PONTIFICIA UNIVERSIDAD CATÓLICA MADRE Y MAESTRA
Departamento de Ciencias e Ingeniería.
Escuela de Ingeniería Telemática y Sistema

Programación II
ISC-307

Proyecto Final

Presentado por:
Nicol Ureña (2018-1669)
Junior Hernández (2018-0999)

Presentado a:

Ing. José Luis Alonso Ochoa

Asignatura:

Programación 2

Fecha de entrega – *10 de diciembre de 2021*

SANTIAGO DE LOS CABALLEROS; REPÚBLICA DOMINICANA

Problema 1:
Modificar el programa solución del ejercicio 3.2 Send Receive del tutorial para que el proceso 0 difunda su identificador de proceso (0) al resto de procesos con identificadores pares, siguiendo un anillo de procesos pares, y el proceso 1 haga lo mismo con los procesos impares. Se deben tener en cuenta soluciones con cualquier número de procesos.

Nuestra solución:
Como para este problema era solo la modificación del código solución del ejercicio 3.2, únicamente para determinar que, dependiendo del identificador del procesador, es decir el número del mismo, si este es un identificador par se debe de mostrar en pantalla un 0 en caso contrario debe de ser 1 para los impares. Para esto solo se colocó una variable que acumulara el residuo de la operación modular del identificador del proceso con el número 2 y después en torno a decisiones simples que presentara en pantalla el mensaje correspondiente, dependiendo si el residuo es 0 siendo par o si el residuo es 1 siendo impar.

Salida:

```
C:\Users\Junior\Desktop\ProyectosC\Encabezado1\bin\Debug>mpiexec -n 9 Encabezado1.exe
Soy el proceso 1 y he recibido 1 de los impares
Soy el proceso 2 y he recibido 0 de los pares
Soy el proceso 3 y he recibido 1 de los impares
Soy el proceso 4 y he recibido 0 de los pares
Soy el proceso 5 y he recibido 1 de los impares
Soy el proceso 6 y he recibido 0 de los pares
Soy el proceso 7 y he recibido 1 de los impares
Soy el proceso 8 y he recibido 0 de los pares
```

Problema 2:
Modificar el programa solución del cálculo paralelo del número π (3.3 Calculo de PI) para que los subintervalos de trabajo sean distribuidos por bloques en lugar de cíclicamente entre los procesos. Por ejemplo, si tuviéramos 3 procesos y n = 11 (número de subintervalos), el proceso 0 debería aproximar las integrales numéricas en los primeros 4 subintervalos consecutivos (1,2,3,4), el proceso 1 calcularía las integrales en los siguientes 4 subintervalos (5,6,7,8,) y el proceso 2 calcularía los últimos tres (9,10,11). Se recomienda empezar derivando matemáticamente un método general para repartir por bloques n subintervalos entre P procesos para cualquier n entero positivo. Modificarlo también la solución para que la aproximación a π se obtenga en todos los procesos.

Nuestra solución:
Para el segundo problema se buscaba resolverlo dándole bloques entre 1 y n. Es decir, si tenemos una precisión de 20 (n) y procesos, cada uno contando de 1 a n debe distribuirse por igual a cada proceso, entonces el proceso {0 tendrá: 1, 2, 3, 4, 5}, proceso {1 tendría: 6, 7, 8, 9, 10}, etc. Se declaró una variable con la cual es posible dividir en subintervalos de trabajo, por ejemplo, de 4 procesos presición de 8, rank = 0, variable = 0, contador = 1, contador <= 2, contador++; rank = 1, variable = 2, contador = 3, contador <= 4, contador++; etc etc. Para luego utilizar un bucle for el cual en su interior contiene una pequeña formula aritmética, la cual nos resuelve nuestro problema para el cálculo de la aproximación de “PI”., aquí se realizan los cálculos necesarios para luego sumarse.

Salida:

```
C:\Users\Junior\Desktop\ProyectosC\Encabezado2\bin\Debug>mpiexec -n 9 Encabezado2.exe
introduce la precision del calculo (n > 0): 50
En el proceso 1 el valor de pi es: 0.390919 con una aproximacion local de 2.75067
En el proceso 8 el valor de pi es: 0.232294 con una aproximacion local de 2.9093
En el proceso 7 el valor de pi es: 0.256058 con una aproximacion local de 2.88554
En el proceso 5 el valor de pi es: 0.307088 con una aproximacion local de 2.8345
En el proceso 3 el valor de pi es: 0.356204 con una aproximacion local de 2.78539
En el proceso 6 el valor de pi es: 0.281225 con una aproximacion local de 2.86037
En el proceso 2 el valor de pi es: 0.376254 con una aproximacion local de 2.76534
En el proceso 0 el valor de pi es: 0.398688 con una aproximacion local de 2.7429
En el proceso 4 el valor de pi es: 0.332568 con una aproximacion local de 2.80902
El valor aproximado de PI es: 2.9313, con un error de 0.210296
```

Problema 3:

Modificar el programa solución del cálculo del producto escalar de dos vectores (4.1 Producto Escalar) para que cada proceso inicialice por su cuenta su parte correspondiente del vector B de forma local, de tal forma que no haya necesidad de inicializar todo el vector B en el proceso 0 y repartir sus bloques entre los procesos.

Nuestra solución:

El objetivo de este punto era realizar el cálculo del producto escalar de dos vectores. En un primer lugar se inició el ambiente paralelo con MPI_Init y se obtuvo la cantidad de procesos y el identificador. Si no se especifica una cantidad de elementos se asigna una. Se crean los vectores locales a y b, y se usa el vector A para asignarle valores, luego creamos un conjunto de variables como ilocal y flocal que nos permiten llenar el vector b de manera local, luego se reparten los valores del vector A con un MPI_Scatter para dar paso a la multiplicación de los vectores, que luego pasará a ser sumada en un MPI_Reduce, y obtener el resultado final de este ejercicio.

Salida:

```
C:\Users\Junior\Desktop\ProyectosC\Encabezado3\bin\Debug>mpiexec -n 4 Encabezado3.exe
Como no se ha especificado un numero de elementos, por defecto sera 400
Uso: <ejecutable> <cantidad>
El resultado es de 214134000
```