Facit och kommentarer till prov 2021-01-11 i DA2004/DA2005

Del A: flervalsfrågor

1. *C*

2. *A,C,D*

3. *D*

4. *A,B,C,D*

5. *C*

6. *D*

7. *B,D*

8. *C*

Del B: kodfrågor

9. Möjlig lösning:

```
def flip2(s):
    s_rev = []
    for item in s[::-1]:
        s_rev.append(item)
    return s_rev
```

10. Möjlig lösning:

```
def my_div_sum(data, x):
    my_sum = 0
    for i in data:
        try:
            my_sum += i / x
        except:
            if type(i) == str:
                print("Fel i indata")
                return None
    return my_sum
```

11. Funktionen returnerar positionerna där elementen är lika. Endast den kortare listan itereras över. En möjlig lösning:

```
def f2(z1,z2):
    indices = []
    n = min(len(z1),len(z2))
    for i in range(n):
        if z1[i] == z2[i]:
            indices.append(i)
    return indices
```

12. Möjlig lösning:

```python
def rec_match(s,t):
    if min(len(s),len(t)) == 0:
        return 0
    if min(len(s),len(t)) <= 1:
        return 1 if s[0] == t[0] else 0
    else:
        m = 1 if s[0] == t[0] else 0
        return m + rec_match(s[1:],t[1:])
```

13. Möjlig lösning:

```python
def show_total(d, l):
    # Note: we could use zip instead of a list comprehension
        here
    intervals = [(l[i], l[i+1]) for i in range(len(l)-1)]
    sum_iv = {}
    for i, j in intervals:
        if i >= j:
            continue
        interval_sum = 0
        for k in d.keys():
            if i <= k < j:
                interval_sum += d[k]
        sum_iv[(i, j)] = interval_sum
    if not sum_iv:
        return
    return sum_iv
```

14. Möjlig lösning:

```python
# a
def h(n, f , k):
    return f(n) % k

# b
print(h(7, lambda x: x , 5))
print(h(312, lambda x: x , 256))
print(h(7, lambda x: x**2 + x , 5))
print(h(312, lambda x: x**2 + x , 256))
```

15. Möjlig lösning:

```python
class FilterSet:
    def __init__(self, k, f1, f2):
        self.s1 = set()
        self.s2 = set()
        self.k = k
        self.f1 = f1
        self.f2 = f2

    # from uppg 14
    def h(self, n, f):
        return f(n) % self.k

    def add(self,n):
        val1 = self.h(n,self.f1)
        val2 = self.h(n,self.f2)
        self.s1.add(val1)
        self.s2.add(val2)

    def is_present(self,n):
        val1 = self.h(n, self.f1)
        val2 = self.h(n, self.f2)
        return val1 in self.s1 and val2 in self.s2

B = FilterSet(256, lambda x: x, lambda x: x**2 + x//2)
B.add(0)
B.add(124)
B.add(1)
B.add(259)
print(B.s1, B.s2)
print(B.is_present(1), B.is_present(259))
print(B.is_present(-1))
print(B.is_present(512)) # incorrect result due to hash-
    conflict
```