**Producer-consumer problem:**

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <semaphore.h>

#define BUFFER\_SIZE 5

sem\_t full,empty;

pthread\_mutex\_t mutex;

int array[BUFFER\_SIZE];

int in=0;

int out=0;

void \*producer(void \*data)

{

int i,num,id;

id=(int)data;

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

{

sem\_wait(&empty);

pthread\_mutex\_lock(&mutex);

num=id;

array[in]=num;

printf("\n\t Producer =%d produced value = %d stored at location=%d ",id,array[in],in);

in=(in+1)%BUFFER\_SIZE;

pthread\_mutex\_unlock(&mutex);

sem\_post(&full);

sleep(2);

}

}

void \*consumer(void \*data)

{

int i,num,r;

do

{

sem\_wait(&full);

pthread\_mutex\_lock(&mutex);

num=array[out];

printf("\n\t\t\t Consumer consumed value = %d from location=%d",num,out);

out=(out+1)%BUFFER\_SIZE;

pthread\_mutex\_unlock(&mutex);

sem\_post(&empty);

sleep(3);

sem\_getvalue(&empty,&r);

}while(r!=BUFFER\_SIZE);

}

int main()

{

pthread\_t tid[4],tid1;

int i;

sem\_init(&full,0,0);

sem\_init(&empty,0,5);

for(i=0;i<4;i++)

pthread\_create(&tid[i],NULL,producer,(void \*)i);

pthread\_create(&tid1,NULL,consumer,NULL);

for(i=0;i<4;i++)

pthread\_join(tid[i],NULL);

pthread\_join(tid1,NULL);

return 0;

}

