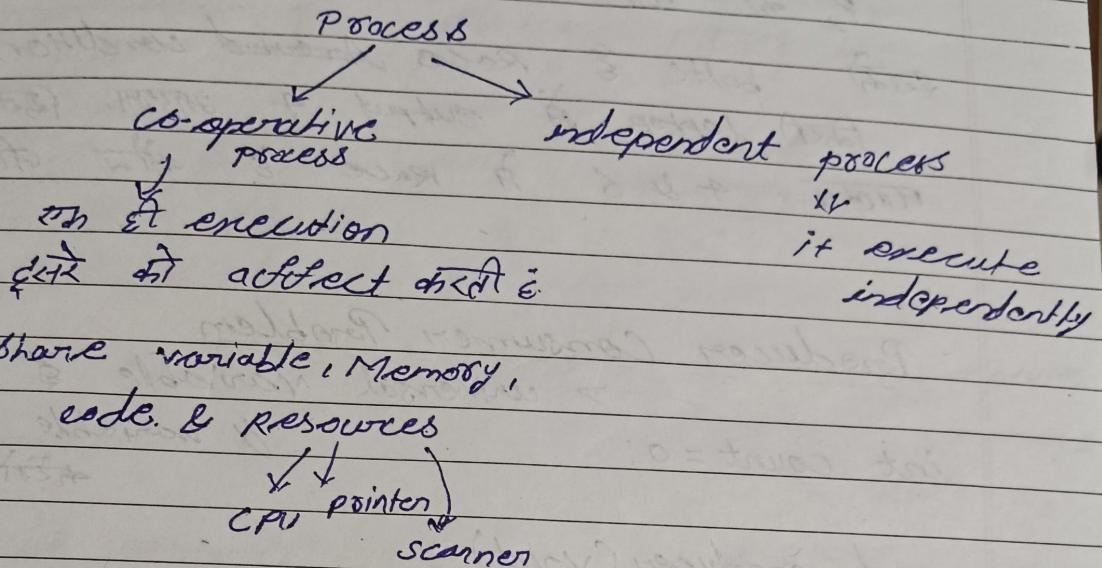


unit 3

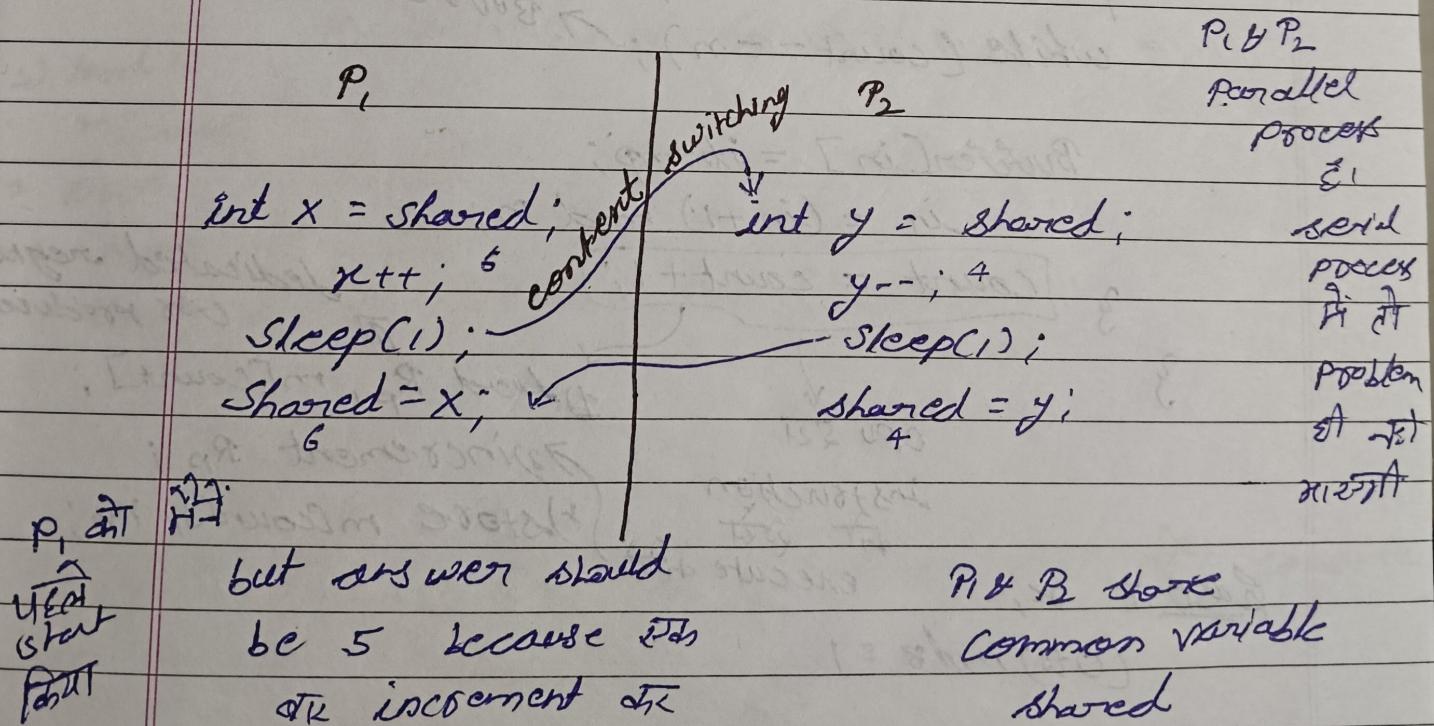
Date _____
Page _____

Process Synchronisation



Co-operative processes \Rightarrow OR \neq \Rightarrow synchronise
 at ~~parallel~~ GTO then they create problem \Rightarrow sed

