N, we need O(N) solutions to really have our code run reasonably!

“Acquire knowledge and impart it to the people.”

“Seek knowledge from the Cradle to the Grave.”