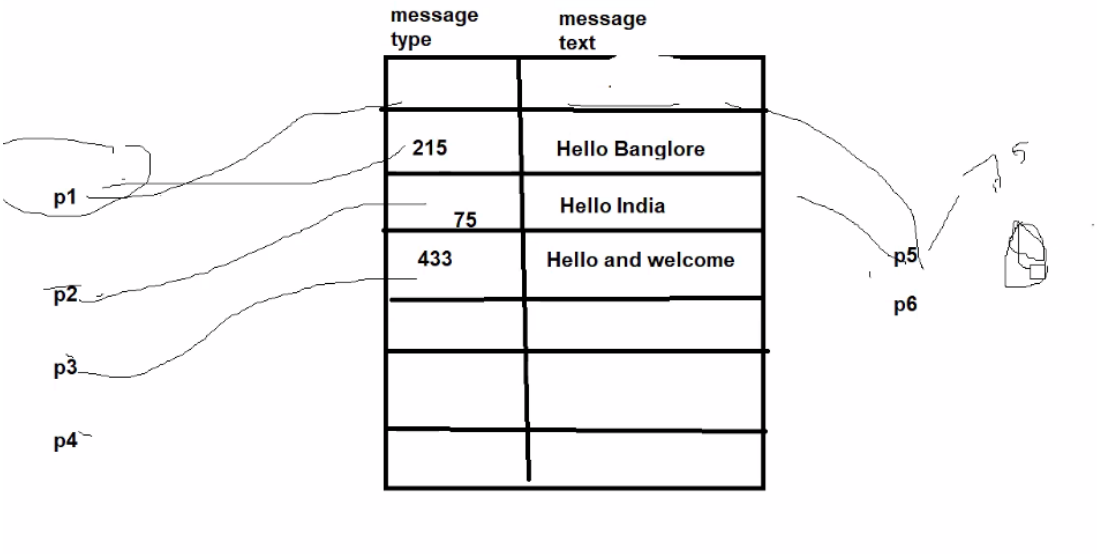
MESSAGE QUEUE



p5(reader) should know the message type.

if p5 doesn't know any message type, it gives 0.

**The moment one process reads msg from msg queue,

that row is deleted.

**If reader doesn't know msg type, then FIFO entry read

and deleted.

**If reader knows msg type, that row entry is deleted

When msg queue is empty and process wants to read

a) Wait ->p6 goes to pending state

b) No Wait ->p6 can't read & can't wait & returns -1

While writing , when msg queue is full

a)Wait ->p4 goes to pending state and waits

until space is created in msg queue

b)No Wait ->p4 cant write & cant wait ..returns -1

Every msg queue has unique msg id

P1 and p2 cant decide the id of msg queue

KEY:given by programmer to the processes

All process use a common key

Key is assigned to msg queue id

Limitations

a)Max no of msg queues=32k

b)Max size of each message =8192 bytes

c)Default max size of queue=16384 bytes

d)Broadcasting not possible as row is deleted

after reading

e) Message queue is present in kernel area

and processes in user area

Message forwarding take time=>slow

Note:0777 (UGO)

7=>111 rwx set to 1

4=>100 rwx .. only read permission

```
prashanth@prashanth-VirtualBox:~/carl_zeiss$ ipcs -q

------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages

prashanth@prashanth-VirtualBox:~/carl_zeiss$ ipcs -m

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x00000000 32768      prashanth  600        7127040    2          dest
0x00000000 4          prashanth  600        16384      1          dest
0x00000000 8          prashanth  600        7372800    2          dest
0x00000000 10         prashanth  600        67108864   2          dest
0x00000000 13         prashanth  600        7372800    2          dest
0x00000000 16         prashanth  600        524288     2          dest
0x00000000 17         prashanth  600        524288     2          dest
0x00000000 58         prashanth  600        3686400    2          dest
0x00000000 61         prashanth  600        524288     2          dest
0x00000000 62         prashanth  600        524288     2          dest
0x00000000 63         prashanth  600        524288     2          dest

prashanth@prashanth-VirtualBox:~/carl_zeiss$ ipcs -s

------ Semaphore Arrays --------
key        semid      owner      perms      nsems
```

