MSYS Midterm Review Solutions

October 17, 2019

[1]:
```python
import math
import os
import random
import re
import sys
```

1 Grading Students

[4]:
```python
def gradingStudents(grades):
    final = []
    for grade in grades:
        if grade < 38 and grade >= 0:
            final.append(grade)
        elif grade >= 38:
            if grade == 100:
                final.append(grade)
            elif (5 - grade % 5) < 3:
                final.append(grade + (5 - grade % 5))
            elif (5 - grade % 5) == 3 or (5 - grade % 5) > 3:
                final.append(grade)
    return final

grades_count = int(input().strip())
grades = []

for _ in range(grades_count):
    grades_item = int(input().strip())
    grades.append(grades_item)

result = gradingStudents(grades)
result
```

```
4
73
67
```

```
38
33
```

[4]:
```
[75, 67, 40, 33]
```

[7]:
```
grades_count = int(input().strip())
grades = []

for _ in range(grades_count):
    grades_item = int(input().strip())
    grades.append(grades_item)

result = gradingStudents(grades)
result
```

```
19
84
94
21
0
18
100
18
62
30
61
53
0
43
2
29
53
61
40
14
```

[7]:
```
[85, 95, 21, 0, 18, 100, 18, 62, 30, 61, 55, 0, 45, 2, 29, 55, 61, 40, 14]
```

2 Time in Words

- Ideally it shouldn't be this many with conditions HAHA
- I did this to check for edge cases as well as make sure any possible input would be considered

[11]:
```
def timeInWords(h,m):
    h_str = ''
    m_str = ''
    link = ''
    minute_list = ["o' clock", "past", "to", 'quarter','half','minute',␣
↪'minutes'] # 00, 1 <= m <= 30, quarter = 15, 45, half = 30
    nums =␣
↪['one','two','three','four','five','six','seven','eight','nine','ten','eleven','twelve','th
             'fifteen','sixteen','seventeen','eighteen','nineteen','twenty',␣
↪'twenty one', 'twenty two', 'twenty three', 'twenty four',
            'twenty five', 'twenty six','twenty seven', 'twenty eight', 'twenty␣
↪nine'] # 0 - 28

    if 1 <= h and 12 >= h and m == 0:
        h_str = nums[h-1]
        m_str = minute_list[0]
        return h_str + ' ' + m_str
    elif 1 <= h and 12 >= h and m >= 1 and m <= 30:
        if m < 2:
            m_str = nums[m-1] + ' ' + minute_list[5]
            link = minute_list[1]
            h_str = nums[h-1]
        else:
            if m == 15:
                m_str = minute_list[3]
                link = minute_list[1]
                h_str = nums[h-1]
            elif m == 30:
                m_str = minute_list[4]
                link = minute_list[1]
                h_str = nums[h-1]
            elif m <= 20:
                m_str = nums[m-1] + ' ' + minute_list[6]
                link = minute_list[1]
                h_str = nums[h-1]
            elif m > 20:
                m_str = nums[19] + ' ' + nums[int(str(m)[1]) - 1] + ' ' +␣
↪minute_list[6]
                link = minute_list[1]
                h_str = nums[h-1]

        return m_str + ' ' + link + ' ' + h_str

    elif 1 <= h and 12 >= h and m > 30 and m < 60:
        if m == 45:
            m_str = minute_list[3]
            link = minute_list[2]
            h_str = nums[h]
        else:
            mins_left = 60 - m
            if mins_left > 1:
                m_str = nums[mins_left - 1] + ' ' + minute_list[6]
                link = minute_list[2]
                h_str = nums[h]
            else:
                m_str = nums[mins_left - 1] + ' ' + minute_list[5]
                link = minute_list[2]
                h_str = nums[h]

        return m_str + ' ' + link + ' ' + h_str

h = int(input())
m = int(input())

timeInWords(h,m)
```

```
5
47
```

[11]: 'thirteen minutes to six'

[12]:
```
h = int(input())
m = int(input())

timeInWords(h,m)
```

```
3
0
```

[12]: "three o' clock"

[13]:
```
h = int(input())
m = int(input())

timeInWords(h,m)
```

```
7
15
```

[13]: 'quarter past seven'

[14]:
```
h = int(input())
m = int(input())

timeInWords(h,m)
```

```
6
35
```

[14]:
```
'twenty five minutes to seven'
```

3 ginorTS

- not the best solution but it works
- note that you won't be allowed to use python libraries so the solution for this would require using a list of uppercase alphabet letters and lowercase alphabet letters

[16]:
```
def ginorts(string):
    ss = []
    for w in string:
        ss.append(w)

    odd = []
    even = []
    lower = []
    upper = []
    sortedd = []
    for ww in ss:
        if re.search('[0-9]',ww):
            if int(ww) % 2 == 1:
                odd.append(ww)
            elif int(ww) % 2 == 0:
                even.append(ww)
        elif re.search('[A-Z]',ww):
            upper.append(ww)
        elif re.search('[a-z]',ww):
            lower.append(ww)

    lower.sort()
    upper.sort()
    odd.sort()
    even.sort()

    sortedd.append(''.join(lower))
    sortedd.append(''.join(upper))
    sortedd.append(''.join(odd))
    sortedd.append(''.join(even))

    ginort = ''.join(sortedd)
    return ginort

string = input()
ginorts(string)
```

```
Sorting1234
```

[16]:
```
'ginortS1324'
```

[17]:
```
string = input()
ginorts(string)
```

```
 1qaz2wsx3edc4rfv5tgb6yhn7ujm8ik9ol0pQWERTYUIOPASDFGHJKLZXCVBNM
```

[17]:
```
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1357902468'
```

4 Encryption

- bonus because math presets aren't allowed
- this is possible to do it by making your own floor division and ceiling division functions using round()

[19]:
```
def encryption(s):

    c = math.ceil(math.sqrt(len(s)))
    r = math.floor(math.sqrt(len(s)))
    encrypt = []
    x = 0
    y = c

    for _ in range(c):
        encrypt.append(s[x:y])
        y = y + c
        x = x + c

    secret = []
    for i in range(c):
        encrypt2 = []
        for word in encrypt:
            encrypt2.append(word[i:i+1])
        secret.append(''.join(encrypt2))

    w = ''
    for word in secret:
        w = w + ' ' + word

    return w[1:]


# if __name__ == '__main__':
#     fptr = open(os.environ['OUTPUT_PATH'], 'w')
#     s = input()
#     result = encryption(s)
#     fptr.write(result + '\n')
#     fptr.close()

s = input()
encryption(s)
```

```
haveaniceday
```

[19]:
```
'hae and via ecy'
```

[20]:
```
s = input()
encryption(s)
```

```
feedthedog
```

[20]:
```
'fto ehg ee dd'
```

[21]:
```
s = input()
encryption(s)
```

```
chillout
```

[21]:
```
'clu hlt io'
```

[22]:
```
s = input()
encryption(s)
```

```
wclwfoznbmyycxvaxagjhtexdkwjqhlojykopldsxesbbnezqmixfpujbssrbfhlgubvfhpfliimvmnn
y
```

[22]:
```
'wmgjpnull cyjqlejgi lyhhdzbui wctlsqsbm fxeoxmsvv ovxjeirfm zadysxbhn nxkkbffpn
bawobphfy'
```

[ ]:
```

```