PROYECTO DE MONITOREO DE SENSORES

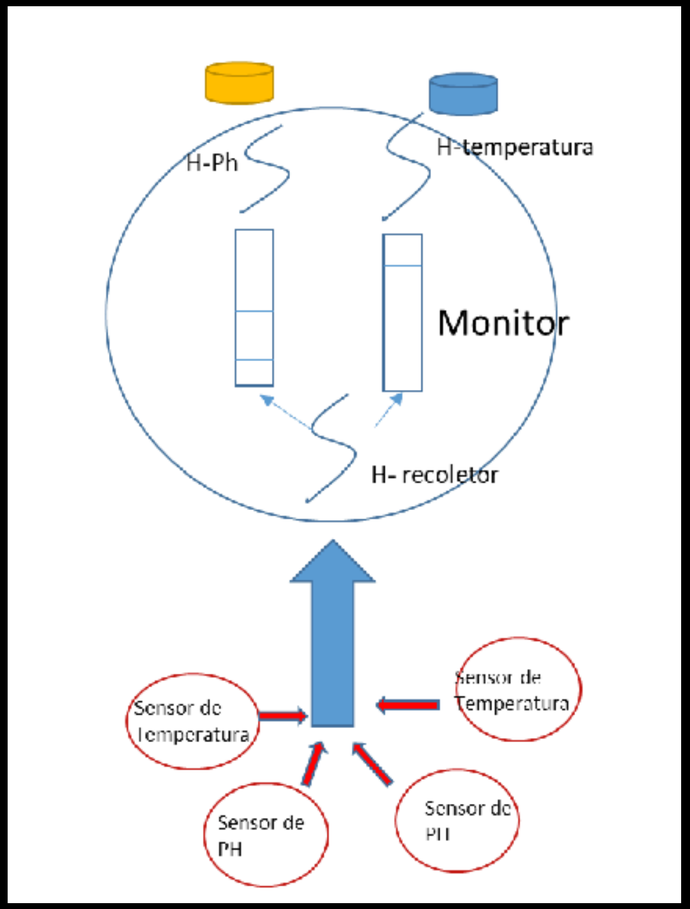
JUAN DAVID RINCÓN

JUAN FELIPE MORALES

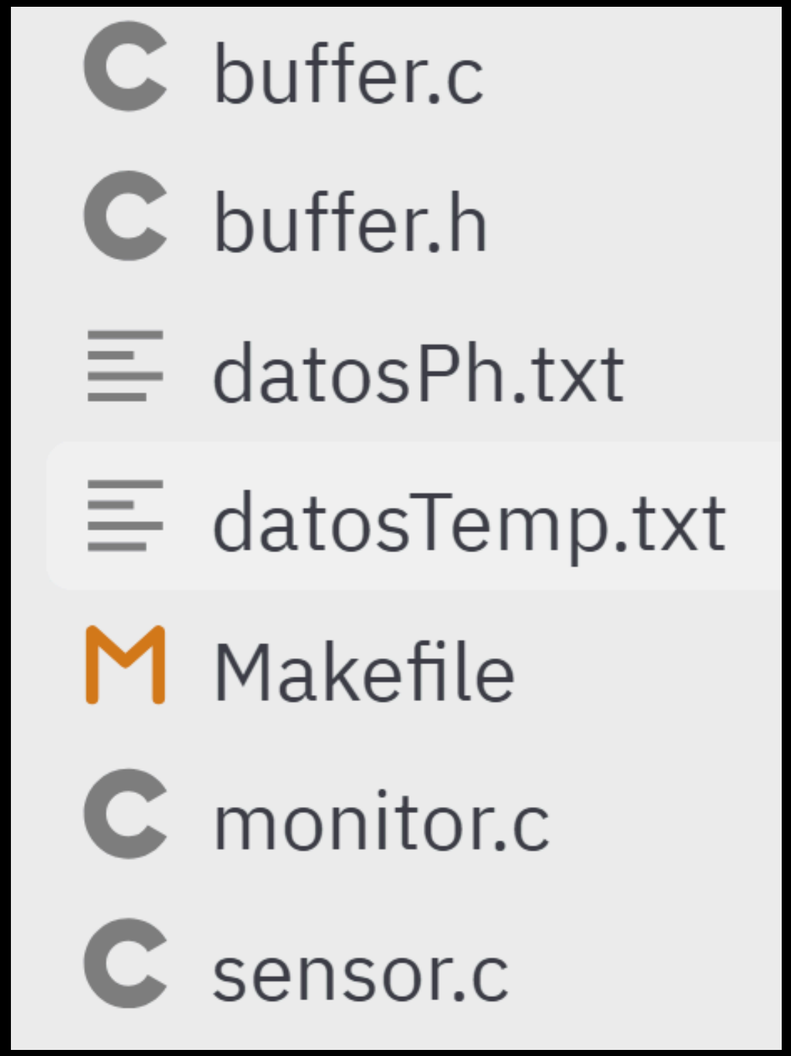
Objetivo del Proyecto

Después de realizar este proyecto, los estudiantes estarán en capacidad de utilizar herramientas
para la comunicación y sincronización de procesos e hilos. Se utilizarán pipes nominales para la
comunicación entre procesos y semáforos para sincronizar los hilos. Adicionalmente, emplearemos
llamadas al sistema para la manipulación de archivos de texto.

