НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №8

з дісципліни **«**Паралельні та розподілені обчислення**»**

Виконав:

студент 3 курсу

ФІОТ гр. ІО-21

Кузьменко Володимир

Перевірив:

Корочкін О. В.

Київ – 2014 р.

Лабораторная работа N 8. (дополнительно)

ПОТОКИ В ЯЗЫКЕ Python

Цель работы: изучение средств языка Python для работы с задачами (процессами).

Выполнение работы: Разработать программу, содержащую параллельны е потоки, каждый из которых реализует функцию F1, F2, F3 из лабораторной работы номер 1. Требования к созданию потоков и необходимые исследования программы описаны в лабораторной работе 2.

Лістинг програми

###############################################

# #

# Laboratory work #8. Threads in Python #

# #

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

# F2: MC = SORT (MA+MB\*MM) #

# F3: p = MAX(SORT(MS)+MA\*MM) #

# #

# author Kuzmenko Vladimir #

# group IO-21 #

# date 12.11.14 #

# #

###############################################

from multiprocessing import Process

import threading

N=3000

def InputVector():

vector = [1 for col in range(N)]

return vector

def InputMatrix():

matrix = [[1 for row in range(N)] for col in range(N)]

return matrix

def OutputVector(vector):

if N <= 5:

print(vector)

def OutputMatrix(matrix):

if N <= 5:

for i in range(N):

print(matrix[i])

def MultiplyMatrices(matrix1, matrix2):

result = [[0 for row in range(len(matrix1))] for col in range(len(matrix2[0]))]

for i in range(len(matrix1)):

for j in range(len(matrix2[0])):

for k in range(len(matrix2)):

result[i][j] += matrix1[i][k]\*matrix2[k][j]

return result

def MultiplyVectorOnMatrix(vector, matrix):

result = InputVector()

for i in range(N):

result[i] = 0

for j in range(N):

result[i] = result[i] + vector[j]\*matrix[j][i]

return result

def MultiplyVectors(vector1, vector2):

vector = InputVector()

result = 0

for i in range(N):

vector[i] = 0

vector[i] = vector1[i] \* vector2[i]

result = result + vector[i]

return result

def AddVectors(vector1, vector2):

result = InputVector()

for i in range(N):

result[i] = vector1[i] + vector2[i]

return result

def AddMatrix(matrix1, matrix2):

result = InputMatrix()

for i in range (N):

for j in range (N):

result[i][j] = matrix1[i][j] + matrix2[i][j]

return result

def SortVector(vector):

for i in range(N):

for j in range(N):

if (vector[i] > vector[j]):

buffer = vector[i]

vector[i] = vector[j]

vector[j] = buffer

return vector

def SortMatrix(matrix):

for i in range(N):

matrix[i] = SortVector(matrix[i])

def MaxMatr(matrix):

max = matrix[0][0]

for i in range (N):

for j in range (N):

if (matrix[i][j] > max):

max = matrix [i][j]

return max

def F1():

print("1 function started.")

A = InputVector()

B = InputVector()

C = InputVector()

MA = InputMatrix()

MZ = InputMatrix()

MT = MultiplyMatrices(MA, MZ)

VMM = MultiplyVectorOnMatrix(C, MT)

VPV = AddVectors(VMM, B)

E = AddVectors(VPV, A)

OutputVector(E)

print("1 function ended.")

def F2():

print("2 function started.")

MA = InputMatrix()

MB = InputMatrix()

MM = InputMatrix()

MmM = MultiplyMatrices(MB, MM)

AdM = AddMatrix(MA,MmM)

SortMatrix(AdM)

OutputMatrix(AdM)

print("2 function ended.")

def F3():

print("3 function started.")

MA = InputMatrix()

MS = InputMatrix()

MM = InputMatrix()

SortMatrix(MS)

MmM = MultiplyMatrices(MA, MM)

AdM = AddMatrix(MS,MmM)

res = MaxMatr(AdM)

print (res)

print("3 function ended.")

if \_\_name\_\_ == "\_\_main\_\_":

print("Main started.")

process1 = Process(target=F1)

process2 = Process(target=F2)

process3 = Process(target=F3)

process1.start()

process2.start()

process3.start()

process1.join()

process2.join()

process3.join()

print("Main ended.")

input()