Міністерство освіти і науки, молоді та спорту України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №8**

З дисципліни «ПРО»

Виконав: Перевірив:

Студент групи ІО-21 доц. Корочкін О.В.

Коноз А.О.

Дата здачі\_\_\_\_\_\_\_\_\_\_\_\_\_

Захищено з балом\_\_\_\_\_

Київ 2014

**Завдання**

1.23 2.13 3.17

1. F1: E:=A+B+C+D(MA\*MZ).

2. F2: MZ:=Min(MA)\*MB+Max(MT)\*(MX\*MZ).

3. F3: d:=Min(A\*Trans(MB\*MM)+B\*Sort(MC).

**Лістинг коду**

from multiprocessing import Process

import threading

#

# Виконав: Коноз Андрій

# Група: ІО-21

# Дата: 14.10.14

# F1: E:=A+B+C+D(MA\*MZ)

# F2: MZ:=Min(MA)\*MB+Max(MT)\*(MX\*MZ)

# F3: d:=Min(A\*Trans(MB\*MM)+B\*Sort(MC))

#

#

# Головний метод

#

n = 3000

if \_\_name\_\_ == '\_\_main\_\_':

T1 = Process(target=TF1)

T2 = Process(target=TF2)

T3 = Process(target=TF3)

T1.start()

T2.start()

T3.start()

T1.join()

T2.join()

T3.join()

input()

def addVect(vect1, vect2):

result = []

for i in range(len(vect1)-1):

result.append(vect1[i]+vect2[i])

return result

def multMatr(matr1, matr2):

result = [[0 for x in range(len(matr1))] for x in range(len(matr1))]

for i in range(len(matr1)):

for j in range(len(matr1)):

temp = 0

for k in range(len(matr1)):

temp = temp + matr1[i][k]\*matr2[k][j]

result[i][j] = temp

return result

def transpMatr(matr):

result = [[0 for x in range(len(matr))] for x in range(len(matr))]

for i in range(len(matr)):

for j in range(len(matr)):

result[i][j] = matr[j][i]

return result

def findMaxMatr(matr):

max = matr[0][0]

for i in range(len(matr)):

for j in range(len(matr)):

if max < matr[i][j]:

max = matr[i][j]

return max

def findMinMatr(matr):

min = matr[0][0]

for i in range(len(matr)):

for j in range(len(matr)):

if min > matr[i][j]:

min = matr[i][j]

return min

def findMinVect(vect):

min = vect[0]

for i in range(len(vect)):

if min > vect[i]:

min = vect[i]

return min

def addMatr(matr1, matr2):

result = [[0 for x in range(len(matr1))] for x in range(len(matr1))]

for i in range(len(matr1)):

for j in range(len(matr1)):

result[i][j] = matr1[i][j]+matr2[i][j]

return result

def multVectMatr(vect, matr):

result = []

for i in range(len(matr)):

temp = 0

for j in range(len(matr)):

temp = temp + vect[j]\*matr[j][i]

result.append(temp)

return result

def sortVect(vect):

sorted = False

while not sorted:

sorted = True

for i in range(len(vect)-1):

if vect[i] > vect[i + 1]:

sorted = False

hold = vect[i + 1]

vect[i + 1] = vect[i]

vect[i] = hold

return vect

def sortMatr(matr):

for k in range(len(matr)):

sorted = False

while not sorted:

sorted = True

for i in range(len(matr)-1):

if matr[k][i] > matr[k][i + 1]:

sorted = False

hold = matr[k][i + 1]

matr[k][i + 1] = matr[k][i]

matr[k][i] = hold

return matr

def multNumbMatr(a, matr):

for i in range(len(matr)):

for j in range(len(matr)):

matr[i][j] = a\*matr[i][j]

return matr;

def TF1():

print('TF1 started')

vectA = [1 for x in range(n)]

vectB = [1 for x in range(n)]

vectC = [1 for x in range(n)]

vectD = [1 for x in range(n)]

matrA = [[1 for x in range(n)] for x in range(n)]

matrZ = [[1 for x in range(n)] for x in range(n)]

vectRes = addVect(addVect(vectA,vectB),addVect(vectC,multVectMatr(vectD,multMatr(matrA,matrZ))));

print('Result TF1:')

print(vectRes)

print('TF1 finished')

def TF2():

print('TF2 started')

matrA = [[1 for x in range(n)] for x in range(n)]

matrB = [[1 for x in range(n)] for x in range(n)]

matrM = [[1 for x in range(n)] for x in range(n)]

matrX = [[1 for x in range(n)] for x in range(n)]

matrZ = [[1 for x in range(n)] for x in range(n)]

matrRes = addMatr(multNumbMatr(findMinMatr(matrA),matrB),multNumbMatr(findMaxMatr(matrM),multMatr(matrX,matrZ)));

print('Result TF2:')

print(matrRes)

print('TF2 finished')

def TF3():

print('TF3 started')

vectA = [1 for x in range(n)]

vectB = [1 for x in range(n)]

matrB = [[1 for x in range(n)] for x in range(n)]

matrM = [[1 for x in range(n)] for x in range(n)]

matrC = [[1 for x in range(n)] for x in range(n)]

result = findMinVect(addVect(multVectMatr(vectA,transpMatr(multMatr(matrB,matrM))),multVectMatr(vectB,sortMatr(matrC))));

print('Result TF3:')

print(result)

print('TF3 finished')

n = 3000

if \_\_name\_\_ == '\_\_main\_\_':

T1 = Process(target=TF1)

T2 = Process(target=TF2)

T3 = Process(target=TF3)

T1.start()

T2.start()

T3.start()

T1.join()

T2.join()

T3.join()

input()