Contexto

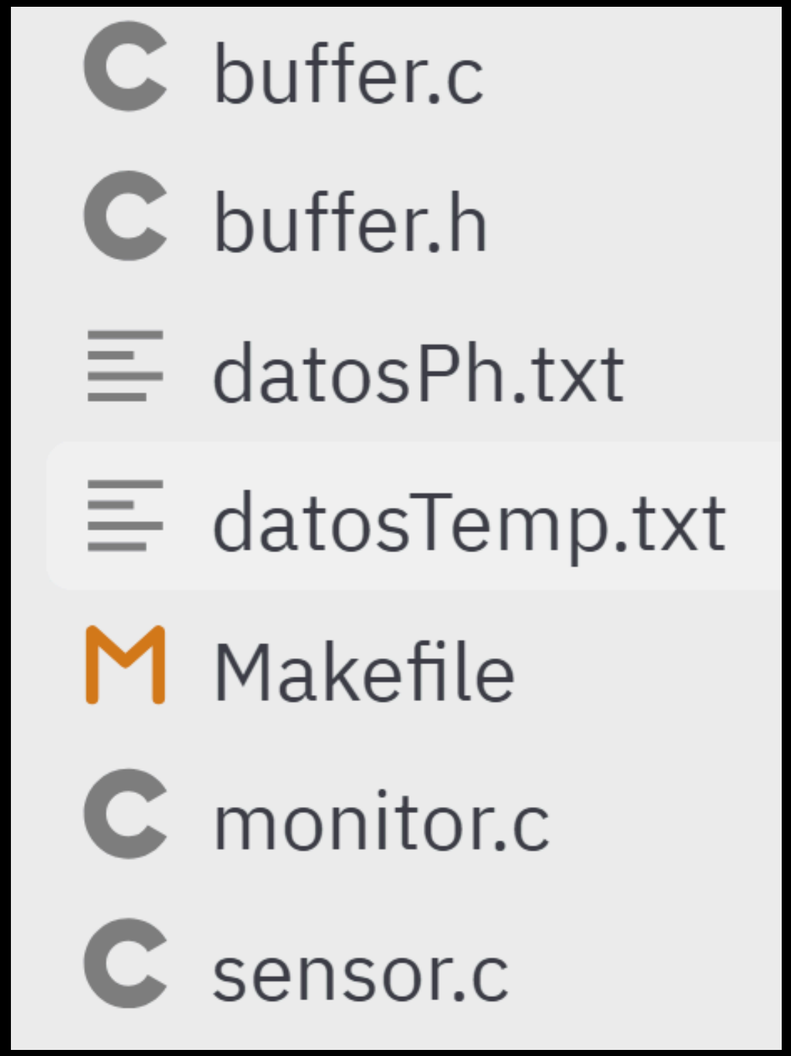

H-Ph
H-temperatura
Monitor
H- recoletor
Sensor de Temperatura
Sensor de Temperatura
Sensor de PH
Sensor de PH

Componentes del proyecto



- **sensor.c :** Este componente se encarga de la lectura del archivo que contiene los datos de pH o temperatura. Además, tiene la tarea de abrir el pipe para la comunicación con el monitor.
- **monitor.c :** Este componente gestiona la lectura y almacenamiento de datos de sensores de temperatura y pH desde un pipe. Se utiliza buffers y semáforos para sincronizar el acceso a los datos, y crea hilos para procesar y registrar los datos en archivos específicos.

Componentes del proyecto



- **buffer.c :** Este archivo implementa las funciones para manejar los buffers para temperatura y pH. Los buffers se controlan mediante semáforos y mutex para sincronizar el acceso. Los hilos recolectores leen datos de un pipe, los almacenan en los buffers y los escriben en archivos.
- **buffer.h :** Contiene la interfaz de las funciones que implementa el componente buffer.c
- **Makefile:** Se encarga de compilar y ejecutar los programas sensor y monitor. Proporciona comandos para ejecutar los sensores y el monitor en segundo plano, leyendo y almacenando datos de temperatura y pH en archivos específicos.

