EAST TechSprint Log:

Use this document to log your work for the TechSprint period. Remember, your log should reflect adequate effort to achieve growth in the skills you are learning. This work is to be done **in addition** to any work that you are doing or skills you are developing for your EAST Project (although the work can be complementary). Each log entry should follow the following format:

- Date
- Description of learning goal or task
- Link to tutorial resource(s)
- Example of **your work** based on the tutorial and goals (screenshot preferred), embedded in this document
- Reflection on what you learned, challenges or other important elements

This log will be due by 11:59 PM on the Sunday following the TechSprint period.

Date: 4/10

Task: Learning how to find mode(s) using a list in python

Link to Resource:

- https://edu.goorm.io/learn/lecture/554/%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98-%EB%AC%B8%EC%A0%9C%ED%95%B4%EA%B2%B0%EA%B8%B0%EB%B2%95-%EC%9E%85%EB%AC%B8/lesson/25592/%EB%AC%B8%EC%A0%9C3a-%EC%A0%84%ED%99%94%EB%B2%88%ED%98%B8

```python
n = int(input()) #전화번호의 수
data = [] # 각 전화번호의 뒷자리
for i in range(n):
  num = int(input())
  data.append(num)

cnt = [0] * 10000 #cnt[XXXX] := 뒷 자리가 XXXX인 전화번호의 수
for j in range(n):
  this_num = data[j]
  cnt[this_num] += 1

max_index = 0 #가장 많이 등장한 뒷자리 후보
for num in range(10000): #모든 뒷자리에 대하여
  if cnt[num] > cnt[max_index]:
    max_index = num

print("%04d" % max_index )
```

Reflection:

When using two-dimensional arrays, detecting a mode among a set of numbers is possible.

Variable = [a] * b makes the variable to be a list while putting a as its value for b times.

Date: 4/11 – 4/12

Task: Learning how to find mode(s) using a list and loop in python

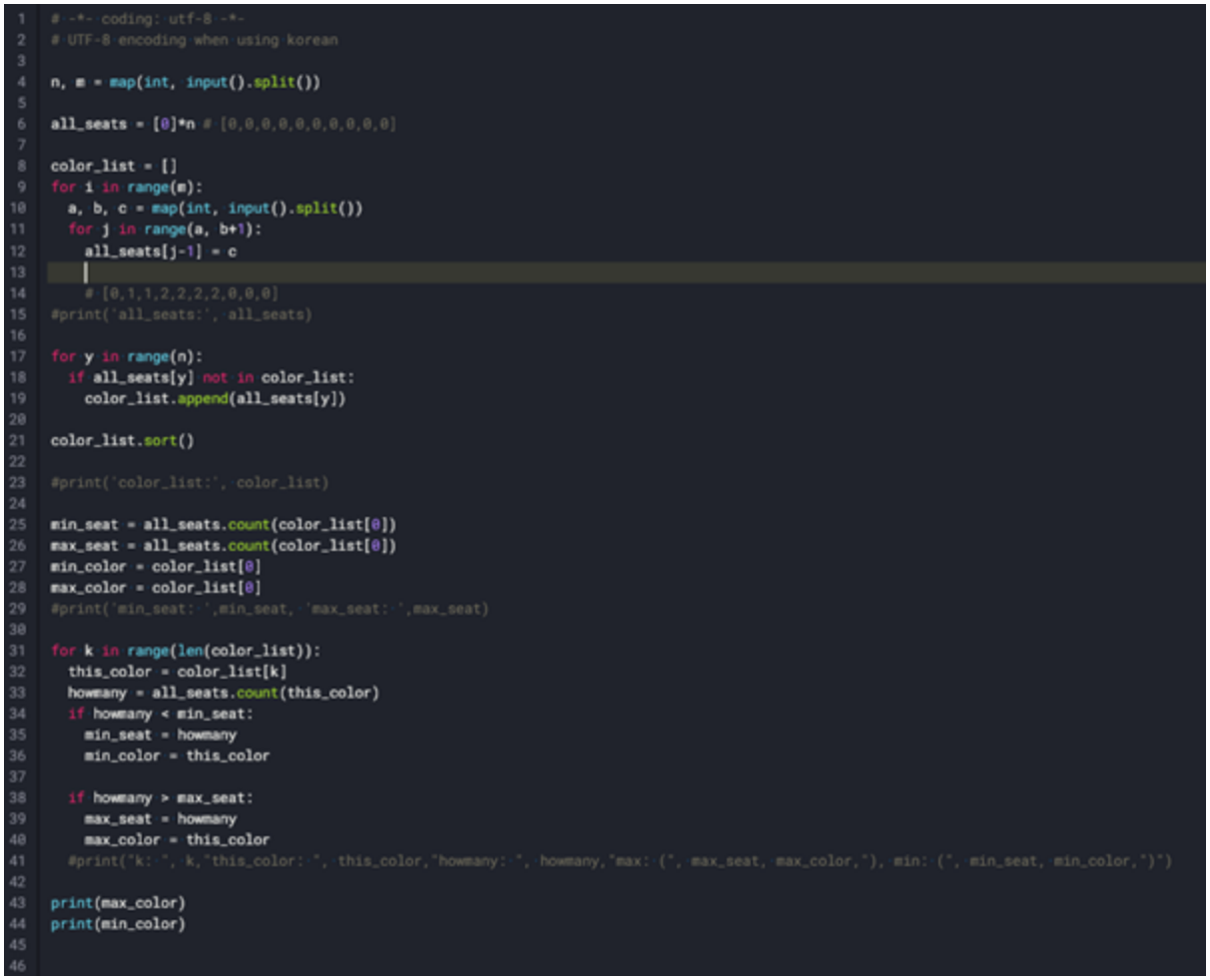
Link to resource:

