# **Operating Systems LAB 04**

**Objective: Understand Inter-process communication**

**Exercise 1. Race Condition & Critical Section**

|  |
| --- |
| 1. **#include <pthread.h>** 2. **#include <stdio.h>** 3. **//#define N (32\*1024\*1024)** 4. **#define N 32** 5. **long long sum = 0;** 6. **//pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER;**  * Use **time ./[file]** to run the program multiple times and observe results. * Now comment line 4 and uncomment line 3. * Run it multiple times and explain if you observe any difference. * Identify the critical region * Fix the problem * Use **time ./[file]** to run the program multiple times and observe results. * What difference did you observe both in time and accuracy? * explain * What is the effect of moving the mutex outside of the loop?  1. **void \*COUNT(void \*arg)** 2. **{** 3. **int value = \*(int\*)arg;** 4. **int i;** 5. **for (i = 0; i < N; i++)** 6. **{** 7. **//pthread\_mutex\_lock(&mutex);** 8. **sum +=value;** 9. **//pthread\_mutex\_unlock(&mutex);** 10. **}** 11. **pthread\_exit(NULL);** 12. **}** 13. **int main()** 14. **{** 15. **pthread\_t T1;** 16. **int INC = 1;** 17. **pthread\_create(&T1, NULL, COUNT, &INC);** 18. **pthread\_t T2;** 19. **int DEC =-1;** 20. **pthread\_create(&T1, NULL, COUNT, &DEC);** 21. **pthread\_join(T1, NULL);** 22. **pthread\_join(T2, NULL);** 23. **printf("Sum = %lld \n", sum);** 24. **}** |

**Exercise 2. Write a c program that demonstrates inter-process communication (Any 2)**

|  |  |
| --- | --- |
| * PIPES AND MESSAGES * **MESSAGE QUEUES** * SHARED MEMORY | * DINNING PHILOSOPHER PROBLEM * SLEEPING BARBER PROBLEM. * PRODUCER-CONSUMER PROBLEM USING SEMAPHORES |

|  |  |
| --- | --- |
| **Example: MESSAGE QUEUE** | |
| **Sender** | **Receiver** |
| #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <sys/ipc.h>  #include <sys/shm.h>  #include <sys/types.h>  #include <sys/msg.h>  struct msgbuf  {  long mtype;  char msgtxt[200];  };  int main(void)  {  struct msgbuf msg;  int msgID;  key\_t key;  if((key = ftok("FILENAME",'b')) ==-1)  {  perror("key");  exit(1);  }  if((msgID=msgget(key,0644|IPC\_CREAT))==-1)  {  perror("msgid");  exit(1);  }  printf("MessageID(%d) %d\n", key, msgID);  printf("Enter the Text \n");  msg.mtype=1;  while(gets(msg.msgtxt),!feof(stdin))  {  if(msgsnd(msgID, &msg, sizeof(msg),0)==-1)  {  perror("msgsnd");  exit(1);  }  }  if(msgctl(msgID,IPC\_RMID, NULL)==-1)  {  perror("msgctl");  exit(1);  }  return 0;  } | #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <sys/ipc.h>  #include <sys/shm.h>  #include <sys/types.h>  #include <sys/msg.h>  struct msgbuf  {  long mtype;  char msgtxt[200];  };  int main(void)  {  struct msgbuf msg;  int msgID;  key\_t key;  if((key = ftok("FILENAME ",'b')) ==-1)  {  perror("key");  exit(1);  }  printf("Key: %d\n", key);  if((msgID=msgget(key,0644))==-1)  {  perror("msgid");  printf("Error.MessageID");  exit(1);  }  printf("Receiving ... \n");  for(;;)  {  if(msgrcv(msgID, &msg, sizeof(msg),1,0)==-1)  {  perror("msgrcv");  printf("Error.Msgrcv");  exit(1);  }  printf("Received: %s\n",msg.msgtxt);  }  return 0;  } |