day2

December 6, 2020

[1]:

```
import pytest
import ipytest
ipytest.autoconfig()
```

We start by parsing the string. Given it's `<n>-<n> <c>: <l>` we can split it by spaces, and then parse each subset

[2]:

```
def parse(s):
    (m,c,l) = s.split()
    mn,mx = [int(i) for i in m.split('-')]
    return (mn,mx,c[0],l)
```

[3]:

```
example1 = "1-3 a: abcde"
actual = parse(example1)
assert (1,3,"a","abcde") == actual
mn,mx,c,l = actual
```

Now we have to be able to test whether the password in l is actually valid It's only valid if it contains at least mn occurences of c, and no more than mx occurences.

[4]:

```
def is_valid(mn,mx,c,l):
    occurrences = l.count(c)
    return mx >= occurrences >= mn
```

[5]:

```
assert is_valid(mn,mx,c,l)
```

We need to count them, this should be simple, but lets make a function to do it anyway

[6]:

```
def count_passwords(lines, validator):
    total = 0
    for line in lines:
        mn,mx,c,l = parse(line)
        if validator(mn,mx,c,l):
            total += 1
    return total
```

So let's count the number of correct passwords in the sample

[7]:

```
sample = ("1-3 a: abcde",
"1-3 b: cdefg",
"2-9 c: ccccccccc")

assert 2 == count_passwords(sample, is_valid)
```

The actual data

[8]:

```
input = open('day2.txt').readlines()
```

[9]:

```
count_passwords(input, is_valid)
```

[9]:

```
586
```

Oh no, we are interpreting the password policy wrong, let's write a new function that confirms the real policy mn and mx are in fact p1 and p2, positions in the password l. the character c must appear at either of p1 and p2, but not at both and not at neither

[10]:

```
def is_valid_position(p1,p2,c,l):
    if l[p1-1] == c and l[p2-1] != c:
        return True
    if l[p1-1] != c and l[p2-1] == c:
        return True
    return False
```

[11]:

```
sample = ("1-3 a: abcde",
          "1-3 a: cbade",
"1-3 b: cdefg",
"2-9 c: ccccccccc")

assert 2 == count_passwords(sample, is_valid_position)
```

Now let's run that on the original password file

[12]:

```
count_passwords(input, is_valid_position)
```

[12]:

```
352
```