- https://edu.goorm.io/learn/lecture/554/%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98-%EB%AC%B8%EC%A0%9C%ED%95%B4%EA%B2%B0%EA%B8%B0%EB%B2%95-%EC%9E%85%EB%AC%B8/lesson/25593/%EB%AC%B8%EC%A0%9C3b-%ED%8E%98%EC%9D%B8%ED%8A%B8
- https://github.com/dongyi-kim/IntroductionToAlgorithmProblemSolvingTechniques/blob/master/Chapter03/Problem03B/solution.cpp

1st trial: syntax error

```python
n, m = map(int, input().split())  # n and m are natural numbers no more than 1000.

data = []
for i in range(m):
  three_nums = list(map(int, input().split()))
  data.append(three_nums)

seats = [0] * n  # Seat numbers are natural numbers between 1 and n.

for j in range(m):
  jj = data[j]
  start = jj[0]
  end = jj[1]
  color = jj[2]  # Color number is an integer, no less than 0 no more than 99.
  for k in range(start, end+1)
    seats[k-1] = color

max_color = 0

for l in range(m):
  color = data[l][2]
  if seats(color) > seats(max_color):
    max_color = color
  elif seats(color) == seats(max_color):
    if color < max_color:
      max_color = color

min_color = max_color
for z in range(m):
  color = data[z][2]
  if seats(color) < seats(min_color):
    min_color = color
  elif seats(color) == seats(min_color):
    if color < min_color:
      min_color = color

print(max_color, min_color, end="")
```

2nd trial: answer



```python
# -*- coding: utf-8 -*-
# UTF-8 encoding when using korean

n, m = map(int, input().split())

all_seats = [0]*n # [0,0,0,0,0,0,0,0,0,0]

color_list = []
for i in range(m):
  a, b, c = map(int, input().split())
  for j in range(a, b+1):
    all_seats[j-1] = c

    # [0,1,1,2,2,2,2,0,0,0]
#print('all_seats:', all_seats)

for y in range(n):
  if all_seats[y] not in color_list:
    color_list.append(all_seats[y])

color_list.sort()

#print('color_list:', color_list)

min_seat = all_seats.count(color_list[0])
max_seat = all_seats.count(color_list[0])
min_color = color_list[0]
max_color = color_list[0]
#print('min_seat: ',min_seat, 'max_seat: ',max_seat)

for k in range(len(color_list)):
  this_color = color_list[k]
  howmany = all_seats.count(this_color)
  if howmany < min_seat:
    min_seat = howmany
    min_color = this_color

  if howmany > max_seat:
    max_seat = howmany
    max_color = this_color
  #print("k: ", k,"this_color: ", this_color,"howmany: ", howmany,"max: (", max_seat, max_color,"), min: (", min_seat, min_color,")")

print(max_color)
print(min_color)
```

Reflection:

list.sort() function can sort the elements in a list from lower number to higher number or alphabetical order.

When using a loop to find the mode, either counting numbers from a list in which all the numbers are aligned or creating a table to pair each number with its frequency of appearing can be a method.