msgget

msgsnd

msgrcv

Advantages

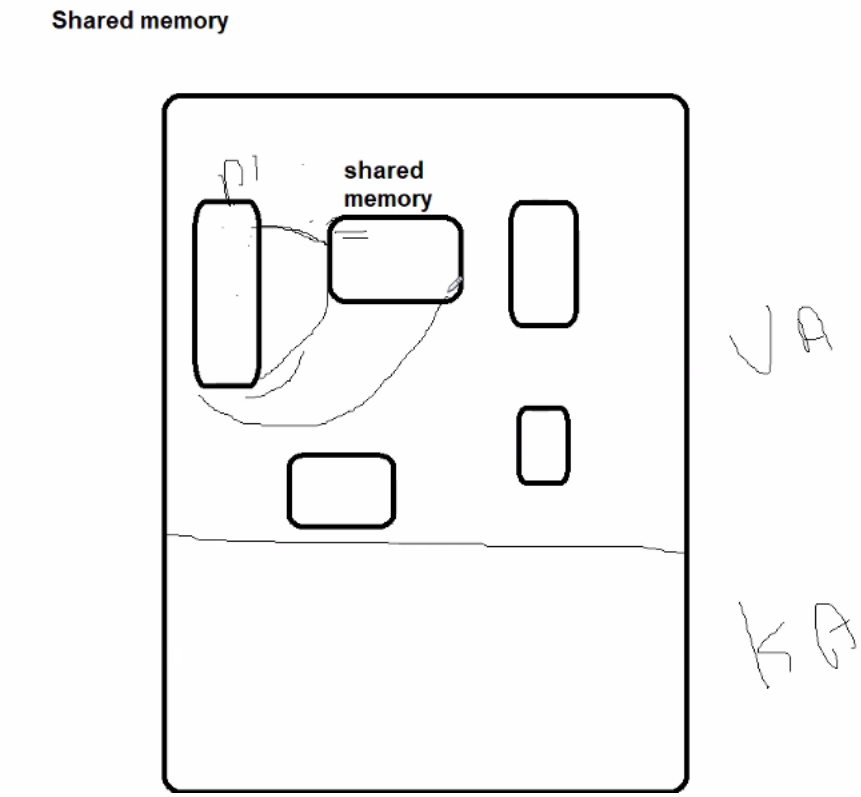
1)Multi message transfer

2)Each msg has specific boundary

3) Unrelated processes can communicate

---

2)SHARED MEMORY



S.M. Present in User space

Process attatches itself to a shared memory & writes

Faster than msg queue as no system call involved after attatching

Then process process detaches

Only one process can write

Many process can attatch at the same time(read)

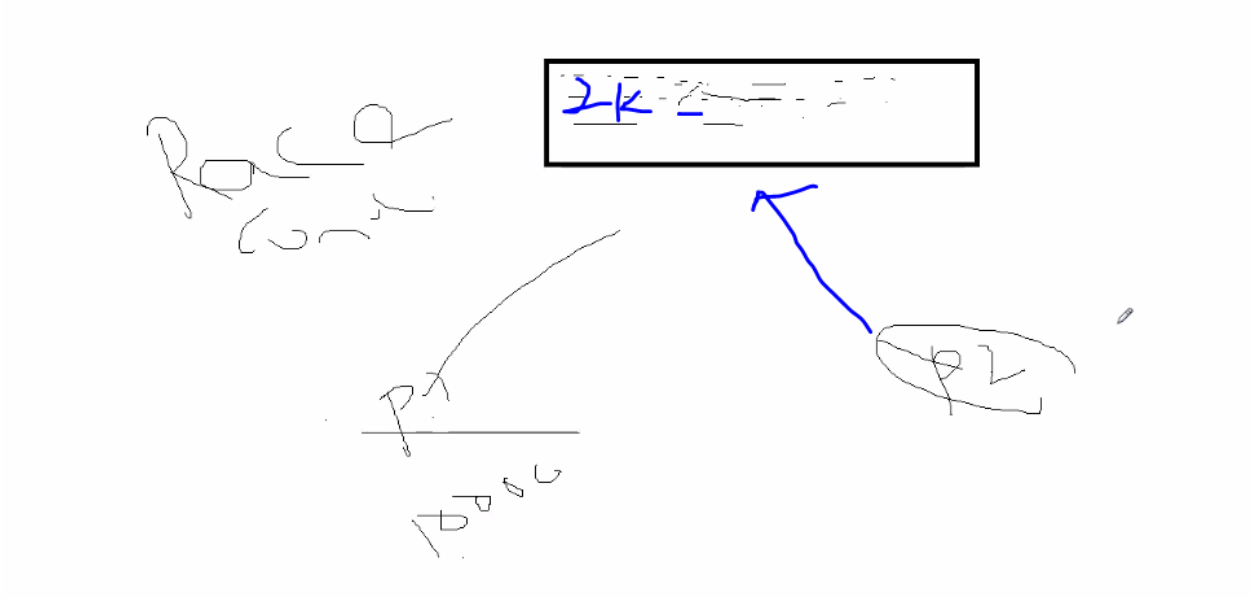
Note:Data in shared memory is never deleted, only overwritten

