Date: 22 June 2019

Day Objectives

- File Handling
  - Basic File Data Processing
    - Accessing and Modifying File Data
  - Character Count
  - Line Count
  - File Size
  - Word Count
  - Unique Word Count

In [1]:
```
# Read a File - File should exist(Read Mode)
# Write to a File - Existing(append) or New File(Write Mode)

def readFile(filePath):
    with open(filePath,'r') as f:   # with is key word open defines open tha
        print(type(f))
    return
filePath = 'DataFiles/data.txt'
readFile(filePath)
```

```
<class '_io.TextIOWrapper'>
```

In [4]:
```
# Read a File - File should exist(Read Mode)
# Write to a File - Existing(append) or New File(Write Mode)

# Function to read entire file data into a single string
def readFile(filePath):
    with open(filePath,'r') as f:   # with is key word open defines open tha
        filedata = f.read()        #Reads the entire file data into a string
    return filedata

filePath = 'DataFiles/data.txt'
#readFile(filePath)    # --->ouput is 'new data\nLine1\nLine2\nLine3\nLine4'
print(readFile(filePath))
```

```
new data
Line1
Line2
Line3
Line4
```

In [2]:
```
# Function to read a file into a list of lines
# Each element in the list is one line in the file
def readFileIntoList(filepath):
    with open(filepath,'r') as f:
        filedata = f.read()
        lines= filedata.split('\n')
        # lines = []
        #for line in f:
        #lines.append(line)
    return lines

filepath='DataFiles/data.txt'
readFileIntoList(filepath)
```

Out[2]:
```
['new data', 'Line1', 'Line2', 'Line3', 'Line4']
```

In [8]:
```
# Fuction to count number of lines in a file

def countLinesFile(filepath):
    count = len(readFileIntoList(filepath))
    return count
countLinesFile(filepath)
```

Out[8]:
```
5
```

In [5]:
```
# Function to count the number of characters in a file

def charCountFile(filepath):
    count= len(readFile(filepath))
    return count
charCountFile(filepath)
```

Out[5]:
```
32
```

In [7]:
```
# Function to count the number of words in a file
import re

def wordCountFile(filepath):
    pattern = '[ \n]'
    filedata = readFile(filepath)
    count = len(re.split(pattern,filedata))
    return count
wordCountFile(filepath)
```

Out[7]:
```
6
```

In [ ]:
```
# Function to count the number of words in a file using loop
```

In [ ]:
```
# Function to count no of unique word in a file
```

In [12]:
```
# Function to get unique elements in a list

# [1,2,3,3,2,1] --> [1,2,3]
# Create a empty unique list [] and then add unique data into it [1,2,3]

def uniqueData(li):   # li has repeat elements
    # Create an empty unique list
    unique = []

    # For every element in the main list,
        # Check if it exists in the unique list
        # If it does not exist,add it to unique list
        # If it already exists, move on to the next element in the main list
    for element in li:
        if element not in unique:
            unique.append(element)
    return unique   # --> [1,2,3]
    # return len(unique)   #--> 3
li = [1,2,3,3,2,1]

uniqueData(li)

```

Out[12]: [1, 2, 3]

In [17]:
```
# Function to get unique words count in a file
def countUniqueWord(li):
    unique = []
    for element in li:
        if element not in unique:
            unique.append(element)
    return unique
li=readFileIntoList(filepath)
countUniqueWord(li)
```

Out[17]: ['new data jumed', 'Line1', 'Line2', 'Line3', 'Line4', 'new data', 'line 1']

In [ ]:
```

```

In [ ]:
```
# Function to get unique words
```

In [ ]:
```
# Function to count the number of unique words in a text file
```

In [21]:
```
# Function to print the frequency count of all words in file
# Frequency Distribution
# Data in Line 1
# Data in Line 2
# Data in Line 3
# O/P
# Data : 3
# in : 3
# Line : 3
# 1 : 1
# 2 : 1
# 3 : 1

def frequencyCount(filepath):
    filedata = readFile(filePath).split()
    print('main file data',':',filedata)
    x=[]
    y=[]
    for i in filedata:
        if i not in x:
            x.append(i)
    print('unique file data',':',x)
    for i in x:
        c=y.append(filedata.count(i))
        print(i,':',filedata.count(i))
    return
frequencyCount(filepath)
```

