DSA: Example Problem for the Practicum Exam

Katja Tuma, Fernanda Madeiral

September 13, 2024

Abstract

This document includes an example exercise for the examination of assignment 1 in TestVision. The practicum examination in part A (A1 + A2) consists of 10 such questions. All questions include several statements that you are asked to read and analyze. Then, each question presents 'multiple choice' where only one choice is the correct answer. Each question is worth 1 point.

1 Assignment 1

Problem 1.1 *(SHARED IN CLASS) Consider the Python implementation of the Selection sort algorithm below and answer the question.*

This algorithm sorts a list of orders by selection time (`t_selection`, finding the good in the warehouse and bringing it to the surface) plus shipping time (`t_shipping`, constant) in ascending order (e.g., [100,300,500]).

```
1  #!/usr/bin/env python3
2  from typing import List, Tuple
3
4  class Order:
5      def __init__(self, id, selection_time, shipping_time):
6          self.id: int = id
7          self.selection_time: int = selection_time
8          self.shipping_time: int = shipping_time
9
10     def __eq__(self, other):
11         '''
12         ==
13         '''
14         assert isinstance(other, Order)
15         print("__eq__")
16         return (self.selection_time + self.shipping_time) == (other.
               selection_time + other.shipping_time)
17
18     def __gt__(self, other):
19         '''
20         >
21         '''
22         assert isinstance(other, Order)
23         # print("__gt__")
24         return (self.selection_time + self.shipping_time) > (other.
               selection_time + other.shipping_time)
25
26 def sort(data: List[Tuple[int, int, int]]) -> List[int]:
```

```
27      array: List[Order] = []
28
29      for id, selection_t, shipping_t in data:
30          array.append(Order(id, selection_t, shipping_t))
31
32      n = len(array)
33
34      for ind in range(n):
35          min_index = ind
36
37          for j in range(ind + 1, n):
38              # select the minimum element in every iteration
39              if array[min_index] > array[j]:
40                  min_index = i
41          # swapping the elements to sort the array
42          (array[ind], array[min_index]) = (array[min_index], array[ind])
43
44      return [a.id for a in array]
```

Read the following statements and determine whether they are true or false:
I. The current minimum is not assigned a correct value. The ***min_index*** *should be assigned the value of* `j` *and not* `i`*. The mistake is in the statement* ***min_index = i in line 40****.*
II. The implementation of the algorithm is correct and will always sort the array in the specified order.
III. The implementation of this algorithm is incorrect because it should return ***TRUE*** *only when the* `selection_time` *is greater and not* `selection_time + shipping_time`*. The mistake is in the statement* ***return (self.selection_time + self.shipping_time) > (other.selection_time + other.shipping_time) in line 24****.*
IV. The implementation of this algorithm is incorrect because the condition in the if statement is not using the correct operator. It should compare whether ***array[min_index] < array[j]*** *instead of* ***array[min_index] > array[j]****. The mistake is* ***in line 39****.*

Which of the following is true (only one option is correct):

- ☐ *Only I.*
- ☐ *Only II.*
- ☐ *Only III.*
- ☐ *I and IV.*

Solution is on the next page.

Solution:

- [x] Only I.
- [ ] Only II.
- [ ] Only III.
- [ ] I and IV.