Code :

#include<stdio.h>

#include<semaphore.h>

#include<pthread.h>

#include<unistd.h>

#define N 5 //Number of philosopher are 5

#define THINKING 0 //Three States Thinking, Hungry and Eating

#define HUNGRY 1

#define EATING 2

#define LEFT (ph\_num+4)%N //Two conditions for picking the fork/chopstick

#define RIGHT (ph\_num+1)%N

sem\_t mutex;

sem\_t S[N];

int count[5];

int FOOD = 0;

void \* philospher(void \*num);

void take\_fork(int);

void put\_fork(int);

void test(int);

int state[N]; //Checks the state of the philosopher

int phil\_num[N]={0,1,2,3,4}; //Sequence for the philosopher's

int main() //main function

{

int i;

pthread\_t thread\_id[N]; //declaration of threads

sem\_init(&mutex,0,1); //Use of semaphores(binary)

for(i=0;i<N;i++)

sem\_init(&S[i],0,0);

for(i=0;i<N;i++) //Creation of threads for all philosophers

{

pthread\_create(&thread\_id[i],NULL,philospher,&phil\_num[i]);

}

for(i=0;i<N;i++)

pthread\_join(thread\_id[i],NULL); // waits for the thread to exit

for(i=0;i<N;i++)

printf("Philospher %d ate %d \n",i,count[i]);

// outputs food count for each philosophers

printf("\n");

}

void \*philospher(void \*num)

{

while(FOOD <= 20) //use of while condition

{

int \*i = num; //picking up and picking down fork condition

usleep(10000);

take\_fork(\*i);

put\_fork(\*i);

}

}

void take\_fork(int ph\_num) //hungry state condition

{

sem\_wait(&mutex);

state[ph\_num] = HUNGRY;

test(ph\_num);

sem\_post(&mutex);

sem\_wait(&S[ph\_num]);

usleep(10000);

}

void test(int ph\_num)

{

if (state[ph\_num] == HUNGRY && state[LEFT] != EATING && state[RIGHT] != EATING)

{ //eating state condition

state[ph\_num] = EATING; //condition for checking the availability of forks(right&left)

usleep(20000);

sem\_post(&S[ph\_num]);

}

}

void put\_fork(int ph\_num) //thinking state condition

{

sem\_wait(&mutex);

state[ph\_num] = THINKING;

count[ph\_num]++;

FOOD++;

test(LEFT);

test(RIGHT);

printf("#Eating Count = %d \n", FOOD);

int i;

for(i=0;i<5;i++){ //setting the states for the philospher

if(state[i]==EATING)

printf("Philosopher %d is eating\n", i);

else if(state[i]==HUNGRY)

printf("Philosopher %d is waiting and calling pickup()\n", i);

else if(state[i]==THINKING)

printf("Philosopher %d is thinking\n", i);

}

sem\_post(&mutex);

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*