Ministry of Education, Singapore
Computing Teachers' Content Upgrading Course 2020

Practical Assessment 1 (Trial B)

19 Feb 2020

Time allowed: 3 hours

Instructions to candidates:

1. This is an **open-book** exam.
2. Answer all **three** questions.
3. You may complete your solutions in any IDE first before copy them into this Jupyter Notebook for submission.
4. Input validation is not required
5. Submit this Jupyter Notebook online before test ends. You may submit multiple times, but only last submission before test end time will be accepted. https://driveuploader.com/upload/xTTmmKYr6C/ (https://driveuploader.com/upload/xTTmmKYr6C/)
6. Please note that the sample test program may not be enough to test your program. Your programs will be tested with other inputs and they should exhibit the required behaviours to get full credit.

Name & Email

Enter your name and your email address.

```
# YOUR NAME
# YOUR EMAIL
```

Question 1

Implement a function `char_frequency()` which takes in a string `s`, and returns character frequencies in a dictionary structure.

In [1]:
```
# WRITE YOUR CODE HERE

def char_frequency(s):
    c = list(s)
    result = {}
    for i in c:
        if not result.get(i):
            result[i] = 1
        else:
            result[i] = result[i] + 1
    return result
```

Test Case 1

Expected output: `{'h': 1, 'e': 1, 'l': 3, 'o': 2, ' ': 1, 'w': 1, 'r': 1, 'd': 1}`

In [2]:
```
char_frequency('hello world')
```

Out[2]:
```
{'h': 1, 'e': 1, 'l': 3, 'o': 2, ' ': 1, 'w': 1, 'r': 1, 'd': 1}
```

Question 2

Implement a function `consecutive_diff()` which takes in a list `s`. It returns a list of numbers, whihc are the absolute difference of 2 consecutive elements in the list.

Hint: Use `abs()` to get absolute value of a number.

In [9]:
```
# WRITE YOUR CODE HERE

def consecutive_diff(s):
    result = []
    for i in range(len(s)-1):
        result.append(abs(s[i] - s[i+1]))
    return result
```

Test Case 1

Expected output: `[2, 6, 1, 2]`

In [10]:
```
consecutive_diff([1,3,9,8,6])
```

Out[10]:
```
[2, 6, 1, 2]
```

Question 3

Implement a function `find_n_smallest()` which takes in an integer list `s` and an integer `n`, it returns `nth` smallest number in the list.

- If `n` is greater than length of list, return greatest value in `s`.

- If `n` is 0 or 1, return smallest value in `s`.

In [14]:
```
# WRITE YOUR CODE HERE

def find_n_smallest(s, n):
    s = s.copy()
    while n > 1:
        if len(s) == 1: return s[0]
        m = min(s)
        s.remove(m)
        n = n - 1
    return min(s)
```

Test Case 1

Expected output: `3`

In [23]:
```
s = [4,1,2,3,5,6]
find_n_smallest(s, 3)
```

Out[23]: 3

Test Case 2

Expected output: `6`

In [24]:
```
s = [4,1,2,3,5,6]
find_n_smallest(s, 7)
```

Out[24]: 6

Question 4

Assume a Singapore car registration number is in this format `AAAxxxxC`, where

- `AAA` is the 3 letters (in capital letter)
- `xxxx` could be 1 to 4 digits
- `C` is the checksum based on values of the checksum of `AAA` and `xxxx`.

If `xxxx` has less than 4 digits, it is prefixed with 0s. For example, `13` will become `0013`.

The checksum is computed using the following steps:

- Create an empty list `X`.
- Take the last two letters from `AAA`; append the numerical position of the two alphabets to `X`.
  - For example, if the alphabets are `'BD'`, then `X = [2,4]`.
  - Hint: to find numerical position of `M`, use formular `ord('M') - ord('A') + 1`.
- Append the digits from `xxxx`, to list X.
  - For example, if `xxxx` is `5678`, then `X = [2, 4, 5, 6, 7, 8]`
- Sum the multiplication of list X with `[9, 4, 5, 4, 3, 2]` item-wise.

- Compute the remainder of the sum divided by `19`.
- Look up the checksum for the remainder using the table below:
  - `0 -> A  5 -> T  10 -> L  15 -> E`
  - `1 -> Z  6 -> S  11 -> K  16 -> D`
  - `2 -> Y  7 -> R  12 -> J  17 -> C`
  - `3 -> X  8 -> P  13 -> H  18 -> B`
  - `4 -> U  9 -> M  14 -> G`

Implement a function `validate_car_ploate()` which takes in a car plate number `s`, validates it and return either `True` or `False`.

In [25]:
```
# WRITE YOUR CODE HERE

def validate_car_plate(s):
    X = []

    letters = s[:3]
    nums = s[3:-1]
    nums = '{:>04}'.format(nums)
    nums = [int(i) for i in nums]
    checksum = s[-1:]
#     print(letters, nums, checksum)

    X.append(ord(letters[1]) - ord('A') + 1)
    X.append(ord(letters[2]) - ord('A') + 1)
    X.extend(nums)
    print(X)

    weights = [9, 4, 5, 4, 3, 2]
    total = 0
    for i in range(len(weights)):
        total = total + weights[i] * X[i]
    r= total % 19
    table = {0:'A', 5:'T', 10:'L', 15:'E',
             1:'Z', 6:'S', 11:'K', 16:'D',
             2:'Y', 7:'R', 12:'J', 17:'C',
             3:'X', 8:'P', 13:'H', 18:'B',
             4:'U', 9:'M', 14:'G'}

    return table[r] == checksum

validate_car_plate('SLF4178X')
```

```
[12, 6, 4, 1, 7, 8]
```

Out[25]:
```
True
```

Test Case 1

Expected output: `True`

In [26]:
```
validate_car_plate('SMG9411Y')
```

```
[13, 7, 9, 4, 1, 1]
```

Out[26]: True

Test Case 2

Expected output: `True`

In [27]:
```
validate_car_plate('SLA123G')
```

```
[12, 1, 0, 1, 2, 3]
```

Out[27]: True

Test Case 3

Expected output: `False`

In [29]:
```
validate_car_plate('SMG9411Z')
```

```
[13, 7, 9, 4, 1, 1]
```

Out[29]: False

Question 5

The harmonic sum is the sum of reciprocals of the positive integers.

$$\sum_{n=1}^{\infty} \frac{1}{n} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \cdots.$$

Implement a recursive function `harmonic_sum(n)` to compute the summation of first `n` items. The function returns final summation of the series.

In [30]:
```
# WRITE YOUR CODE HERE

def harmonic_sum(n):
    if n==1:
        return 1
    return 1/n + harmonic_sum(n-1)
```

Test Case 1

Expected output: `2.083333333333333`

In [31]:
```
harmonic_sum(4)
```

Out[31]:
```
2.083333333333333
```