CST8234 - C Programming

# LAB 6

# PRACTISE LAB

# NOT MARKED

## LAB OBJECTIVE

By completing this lab, you will learn to:

- Creating multiple process
- Creating pipe for IPC
- Reading and writing to pipes

LAB INSTRUCTIONS:

## STATEMENT OF THE PROBLEM:

In this lab we will look at very basic inter-process communication technique. We will build a program that simulates a sensor that reads a temperature in a machine, and sends it to another process for reporting.

We will have a parent process that is responsible for reading the temperature from the sensor, which is the child process, and prints it to the screen. The child on the other hand will keep writing a new temperature reading every second. The child will not print the data at all, leaving that responsibility to the parent process.

In order to simulate the solution, we will assume that the temperature reading will start at 20.1 degrees celsius, then it will keep increasing by 0.1 degrees every second.

## REQUIREMENTS:

The Following is a complete list of the program requirements:

1. Declare an array that will hold the pipe descriptors of the appropriate type. Call that array `pipefd`.
2. Create a pipe using `pipefd`, the array declared above, and save the pipe descriptors in it.
3. Create two process and store the pid returned back
4. In the parent process:
    1. Make sure to close the writing end of the pipe and store the reading end descriptor in a variable that is called `fdRead`.
    2. Loop infinitely and keep reading from the child, storing the readings in a buffer variable.
    3. Print the reading whenever it is available.
5. In the child process:
    1. Make sure to close the reading end of the pipe and store the writing end descriptor in a variable that is called fdWrite.
    2. Declare two double variables, one for the initial temperature value which is set to 20.1. A second constant that represent the increase value which is set to 0.1.
    3. Loop infinitely and keep writing the temperature value to the pipe using a buffer of the appropriate type.

4. Recalculate the temperature value to be the old value + the increase constant.
5. Make the process sleep for 1 second before it loops again.

The following is a sample of the output of the file.