Funcionamiento del Sensor

Manejo de las flags

```c
int flags;
char *sensorType = NULL;
char *timeInterval = NULL;
char *fileName = NULL;
char *pipeName = NULL;
```

```c
// Maneja de banderas mediante argumentos de línea de comandos
while ((flags = getopt(argc, argv, "s:t:f:p:")) != -1) {
  switch (flags) {
  case 's': // Bandera de sensor
    sensorType = argv[optind - 1];
    break;
  case 't': // Bandera de sensor del intervalo de tiempo
    timeInterval = argv[optind - 1];
    break;
  case 'f': // Bandera del nombre del archivo
    fileName = argv[optind - 1];
    break;
  case 'p': // Bandera del nombre del pipe
    pipeName = argv[optind - 1];
    break;
  default: // Mensaje de uso en caso de argumentos incorrectos
    fprintf(
        stderr,
        "Usage: %s -s sensorType -t timeInterval -f fileName -p pipeName\n",
        argv[0]);
    return 1;
  }
}
```

Funcionamiento del buffer

Declaracion de variables

```
// Definiciones de banderas para identificar tipos de datos
#define TEMPERATURA_FLAG "TEMP" // Bandera para datos de temperatura
#define PH_FLAG "PH"            // Bandera para datos de pH

// Declaración de variables y estructuras globales
extern int BUFFER_SIZE;          // Tamaño del buffer
extern sem_t empty_temp, full_temp, empty_ph, full_ph; // Semáforos para controlar el acceso al buffer
extern pthread_mutex_t mutex_temp, mutex_ph;           // Mutex para garantizar la exclusión mutua al acceder al buffer
extern char **buffer_temp;       // Buffer para datos de temperatura
extern char **buffer_ph;         // Buffer para datos de pH
extern int in_temp, out_temp;    // Índices de entrada y salida para el buffer de temperatura
extern int in_ph, out_ph;        // Índices de entrada y salida para el buffer de pH
extern char *file_temp, *file_ph; // Nombres de los archivos de datos de temperatura y pH
```

Funcionamiento del buffer

Funciones

```
// Prototipos de funciones
void ini_buffers();              // Inicializa los buffers
void free_memory_from_buffers();          // Libera la memoria asignada para los buffers
void *recolector(void *param); // Función para recolectar datos
void *ph_thread_collec();      // Función de hilo para recolectar datos de pH
void *temper_thread_collec();  // Función de hilo para recolectar datos de temperatura
```

Resultados para la temperatura

file-temp

```
1   {2024-05-21 14:24:33} 30.000000
2   {2024-05-21 14:24:36} 27.000000
3   {2024-05-21 14:24:45} 21.000000
4   {2024-05-21 14:24:57} 30.000000
5   {2024-05-21 14:25:00} 30.000000
6   {2024-05-21 14:25:06} 30.000000
7   {2024-05-21 14:25:12} 25.000000
8   {2024-05-21 14:25:15} 29.000000
9   {2024-05-21 14:25:24} 24.000000
10  {2024-05-21 14:25:27} 22.000000
11  {2024-05-21 14:25:33} 25.000000
12  {2024-05-21 14:25:36} 31.000000
13  {2024-05-21 14:25:39} 23.000000
14  {2024-05-21 14:25:45} 20.000000
15  {2024-05-21 14:25:48} 22.000000
16  {2024-05-21 14:25:54} 23.000000
17  {2024-05-21 14:25:57} 30.000000
18
```

- Como resultado de la compilación guarda todos los datos correspondientes a la temperatura en un archivo txt llamado File-temp.txt.

Resultados para pH

file-pH

```
1   {2024-05-21 14:29:08} 6.000000
2   {2024-05-21 14:29:11} 6.500000
3   {2024-05-21 14:29:17} 7.000000
4   {2024-05-21 14:29:20} 7.200000
5   {2024-05-21 14:29:23} 7.600000
6   {2024-05-21 14:29:26} 6.100000
7   {2024-05-21 14:29:29} 7.900000
8   {2024-05-21 14:29:32} 7.000000
9   {2024-05-21 14:29:35} 6.300000
10  {2024-05-21 14:29:38} 7.800000
11  {2024-05-21 14:29:44} 6.500000
12  {2024-05-21 14:29:47} 6.000000
13  {2024-05-21 14:29:50} 7.100000
14  {2024-05-21 14:29:53} 8.000000
15  {2024-05-21 14:29:56} 7.700000
16  {2024-05-21 14:29:59} 6.400000
```

- Como resultado de la compilación guarda todos los datos correspondientes del ph en un archivo txt llamado File-ph.txt.

```
17  {2024-05-21 14:30:02} 6.300000
18  {2024-05-21 14:30:05} 6.700000
19  {2024-05-21 14:30:08} 7.200000
20  {2024-05-21 14:30:11} 7.800000
21  {2024-05-21 14:30:14} 6.700000
22  {2024-05-21 14:30:17} 7.100000
23  {2024-05-21 14:30:20} 7.600000
24  {2024-05-21 14:30:23} 7.000000
25  {2024-05-21 14:30:26} 6.300000
26  {2024-05-21 14:30:29} 7.900000
27  {2024-05-21 14:30:32} 7.100000
28  {2024-05-21 14:30:35} 7.200000
29
```

Thanks!