Andres Aguirre Gonzalez

A01039656

Método de aproximación por rectangulos. Es un calculo aproximado del área bajo la curva de gráfica de coseno.

Para el multi threading utilizo la librería omp, que es una librería que se encarga de del metodo fork-join automaticamente

**Codigo:**

#include <stdio.h>

#include <math.h>

#include <omp.h>

int omp\_get\_thread\_num();

void single(){

int iRec = 0;

printf("Number of rectangles\n");

scanf("%d", &iRec);

double width = 1.0 / iRec;

double x;

double height;

double area = 0.0;

double pi;

int i;

for (i = 0; i < iRec; i++){

x = i \* width;

height = sqrt(1.0 - x \* x);

area += width \* height;

}

pi = 4.0 \* area;

printf("%.8g\n", pi);

return;

}

void multi(){

int iRec = 0;

printf("Number of rectangles\n");

scanf("%d", &iRec);

double width = 1.0 / iRec;

double x;

double height;

double area = 0.0;

double pi;

int i;

#pragma omp parallel

{

#pragma omp for private(x, i, height) reduction(+:area)

for (i = 0; i < iRec; i++){

x = i \* width;

height = sqrt(1.0 - x \* x);

area += width \* height;

}

//printf("Thread rank: %d \n", omp\_get\_thread\_num());

}

pi = 4.0 \* area;

printf("%.8g\n", pi);

return;

}

int main() {

int option = 0;

printf("1. Single thread\n");

printf("2. Multi thread\n");

scanf("%d", &option);

if(option == 1)

single();

else if(option == 2)

multi();

else

printf("Choose between the options\n");

return 0;

}

**Evidencia:**

Hay una mejora de cpu y tiempo usando multi threading