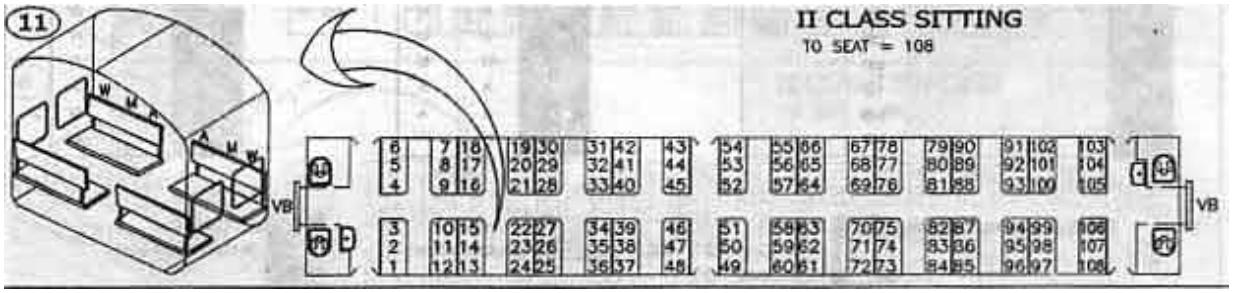
```
main file data : ['new', 'data', 'jumed', 'Line1', 'Line2', 'Line3', 'Line4',
'Line4', 'new', 'data', 'line', '1', 'Line1', 'new', 'data', 'Line2']
unique file data : ['new', 'data', 'jumed', 'Line1', 'Line2', 'Line3', 'Line4',
'line', '1']
new : 3
data : 3
jumed : 1
Line1 : 2
Line2 : 2
Line3 : 1
Line4 : 2
line : 1
1 : 1
```

In [18]:
```
# Contacts Application

# Find and Replace Application?
```

Seating Arrangement

- Akash and Vishal are quite fond of travelling. They mostly travel by railways. They were travelling in a train one day and they got interested in the seating arrangement of their compartment. The compartment looked something like

TrainCompartment (TrainCompartment.jpg)



- So they got interested to know the seat number facing them and the seat type facing them. The seats are denoted as follows :
- Window Seat : WS
- Middle Seat : MS
- Aisle Seat : AS
- You will be given a seat number, find out the seat number facing you and the seat type, i.e. WS, MS or AS.
- INPUT
  - First line of input will consist of a single integer T denoting number of test-cases. Each test-case consists of a single integer N denoting the seat-number.
- OUTPUT
  - For each test case, print the facing seat-number and the seat-type, separated by a single space in a new line.
- CONSTRAINTS
  - 1<=T<=105
  - =N<=108
- SAMPLE INPUT
  - 2
  - 18
  - 40
- SAMPLE OUTPUT
  - 19 WS
    - 45 AS

In [11]:
```
t=int(input())
for i in range(1,t+1):
    sn=int(input())

    if(sn%12==0):
        print(sn-11,'WS')
    elif(sn%12==11):
        print(sn-9,'MS')
    elif(sn%12==10):
        print(sn-7,'AS')
    elif(sn%12==9):
        print(sn-5,'AS')
    elif(sn%12==8):
        print(sn-3,'MS')
    elif(sn%12==7):
        print(sn-1,'WS')
    elif(sn%12==6):
        print(sn+1,'WS')
    elif(sn%12==5):
        print(sn+3,'MS')
    elif(sn%12==4):
        print(sn+5,'AS')
    elif(sn%12==3):
        print(sn+7,'AS')
    elif(sn%12==2):
        print(sn+9,'MS')
    elif(sn%12==1):
        print(sn+11,'WS')
```

```
1
45
40 AS
```

In [10]:
```
n=4
print(n,"WS")
```

```
4 WS
```

In [15]:
```
N = int(input())
A = input().split()
#print(A)
```

```
2
2 3 5 6 7
```

In [17]:
```
N = int(input())
A = input().split()
li = []
for i in range(n+1):
    li.append(int(A[i]))
    print(li[i],end=' ')


