과목 명: 시스템프로그래밍

담당 교수 명: 김 지 환

<<Assignment 4>>

**서강대학교 컴퓨터학과**

**[20151616]**

**[최승환]**

목 차

1. 프로그램 개요 3

2. 프로그램 설명 3

2.1 프로그램 흐름도 3

3. 모듈 정의 3

3.1 모듈 이름 : Max Counters 3

3.1.1 기능 3

3.1.2 사용 변수 3

3.2 모듈 이름: Common Prime Divisors 3

3.2.1 기능 3

3.2.2 사용 변수 3

3.3 모듈이름: Min abs sum of two 4

3.3.1 기능 4

3.3.2 사용변수 4

3.4 모듈이름: Max pooling 4

3.4.1 기능 4

3.4.2 사용변수 4

4. 전역 변수 정의 (생략) 4

5. 코드

# 프로그램 개요

Python 을 이용해 작성한 코드로 기본적인 문제 해결능력을 Python 을 이용해, 언어의 특징을 잘 살려 짜는 것에 초점을 맞춰 작성했습니다. 작업환경은 Google Colaboratory 에서 jupyter notebook 을 사용했습니다. 입출력은 기본 Python 함수들을 사용했고, 유저에 대한 입력에 따라 스크린에 출력되도록 작성되어 있습니다. 문제는 총 4개로 3장에서 각 문제에 대한 알고리즘 설명을 참고할 수 있습니다.

# 프로그램 설명

## 프로그램 흐름도(생략)

# 모듈 정의

## Max Counters

### 기능

숫자 N 를 입력 받은 후, 리스트 lista 를 입력받습니다. N 개의 인덱스를 가지는 리스트 listb를 하나 선언하고, lista 의 값에 따라, listb 의 값을 적절히 바꿔줍니다.

추가적으로, 빈 입력이 들어오는 경우에 대하여, 정상적인 입력이 들어올 때까지 입력을 받도록 되어있습니다.

바꾸는 내용은 다음과 같습니다.

lista의 첫 인덱스부터 마지막 인덱스까지 참조합니다. 이 때 인덱스를 index. lista 에 저장되어 있는 해당 인덱스의 값을 lista[index] 라고 하겠습니다.

1. IncreaseX  
   lista의 첫 인덱스부터 마지막 인덱스까지 참조해 lista[index] 가 1보다 크고, N 보다 작으면,  
   listb[lista[index]] 의 값을 1 늘려줍니다.
2. MaxCounter  
   lista[index] 가 N+1 과 같은 값을 가지면, listb의 모든 인덱스의 값을 listb 에 저장된 값들 중 가장 큰 숫자로 바꿔줍니다.

사용 변수

* + - 1. lista , lnum , listb

lista 는 사용자가 입력하는 리스트의 숫자들을 저장하는 리스트입니다. 입력 받을 때에는 string 형의 데이터들을 저장하고 있지만, 실제 사용할 때는 그 string 들을 int 형으로 변환해 사용합니다.

lnum 는 사용자가 처음 입력하는 N 을 저장하는 변수입니다. N에 따라서, listb의 크기가 결정되고, IncreaseX 의 호출여부가 결정됩니다.

listb 는 lnum 에 의해 생성된 리스트입니다. lista 에 저장된 값들에 따라 listb의 값이 달라집니다. lista에 저장된 값을 listb의 인덱스 로 사용해 IncreaseX를 호출합니다.

코드부

def IncreaseX(aList, index):

aList[index] = aList[index]+1

def MaxCounter(aList):

maxint = aList[0]

for i in range(lnum):

if(aList[i] > maxint):

maxint = aList[i]

for i in range(lnum):

aList[i] = maxint;

lnum = input().split("\n")[0]

while(len(lnum) == 0):

lnum = input().split("\n")[0]

lnum = int(lnum)

lista = input().split("\n")[0]

while(len(lista) == 0):

lista = input().split("\n")[0]

lista = lista.split(" ")

listb = [0]\*lnum

for i in range(len(lista)):

lista[i] = int(lista[i])

#list variables type changed to int

for i in range(len(lista)):

if(1<= lista[i] <= lnum):

IncreaseX(listb, lista[i]-1)

#lista[i] 는 몇번째인지를 가리키기 때문에 인덱싱에는 1 작은 값을 사용한다.

elif(lista[i] == (lnum+1)):

MaxCounter(listb)

for i in listb:

print(i,end=" ")

## Common Prime Divisors

### 기능

숫자 D 에 대한 소수인 인수들을 찾아 Set 으로 저장합니다. 두 숫자 D1 D2 의 소인수 Set이 같은 경우 조건을 만족하는 것으로 판별합니다. n개의 숫자 쌍을 입력받아 몇 쌍이 조건을 만족하는지 출력합니다.