Limitations

1)Only 1 entity , we are writing

2)When 2 processes are writing simultaneously

Data collision / Race condition

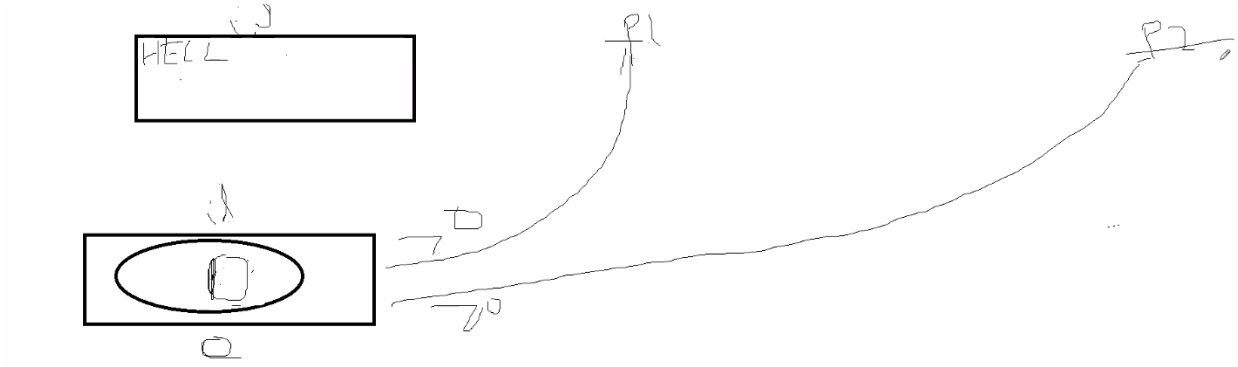


---

3)SEMAPHORES -avoids RACE Condition using SIGNALS

->Semaphore is like a traffic signal =>Red(0) or Green light(1)

p1 and p2



p1 wants to write in shared memory

p1 checks if S is 0 or 1

if s is 1 , then p1 goes to shared memory and changes its value to S=S-1

When p1 is executing , p2 checks if S is 0 or 1

Its set to 0 now

So p2 has to wait now

Once task of p1 is done , it ends and sets S=1

Now p2 can come inside shared memory

---

msgget

shmget

semget->creates an array of semaphores

because requirement of s is much much higher

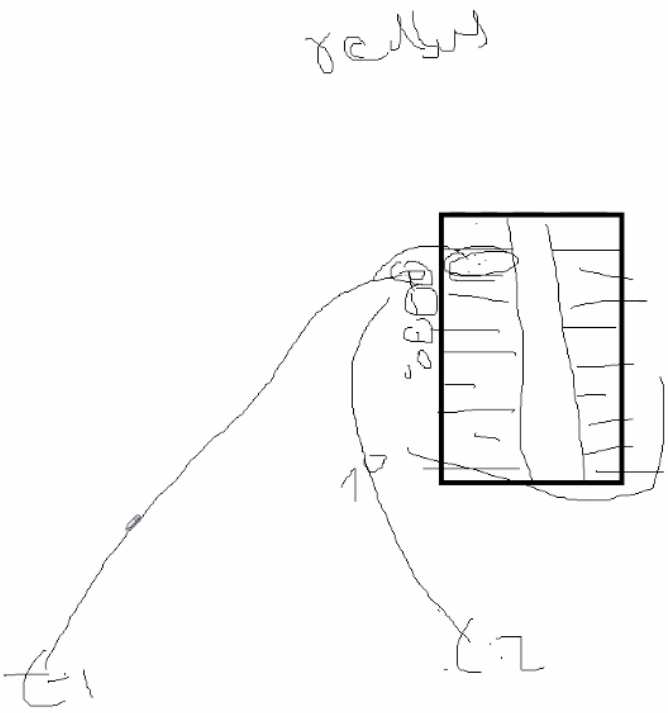
Why array ?

