Name: P.MALREDDY

Reg-No: 192372015

12. Design a C program to simulate the concept of Dining-Philosophers problem

Aim

The **Dining Philosophers Problem** is a classic synchronization problem that illustrates how to allocate limited resources (e.g., forks) among multiple processes (e.g., philosophers) to avoid deadlock and ensure fairness.

Algorithm

1. Philosophers alternate between **thinking** and **eating**.
2. Each philosopher needs two forks (shared resources) to eat.
3. Use a synchronization mechanism (e.g., semaphores or mutexes) to prevent deadlocks and ensure mutual exclusion.

Procedure

1. Initialize a mutex or semaphore for each fork.
2. Create threads for each philosopher.
3. Implement the **thinking**, **picking up forks**, **eating**, and **putting down forks** states.
4. Use synchronization to avoid deadlock or starvation.

 Code:

```c
#include <stdio.h>

#include <pthread.h>

#include <unistd.h>

void *print_message(void *thread_id) {

  int tid = *(int *)thread_id;

  printf("Thread %d is running\n", tid);

  sleep(1);  // Simulate work

  printf("Thread %d has finished\n", tid);

  return NULL;
```

```
    }

    int main() {

        pthread_t threads[3];

        int thread_ids[3];

        for (int i = 0; i < 3; i++) {

            thread_ids[i] = i + 1;

            pthread_create(&threads[i], NULL, print_message, &thread_ids[i]);

        }

        for (int i = 0; i < 3; i++) {

            pthread_join(threads[i], NULL);

        }

        printf("All threads have completed execution.\n");

        return 0;

    }
```
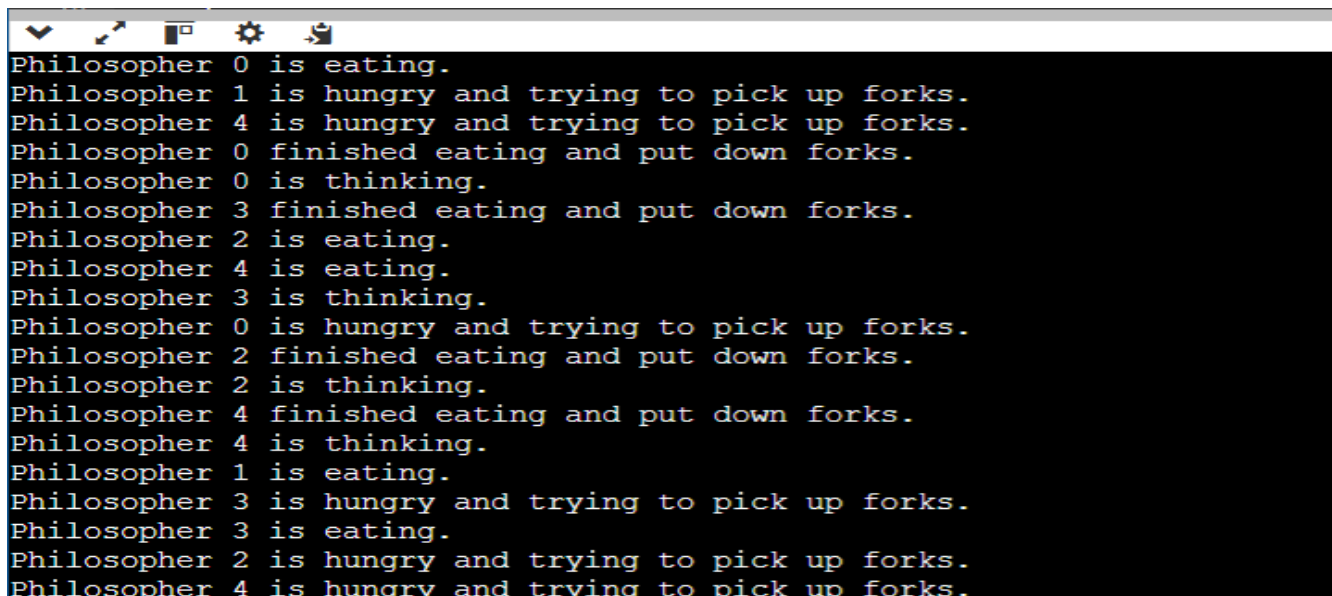
Result

The program simulates philosophers alternately **thinking** and **eating** while ensuring that no two adjacent philosophers eat simultaneously, avoiding deadlock.

Output:

```
Philosopher 0 is eating.
Philosopher 1 is hungry and trying to pick up forks.
Philosopher 4 is hungry and trying to pick up forks.
Philosopher 0 finished eating and put down forks.
Philosopher 0 is thinking.
Philosopher 3 finished eating and put down forks.
Philosopher 2 is eating.
Philosopher 4 is eating.
Philosopher 3 is thinking.
Philosopher 0 is hungry and trying to pick up forks.
Philosopher 2 finished eating and put down forks.
Philosopher 2 is thinking.
Philosopher 4 finished eating and put down forks.
Philosopher 4 is thinking.
Philosopher 1 is eating.
Philosopher 3 is hungry and trying to pick up forks.
Philosopher 3 is eating.
Philosopher 2 is hungry and trying to pick up forks.
Philosopher 4 is hungry and trying to pick up forks.
```