Application to demonstrate process/thread synchronization using semaphores and mutex. Implement, reader-writer or producer-consumer.

1) Reader-writer

#include<stdio.h> #include<stdlib.h> #include<pthread.h> #include<semaphore.h>

int count=0,rcount=0; sem\_t mutex,wr;

void\* writer(void \*p){ int\* i =(int\*)p; sem\_wait(&wr);

printf("\nWriter %d writes page number %d",\*i,++count); sem\_post(&wr);

}

void\* reader(void\* p){ int\* i =(int\*)p; sem\_wait(&mutex); rcount++; if(rcount==1) sem\_wait(&wr); sem\_post(&mutex);

printf("\nReader %d reads page number %d ",\*i,count); sem\_wait(&mutex);

rcount--; if(rcount==0) sem\_post(&wr); sem\_post(&mutex);

}

int main(){ sem\_init(&mutex,0,1);

sem\_init(&wr,0,1); int a[6]={1,2,3,1,2,3}; pthread\_t p[6];

for(int i=0;i<3;i++) pthread\_create(&p[i],NULL,writer,&a[i]); for(int i=3;i<6;i++) pthread\_create(&p[i],NULL,reader,&a[i]); for(int i=0;i<6;i++) pthread\_join(p[i],NULL);

}

Producer consumer:

#include<stdio.h> #include<semaphore.h> #include<pthread.h>

#include<stdlib.h> #define buffersize 10 pthread\_mutex\_t mutex;

pthread\_t tidP[20],tidC[20]; sem\_t full,empty;

int counter;

int buffer[buffersize];

void initialize()

{

pthread\_mutex\_init(&mutex,NULL); sem\_init(&full,1,0); sem\_init(&empty,1,buffersize); counter=0;

}

void write(int item)

{

buffer[counter++]=item;

}

int read()

{

return(buffer[--counter]);

}

void \* producer (void \* param)

{

int waittime,item,i; item=rand()%5; waittime=rand()%5; sem\_wait(&empty); pthread\_mutex\_lock(&mutex);

printf("\nProducer has produced item: %d\n",item); write(item);

pthread\_mutex\_unlock(&mutex); sem\_post(&full);

}

void \* consumer (void \* param)

{

int waittime,item; waittime=rand()%5; sem\_wait(&full); pthread\_mutex\_lock(&mutex); item=read();

printf("\nConsumer has consumed item: %d\n",item); pthread\_mutex\_unlock(&mutex); sem\_post(&empty);

}

int main()

{

int n1,n2,i; initialize();

printf("\nEnter the no of producers: "); scanf("%d",&n1);

printf("\nEnter the no of consumers: "); scanf("%d",&n2);

for(i=0;i<n1;i++)

pthread\_create(&tidP[i],NULL,producer,NULL); for(i=0;i<n2;i++)

pthread\_create(&tidC[i],NULL,consumer,NULL); for(i=0;i<n1;i++)

pthread\_join(tidP[i],NULL); for(i=0;i<n2;i++)

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

//sleep(5); exit(0);

}