## Національний Технічний Університет України

## “Київський Політехнічний Інститут”

## Факультет Інформатики та Обчислювальної Техніки

## Кафедра обчислювальної техніки

#### Лабораторна робота №8

по курсу:

Паралельні та розподілені обчислення

Виконав:

студент 3-го курсу ФІОТ

групи ІО-92

Петрук В.О.

## Київ 2011 р.

# -\*- coding: utf-8 -\*-

#----------------------------------------------------------------------------------------------

# Laboratory work 3

# Petruk Vadim, IO-92

# Variant: 1.13 C = A - B + D

# 2.28 MD = MIN(MA)\*MB\*MC

# 3.14 D = (A + B)\*(MA - MB)

#----------------------------------------------------------------------------------------------

import time

import threading

from InputOutputClass import InputOutput

N = 1000

#----------------------------------------------------------------------------------------------

# Function1: C := A - B + D

def Function1():

print "Function1 is started"

time.sleep(3)

ioObject = InputOutput()

A = ioObject.VectorIn(N);

B = ioObject.VectorIn(N);

C = ioObject.VectorIn(N);

D = ioObject.VectorIn(N);

for i in range(N):

C[i] = A[i] - B[i] + D[i]

if N <= 8:

ioObject.VectorOut(C);

print "Function1 is finished"

#----------------------------------------------------------------------------------------------

# Function2: MD := MIN(MA)\*MB\*MC

def Function2():

print "Function2 is started"

ioObject = InputOutput()

MA = ioObject.MatrixIn(N)

MB = ioObject.MatrixIn(N)

MC = ioObject.MatrixIn(N)

MD = ioObject.MatrixIn(N)

#MD = 0

for i in range(N):

for j in range(N):

MD[i][j]=0;

#MD := MB\*MC

for i in range(N):

for j in range(N):

for k in range(N):

MD[i][j] = MD[i][j]+MB[i][k]\*MC[k][j]

#search MIN(MA)

min\_MA = MA[0][0]

for i in range(N):

for j in range(N):

if MA[i][j] < min\_MA:

min\_MA = MA[i][j]

#MD := MIN(MA)\*MB\*MC

for i in range(N):

for j in range(N):

MD[i][j] = MD[i][j]\*min\_MA

if N<=8 :

ioObject.MatrixOut(MD);

print "Function2 is finished"

#----------------------------------------------------------------------------------------------

# Function3: D := (A + B)\*(MA - MB)

def Function3():

print "Function3 is started"

ioObject = InputOutput()

A = ioObject.VectorIn(N)

B = ioObject.VectorIn(N)

D = ioObject.VectorIn(N)

MA = ioObject.MatrixIn(N)

MB = ioObject.MatrixIn(N)

#C:=A + B

C = ioObject.VectorIn(N)

for i in range(N):

C[i] = A[i]+B[i]

#MC:=MA - MB

MC = ioObject.MatrixIn(N)

for i in range(N):

for j in range(N):

MC[i][j] = MA[i][j] - MB[i][j]

#MD = 0

for i in range(N):

for j in range(N):

D[i]=0

#D := C \* MC

for i in range(N):

for j in range(N):

D[j] = D[j] + C[j]\*MC[j][i]

if N<=8 :

ioObject.VectorOut(D);

print "Function3 is finished"

#----------------------------------------------------------------------------------------------

class Thread1 (threading.Thread):

def run(self):

Function1();

class Thread2 (threading.Thread):

def run(self):

Function2();

class Thread3 (threading.Thread):

def run(self):

Function2();

#----------------------------------------------------------------------------------------------

def main():

print "Main started"

thread1 = Thread1()

thread2 = Thread2()

thread3 = Thread3()

thread1.start()

thread2.start()

thread3.start()

print "Main finished"

**main()**

# -\*- coding: utf-8 -\*-

class InputOutput():

def VectorIn(self,N):

X = [1, ] \* N

return X

#----------------------------------------------------------------------------------------------------------

def VectorOut(self,X):

print X,

print

#----------------------------------------------------------------------------------------------------------

def MatrixIn(self,N):

MX = []

for i in range(N):

MX.append(self.VectorIn(N))

return MX

#----------------------------------------------------------------------------------------------------------

def MatrixOut(self,MX):

for i in range(len(MX)):

print(MX[i])