### 사용 변수

* + - 1. s1, s2 count, index, buf1, buf2

s1 은 D1의 소인수 집합입니다. s2는 D2의 소인수 집합입니다. s1과 s2 는 IsDiv 함수에서 소수들을 입력해 나뉘어지는 것이 있는지 확인한 뒤, 나눠지면 그 소수를 저장하는 방식으로 두 번의 함수 호출로 s1 s2가 각각 저장됩니다.

count 는 조건을 만족하는 쌍이 몇 개인지를 저장하는 변수입니다. 마지막에 출력되는 값입니다. index 는 입력으로 생성된 두 개의 list의 인덱스를 저장하고 있습니다. s1과 s2 에 저장된 값들을 참조할 때 사용되는 인덱스입니. 이를 IsDiv 에 넘겨 set을 만듭니다.buf1과 buf2 는 입력받은 숫자 쌍의 D1 과 D2를 저장하는 리스트들입니다. buf1의 숫자 하나와 buf2의 숫자 하나를 D1 D2로 사용합니다.

def IsDiv(n,mySet):

primeInt = [2,3,5,7,11,13];

for i in primeInt:

if(n%i == 0):

mySet.add(i);

s1=set();

s2=set();

count=0;

index=0;

buf1 = input().replace("\n","");

buf1 = buf1.split(" ");

buf2 = input().replace("\n","");

buf2 = buf2.split(" ");

for i in buf1:

IsDiv(int(buf1[index]),s1);

IsDiv(int(buf2[index]),s2);

if(s1 == s2):

count = count+1;

s1.clear();

s2.clear();

index = index+1;

print(count);

## Min abs sum of two

### 기능

### 사용자로부터 int 리스트를 하나 입력받아 리스트 내의 두 숫자를 더한 값의 절댓값들 중 가장 작은 값을 출력합니다.

### 사용변수

* + - 1. buf, i, j, minint, currnum

buf 는 사용자의 입력을 받아 리스트로 변환해 저장한 값입니다. i 와 j 는 모든 숫자쌍을 다 시도해보기 위해서 2차원 배열로 buf를 참조하는데 쓰이는 두 인덱스입니다. minint 는 최종적으로 출력되는 값입니다. 숫자쌍들의 합의 절댓값이 기존의 minint 보다 작을 경우 이 값이 갱신됩니다. currnum 은 2차원배열을 통해 찾은 숫자쌍의 합의 절댓값으로 루프 내에서 현재 참조하고 있는 값입니다.

buf = input().replace("\n","");

buf = buf.split(" ");

i=0;

j=0;

minint=None;

currnum=0;

for i in range(len(buf)):

for j in range(i+1,len(buf)):

currnum = int(buf[i])+int(buf[j]);

currnum = abs(currnum);

if(minint == None):

minint = currnum;

elif(currnum < minint):

print("i=%d j=%d, currnum=%d"%(i,j,currnum));

minint = currnum;

print(minint);

## Max pooling

### 기능

### 사용자로부터 짝수 N 을 입력받습니다. N\*N 크기의 행렬 arr 을 생성해 행렬의 각 칸에 랜덤한 float 형 값을 넣어줍니다. 출력값은 N/2\*N/2 크기의 행렬 halfarr 입니다. 이 행렬의 값은 다음과 같은 방식으로 정해집니다.

arr 를 2\*2 크기의 행렬들로 나누어 생각해봅니다. 그러면 N/2\*N/2 개의 작은 행렬들이 생기게 됩니다. 이 각각의 2\*2 크기의 행렬에 대해 행렬 내의 최대값을 정해서, halfarr의 인덱스에 대입해줍니다.