```

```
5
1 2 3 4 5 6 7
1 2 3 4 5
```

Find Product

- You have been given an array A of size N consisting of positive integers. You need to find and print the product of all the number in this array Modulo 10^9+7 .
- Input Format:
  - The first line contains a single integer N denoting the size of the array. The next line contains N space separated integers denoting the elements of the array
- Output Format:
  - Print a single integer denoting the product of all the elements of the array Modulo 10^9+7 .
- Constraints:
  - 1<=N<=10^3
  - 1<=A[i]<10^3
- SAMPLE INPUT
  - 5
  - 1 2 3 4 5
- SAMPLE OUTPUT
  - 120
- Explanation
  - There are 5 integers to multiply. Let's store the final answer in 'answer' variable. Since 1 is identity value for multiplication, initialize 'answer' as 1.
  - answer = 1
  - answer = (answer*1)%(10^9+7)
  - answer = (answer*2)%(10^9+7)
  - answer = (answer*3)%(10^9+7)
  - answer = (answer*4)%(10^9+7)
  - answer = (answer*5)%(10^9+7)
  - The above process will yield answer as 120

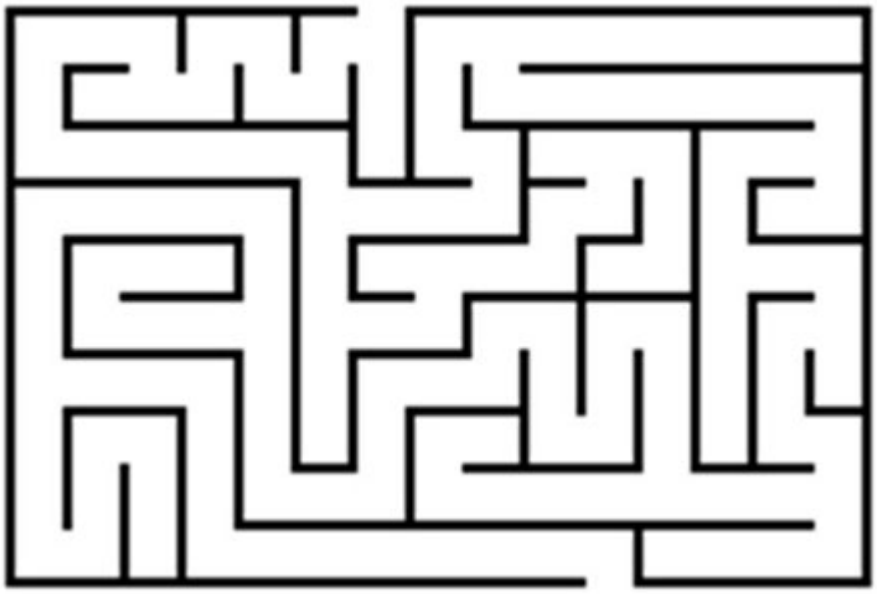
In [26]:
```
N = int(input())
A = input().split()
#li = []
answer=1
for i in range(N):
    #li.append(int(A[i]))
    # print(li[i],end=' ')
    answer=(answer*int(A[i]))%(1000000007)
print(answer)

```

```
5
1 2 3 4 5
120
```

e-maze-in

- Ankit is in maze. The command center sent him a string which decodes to come out from the maze. He is initially at (0, 0). String contains L, R, U, D denoting left, right, up and down. In each command he will traverse 1 unit distance in the respective direction.

e-maze-in (e-maze-in.jpg)



- For example if he is at (2, 0) and the command is L he will go to (1, 0).
- Input:
- Input contains a single string.
- Output:
- Print the final point where he came out.
- Constraints:
  - 1 ≤ |S| ≤ 200
- SAMPLE INPUT
  - LLRDDR
- SAMPLE OUTPUT
  - 0 -2

In [25]:

```
# Puzzle game ---> maze
s=input()
x=0
y=0
for i in s:
    if(i=='L'):
        x = x-1
    elif(i=='R'):
        x = x+1
    elif(i=='D'):
        y = y-1
    elif(i=='U'):
        y = y+1
print(x,y)
```

```
LLRDDR
0 -2
```

In [ ]:

```

```