Operating Systems

Assignment 3

Aviral Chauhan 2021029

Question 1:

1. This is the code for the dining philosophers problem. It creates 5 threads, each representing a philosopher. Philosopher with even id picks up their right fork first and philosopher with odd id picks up their left fork first in order to eat. The philosopher will eat for a short period of time and then return the forks before going back to thinking.

2. In this part, the code is implemented differently. Each philosopher tries to pick up their left and right forks in order to eat, but if the other philosopher has already picked up one of the forks, the philosopher will go back to thinking. The philosopher will eat for a short period of time and then return the forks before going back to thinking. The program uses semaphores to control access to the forks and prevent deadlock. It also keeps track of how many times each philosopher has eaten and prints this information after each philosopher finishes eating.

a. There are 5 philosophers and 5 forks, and each philosopher needs two forks to eat. But now there are 2 bowls to eat from. The program uses semaphores to control access to the forks and prevent deadlock.
b. The question is the same but in this instead of using semaphores I used a normal while and a sleep statement.

Question 2:
In this question, we are generating 50 random for inter process communication. The first program is sending 5 strings at a time to the second program with its ID and the second program is returning the maximum ID. We are using unix domain sockets, FIFO's and shared memory. We are also measuring the time taken for this process to complete.

Question 3:

In this question I have implemented a kernel system module as a module in which it reads the entries of the process task_struct which prints the values of the following field: pid, user id, process group id (pgid) and command path.