### 사용변수

* + - 1. rowsize, halfrow, arr, halfarr

rowsize 는 사용자에게 입력받은 N 을 저장하는 변수입니다. halfrow는 N/2 를 저장하고 있는 변수입니다. 이 값은 halfarr의 가로, 세로 크기이기도 합니다. arr 는 N\*N 크기의 행렬이며, 랜덤한 float형 값들이 들어갑니다. halfarr 는 halfrow\*halfrow 크기의 행렬이며, arr 를 3.4.1 에서 설명한 바와 같이 나누어 얻은 최대값들이 들어갑니다.

import numpy

def MaxPool(myArr,row,col):

maxnum = myArr[row][col]

for i in range(2):

for j in range(2):

if(myArr[row+i][col+j] > maxnum):

maxnum = myArr[row+i][col+j]

return maxnum

rowsize = input().split("\n")[0]

while(len(rowsize) == 0):

rowsize = input().split("\n")[0]

rowsize = int(rowsize)

halfrow = rowsize // 2

arr = numpy.random.random((rowsize,rowsize))

for i in range(rowsize):

for j in range(rowsize):

arr[i][j] = round(arr[i][j],2)

print(arr)

halfarr = numpy.zeros(shape=(halfrow,halfrow))

for i in range(halfrow):

for j in range(halfrow):

halfarr[i][j] = MaxPool(arr,2\*i,2\*j)

numpy.set\_printoptions(precision=2)

print(halfarr)

* 코드부

p1.jpynb

def IncreaseX(aList, index):

aList[index] = aList[index]+1

def MaxCounter(aList):

maxint = aList[0]

for i in range(lnum):

if(aList[i] > maxint):

maxint = aList[i]

for i in range(lnum):

aList[i] = maxint;

lnum = input().split("\n")[0]

while(len(lnum) == 0):

lnum = input().split("\n")[0]

lnum = int(lnum)

lista = input().split("\n")[0]

while(len(lista) == 0):

lista = input().split("\n")[0]

lista = lista.split(" ")

listb = [0]\*lnum

for i in range(len(lista)):

lista[i] = int(lista[i])

#list variables type changed to int

for i in range(len(lista)):

if(1<= lista[i] <= lnum):

IncreaseX(listb, lista[i]-1)

#lista[i] 는 몇번째인지를 가리키기 때문에 인덱싱에는 1 작은 값을 사용한다.

elif(lista[i] == (lnum+1)):

MaxCounter(listb)

for i in listb:

print(i,end=" ")

p2.jpynb

def IsDiv(n,mySet):

primeInt = [2,3,5,7,11,13];

for i in primeInt:

if(n%i == 0):

mySet.add(i);

s1=set();

s2=set();

count=0;

index=0;

buf1 = input().replace("\n","");

buf1 = buf1.split(" ");

buf2 = input().replace("\n","");

buf2 = buf2.split(" ");

for i in buf1:

IsDiv(int(buf1[index]),s1);

IsDiv(int(buf2[index]),s2);

if(s1 == s2):

count = count+1;

s1.clear();

s2.clear();

index = index+1;

print(count);

p3.jpynb

buf = input().replace("\n","");

buf = buf.split(" ");

i=0;

j=0;

minint=None;

currnum=0;

for i in range(len(buf)):

for j in range(i+1,len(buf)):

currnum = int(buf[i])+int(buf[j]);

currnum = abs(currnum);

if(minint == None):

minint = currnum;

elif(currnum < minint):

print("i=%d j=%d, currnum=%d"%(i,j,currnum));

minint = currnum;

print(minint);

p4.jpynb

import numpy

def MaxPool(myArr,row,col):

maxnum = myArr[row][col]

for i in range(2):

for j in range(2):

if(myArr[row+i][col+j] > maxnum):

maxnum = myArr[row+i][col+j]

return maxnum

rowsize = input().split("\n")[0]

while(len(rowsize) == 0):

rowsize = input().split("\n")[0]

rowsize = int(rowsize)

halfrow = rowsize // 2

arr = numpy.random.random((rowsize,rowsize))

for i in range(rowsize):

for j in range(rowsize):

arr[i][j] = round(arr[i][j],2)

print(arr)

halfarr = numpy.zeros(shape=(halfrow,halfrow))

for i in range(halfrow):

for j in range(halfrow):

halfarr[i][j] = MaxPool(arr,2\*i,2\*j)

numpy.set\_printoptions(precision=2)

print(halfarr)