

**UNIVERSIDAD DEL VALLE DE GUATEMALA**

Paralela, sección 20



# Corto # 5

## Pseudocódigo

Diego García - 22404  
Joaquin Campos – 22155

Guatemala, Agosto 2025

- **Declaración de Variables:**  $n$  = tamaño del

array (Número de pacientes)

`data[]` = array con  $n$  elementos (representa los pacientes y su tiempo de espera)

`total = 0` # Variable para almacenar la suma total de los tiempos de espera

`max = -∞` # Variable para almacenar el valor máximo de tiempo de espera

`min = ∞` # Variable para almacenar el valor mínimo de tiempo de espera

`i` = índice para los bucles

`promedio = 0` # Variable para almacenar el promedio de los tiempos de espera

- **Asignación de Pacientes a Doctores y Cálculo del Tiempo de Espera:**

# Usamos OpenMP para paralelizar el bucle que asigna pacientes a doctores

# y calcula el tiempo de espera

#pragma omp parallel for reduction(+:total) shared(data) Para

`i = 0` hasta `n-1`:

`doctorId = i % número de doctores` # Asignar paciente cíclicamente a los doctores

# Asignar paciente a doctor y calcular el tiempo de espera  
`total += data[i]` # Sumar el tiempo de espera de cada paciente

- **Cálculo del Promedio de los Tiempos de Espera:** `promedio = total / n`

# Promedio = suma total / número de pacientes

- **Encontrar el Máximo y el Mínimo de los Tiempos de Espera:**

# Usamos OpenMP sections para paralelizar el cálculo del máximo y mínimo de tiempos de espera #pragma omp parallel sections

{

# Sección 1: Buscar el valor máximo de tiempo de espera

```

# Sección paralela que encuentra el valor máximo
#pragma omp section
{
    max = data[0] # Inicializar el máximo con el primer valor
Para i = 1 hasta n-1:      Si data[i] > max:
        max = data[i] # Actualizar el máximo si encontramos
un valor mayor
}

```

```

# Sección 2: Buscar el valor mínimo de tiempo de espera
# Sección paralela que encuentra el valor mínimo
#pragma omp section
{
    min = data[0] # Inicializar el mínimo con el primer valor
Para i = 1 hasta n-1:      Si data[i] < min:
        min = data[i] # Actualizar el mínimo si encontramos
un valor menor
}

```

- **Cálculo de la Suma Total de los Tiempos de Espera:**

```

# Usamos reduction para evitar condiciones de carrera mientras
calculamos la suma
#pragma omp parallel for reduction(+:total) shared(data) Para
i = 0 hasta n-1:
    total += data[i] # Sumar los tiempos de espera de los pacientes

```

- **Impresión de los Resultados de la Simulación:**

```

Imprimir "Promedio de los tiempos de espera:", promedio
Imprimir "Valor máximo de espera:", max
Imprimir "Valor mínimo de espera:", min
Imprimir "Suma total de los tiempos de espera:", total

```

Link del repositorio:

<https://github.com/JoaquinCampos22155/corto-5P>