Date: 4/14 – 4/15

Task: Learning the advantages of an array in Python

Link to resource:

- https://edu.goorm.io/learn/lecture/554/%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98-%EB%AC%B8%EC%A0%9C%ED%95%B4%EA%B2%B0%EA%B8%B0%EB%B2%95-%EC%9E%85%EB%AC%B8/lesson/25594/%EB%AC%B8%EC%A0%9C3c-%EC%9D%91%EB%AA%A8

- https://jupiny.tistory.com/entry/list%EC%97%90%EC%84%9C-%EC%9B%90%EC%86%8C%EB%A5%BC-%EB%B3%80%EA%B2%BD%ED%95%98%EB%8A%94-%EB%B0%A9%EB%B2%95

```
n = int(input())

data = list(map(int, input().split()))

for i in range(n):
  if data.count(data[i]) >= 2:
    a = data[i]
    if a != 0:
      for j in range(n):
        if data[j] == a:
          data[j] = 0
      data[i] = 0

zero_num = data.count(0)
for k in range(zero_num):
  data.remove(0)

data.sort()

for l in range(len(data)):
  print(data[l], end=" ")
```

Reflection:

list.remove(a) function only removes one value of a that shows up the first.

When comparing numbers in a for-loop, the number used for comparison should be set as a variable to remain constant.

When removing numbers using a for-loop, it is effective to replace the numbers with other unnecessary values and then remove them at the end.

Date: 4/17-4/18

Task: Learning how to create and find the value of a certain number in the Fibonacci sequence

Link to resource:

- https://edu.goorm.io/learn/lecture/554/%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98-%EB%AC%B8%EC%A0%9C%ED%95%B4%EA%B2%B0%EA%B8%B0%EB%B2%95-%EC%9E%85%EB%AC%B8/lesson/25595/%EB%A

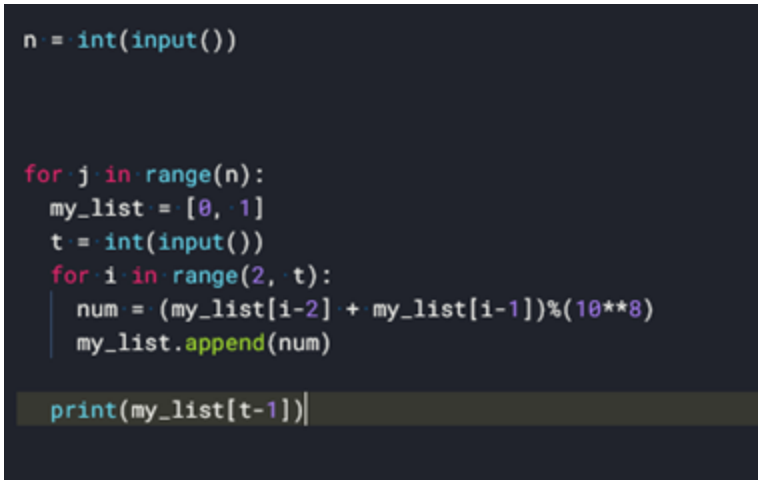
C%B8%EC%A0%9C3d-%ED%94%BC%EB%B3%B4%EB%82%98%EC%B9%98-%EB%82%98%EB%A8%B8%EC%A7%80

- https://webisfree.com/2018-12-30/python-%EC%88%AB%EC%9E%90%EA%B0%92%EC%9D%98-%EC%98%AC%EB%A6%BC-%EB%B2%84%EB%A6%BC-%EB%B0%8F-%EB%B0%98%EC%98%AC%EB%A6%BC-%EB%B0%A9%EB%B2%95
- https://www.codingfactory.net/10034

1st trial: error

```
n = int(input())

my_list = [0, 1]
t = int(input())


def eight_function(num):
  a = num / 10**8
  b = int(a)
  c = a - b
  out_num = c * 10**8
  return(out_num)

for i in range(2, n):
  i_num = my_list[i-2] + my_list[i-1]
  this_num = eight_function(i_num)
  my_list.append(this_num)

result = my_list[t-1]


print(result)
```

2nd trial:



```
n = int(input())



for j in range(n):
  my_list = [0, 1]
  t = int(input())
  for i in range(2, t):
    num = (my_list[i-2] + my_list[i-1])%(10**8)
    my_list.append(num)

  print(my_list[t-1])
```

Reflection:

When creating a Fibonacci sequence, it is effective to set the first and second values as 0 and 1, respectively.

When getting only the last eight numbers from each number, it is effective to first get the remainder of dividing it by 8 and then multiply them by 10**8.