scenario:bus booking

msgget

shmget

semget



40 seats =>40 semaphores

C1 finds S of Seat1 is >0 , goes inside and makes S<0

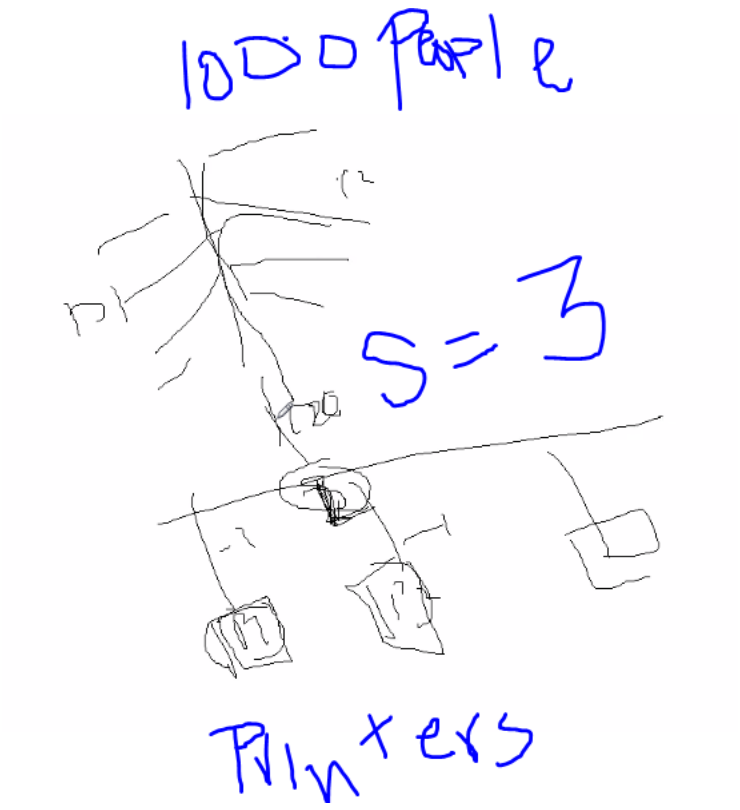
Now , C2 either has to wait until Seat 1 semaphore>0 or book another seat except Seat1

Note: with 1 key and id =>1 array of semaphore

If 100 buses with 40 seats =>We need 100 array of semaphores with 100 keys

---

Counting semaphores eg



1000 people and 3 instances of printer

S=3

AVOIDING DEADLOCK

Either a process will have both printer and file

or none=>Complete Ownership

LIBRARIES (Static vs Dynamic)

how to create our own libraries ?

---

static:in compilation only , all fn definitions brought to executable object file

After compiling , we can delete the library

During runtime , we dont want the library

Limitation:size of executable file is very big

```
evenodd.c  evenodd.o  myfunctions.h  palindrome.c  palindrome.o  perfect.c  perfect.o  prime.c  prime.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ rm evenodd.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ rm perfect.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ rm prime.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.o  myfunctions.h  palindrome.c  palindrome.o  perfect.o  prime.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ rm palindrome.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.o  myfunctions.h  palindrome.o  perfect.o  prime.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$
```

```
evenodd.c  myfunctions.h  palindrome.c  perfect.c  prime.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ gcc -c evenodd.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.c  evenodd.o  myfunctions.h  palindrome.c  perfect.c  prime.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ gcc -c palindrome.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.c  evenodd.o  myfunctions.h  palindrome.c  palindrome.o  perfect.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ gcc -c perfect.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.c  evenodd.o  myfunctions.h  palindrome.c  palindrome.o  perfect.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ gcc -c prime.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
```

```
evenodd.c  myfunctions.h  palindrome.c  perfect.c  prime.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ gcc -c evenodd.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.c  evenodd.o  myfunctions.h  palindrome.c  perfect.c  prime.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$
```

```
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ vi mainapp.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ gcc mainapp.c
/tmp/ccTJdwYt.o: In function `main':
mainapp.c:(.text+0x99): undefined reference to `isEvenOdd'
mainapp.c:(.text+0xf2): undefined reference to `isPrimeNum'
mainapp.c:(.text+0x14b): undefined reference to `isPerfectNum'
mainapp.c:(.text+0x19e): undefined reference to `isPalindrome'
collect2: error: ld returned 1 exit status
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ pwd
/home/prashanth/cprog/myproj
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ gcc mainapp.c -L. -lmynum -o app
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ./app
1 : Even or Odd
2 : Prime Number
3 : Perfect Number
4 : Palindrome Number
0 : Exit
1
Enter a number to check for Even or odd :: 3
Given Number is Odd
1 : Even or Odd
2 : Prime Number
3 : Perfect Number
4 : Palindrome Number
0 : Exit
```

```
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ar -cr libmynum.a prime.o palindrome.o evenodd.o perfect.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.o  libmynum.a  myfunctions.h  palindrome.o  perfect.o  prime.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ rm *.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
libmynum.a  myfunctions.h
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$
```

```
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ rm prime.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.o  myfunctions.h  palindrome.c  palindrome.o  perfect.o  prime.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ rm palindrome.c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.o  myfunctions.h  palindrome.o  perfect.o  prime.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ man ar
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ c
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ar -cr libmynum.a prime.o palindrome.o evenodd.o perfect.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$ ls
evenodd.o  libmynum.a  myfunctions.h  palindrome.o  perfect.o  prime.o
prashanth@DESKTOP-EEREG1R:~/cprog/myproj$
```

Dynamic lib: 10 fns are not replaced by fn definition in executables

They are replaced by the [addresses] not content

Library has to be there during runtime

(.dll)

gcc -c -fPIC prime.c
