Practical 12

Error Detection : CRC

* Objective

-> The objective of this code is to demonstrate inter-process communication (IPC) using a FIFO (named Pipe). The parent process (sender) sends a CRC divisor in binary format to the child process (receiver). The receiver then validates the divisor, ensuring it starts and ends with '1' and is at least 2 bits long, and prints the result.

* Code

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <fcntl.h>
#include <sys/wait.h>

#define FIFO_NAME "/tmp/crc_fifo"
#define MAX_SIZE 32

int is_valid_divisor(const char *divisor)
{
    int len = strlen(divisor);

    if (len < 2 || divisor[0] != '1' || divisor[len-1] != '1')
```

```
    {
        return 0;
    }
    return 1;
}

int main()
{
    int fd;
    char divisor[MAX_SIZE];
    pid_t pid;

    mkfifo(FIFO_NAME, 0666);

    pid = fork();

    if (pid < 0)
    {
        perror("Fork failed");
        exit(1);
    }

    if (pid == 0)
    {
        sleep(1);
        printf("\n[Receiver] Waiting for divisor...\n");

        fd = open(FIFO_NAME, O_RDONLY);
        if (fd == -1)
        {
            perror("Error opening FIFO for reading");
```

```
        exit(1);
    }

    read(fd, divisor, MAX_SIZE);
    close(fd);

    printf("[Receiver] Received divisor: %s\n", divisor);

    if (is_valid_divisor(divisor))
    {
        printf("[Receiver] ☑ Valid CRC divisor.\n");
    }
    else
    {
        printf("[Receiver] ☒ Invalid CRC divisor.
    }

    unlink(FIFO_NAME);
}

else
{
    printf("[Sender] Enter CRC Divisor in binary
            format: ");
    scanf("%s", divisor);

    fd = open(FIFO_NAME, O_WRONGLY);
    if (fd == -1)
    {
        perror("Error opening FIFO for writing");
        exit(1);
    }
```

```
    write(fd, divisor, strlen(divisor)+1);
    close(fd);

    printf("[Sender] Divisor ~~send~~sent to
            receiver.\n");

    wait(NULL);
}

return 0;
}
```

* Output

[Sender] Enter CRC Divisor in binary format : 1111
[~~Sender~~Sender] Divisor sent to receiver
[Receiver] Waiting for divisor...
[Receiver] Received divisor : 1111
[Receiver] ☑ Valid CRC divisor.

* Learning Outcome

1. Fork a process to create a parent (sender) and child (receiver) process.

2. Send data (CRC divisor) from the parent to the child using a FIFO