Facit och kommentarer till tenta 2024-04-11 i DA2004 och DA2005

Del A

1. E
2. B
3. D
4. A
5. C
6. D, E
7. E
8. B

Del B: kodfrågor

9. Suggested solution: First check if the list is empty and raise the exception if that is the case. Otherwise loop through the numbers, add them up and count them, and return the average.

```
def average(lst):
    if not lst:
        raise ValueError
    total = 0
    n = 0
    for val in lst:
        total += val
        n += 1
    return total / n
```

10. Use a while-loop and return immediately when an acceptable answer is given.

```
def get_user_int(a, b):
    while True:
        answer = input('Which integer? ')
        try:
            int_answer = int(answer)
        except:
            print('Please answer with an integer.')
            continue
        if int_answer >= a and int_answer <= b:
            return int_answer
        else:
            print(f'Integer must be between {a} and {b}.')
```

11. The `in` operator is useful here, but not necessary.

```
def count_ampersands(filename):
    count = 0
    with open(filename) as h:
        for line in h:
            if '&' in line:
                count += 1
    return count
```

12.
- Error 1: You cannot do `range(lst)`, it has to be `range(len(lst)- 1)`.
- Error 2: The True and False return values have been switched. The loop is designed to detect when the sequence of numbers is not monotonically increasing, and it should then return False.

  Note: if switching $>$ for $\leq$, then only the first pair of numbers is tested if they are increasing.

13. One solution is:

```
def monotonic(lst):
    if increasing(lst) or decreasing(lst):
        return True
    else:
        return False
```

This can also be simplified to:

```
def monotonic(lst):
    return increasing(lst) or decreasing(lst)
```

14.

```
class Exercise:
    def __init__(self):
        self.data = {}

    def register(self, activity, duration):
        if activity in self.data:
            self.data[activity] += duration
        else:
            self.data[activity] = duration

    def show(self):
        print('Activity  Duration (min)')
        for activity, duration in self.data.items():
            print(f'{activity:9} {duration:>8}')
```