$$\text{int shared} = 5 \quad 4$$



but answer should
 be 5 because it's
 increment at

$P_1 \& P_2$ share
 common variable
 shared

2) after 1 bar

decreased \therefore answer should be 5

Q 4 states that because process synchronised \Rightarrow

$P_1 \rightarrow$ start 1st at answer 4 3/1/21
 $P_2 \rightarrow$ start 2nd at answer 6 3/1/21

both 8 race Around condition
 both laptop & output 4 3/1/21 both 6,
 4 & 6 in race & after 8 10/1

out

8

out

10

 $P_C = P_2$

Producer Consumer Problem

universal variable & consumer
 int count = 0;

variable share
 8 10/1

void producer(void)

{

int itemp;

while (true)

{

produce_item(itemp);

while (count == n); \rightarrow Buffer full

Buffer[in] = itemp;

in = (in + 1) mod n;

Count = count + 1; $\boxed{}$

3

CPU S21

instruction

at 1st

execute in S21

dedicated register
 for producer

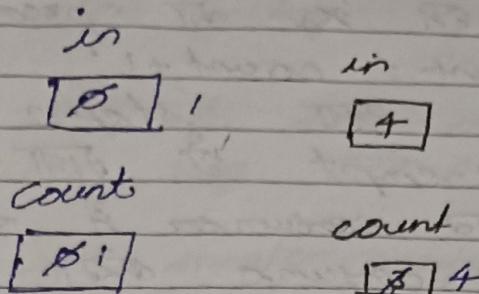
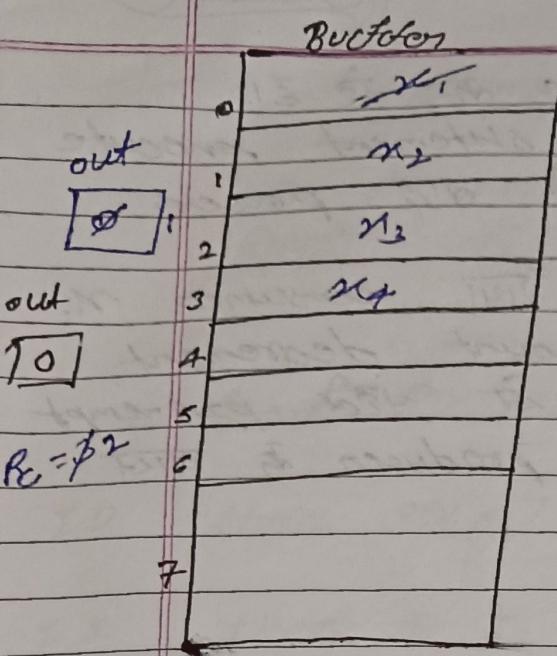
1) load $R_p, m[count]$;

2) increment R_p ;

3) store $m[count], R_p$;

Case I x,

$(0+1) \text{ mod } 8 = 1$



case: produces
product I_1, I_2 consumer I_1 ,
 I_2 producer I_3

consumer I_3 net R_c
count = 4

void consumer(void)

8 int itemC;
while (true)

9 {
10 while (count == 20);

11 itemC = Buffer(out);

12 m[count]

13 m[count] =

14 out = (out + 1) mod n

15 count = count - 1

16 process-item(itemC);

17 18 DEC R_c

19 m[count], R_c

20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1041 1042 1043 1044 1045 1046 1047 1048 1049 1040 1041 1042 1043 1044 1045 1046 1047 1048 1049 1050 1051 1052 1053 1054 1055 1056 1057 1058 1059 1050 1051 1052 1053 1054 1055 1056 1057 1058 1059 1060 1061 1062 1063 1064 1065 1066 1067 1068 1069 1060 1061 1062 1063 1064 1065 1066 1067 1068 1069 1070 1071 1072 1073 1074 1075 1076 1077 1078 1079 1070 1071 1072 1073 1074 1075 1076 1077 1078 1079 1080 1081 1082 1083 1084 1085 1086 1087 1088 1089 1080 1081 1082 1083 1084 1085 1086 1087 1088 1089 1090 1091 1092 1093 1094 1095 1096 1097 1098 1099 1090 1091 1092 1093 1094 1095 1096 1097 1098 1099 1100 1101 1102 1103 1104 1105 1106 1107 1108 1109 1100 1101 1102 1103 1104 1105 1106 1107 1108 1109 1110 1111 1112 1113 1114 1115 1116 1117 1118 1119 1110 1111 1112 1113 1114 1115 1116 1117 1118 1119 1120 1121 1122 1123 1124 1125 1126 1127 1128 1129 1120 1121 1122 1123 1124 1125 1126 1127 1128 1129 1130 1131 1132 1133 1134 1135 1136 113

$$\lim_{\alpha \rightarrow 0} (x+1)^{\cot x}$$

जब एक प्रोड्यूसर ने एक बटर में 1 produce कर दी।

count = count + 1; if statement execute
तो अब step 2 के पश्चात्
pre-empt हो जाता

अब consumer A लेता है consumer X,
जो consume करता है count decrement
करता है लेकिन store करता है और pre-empt
हो जाता है तभी producer को पास
जाता है CPU.

flow of process

producer I₁, I₂ consumer I₁, I₂ producer I₃,
consumer I₃

↓
count = 2
↓

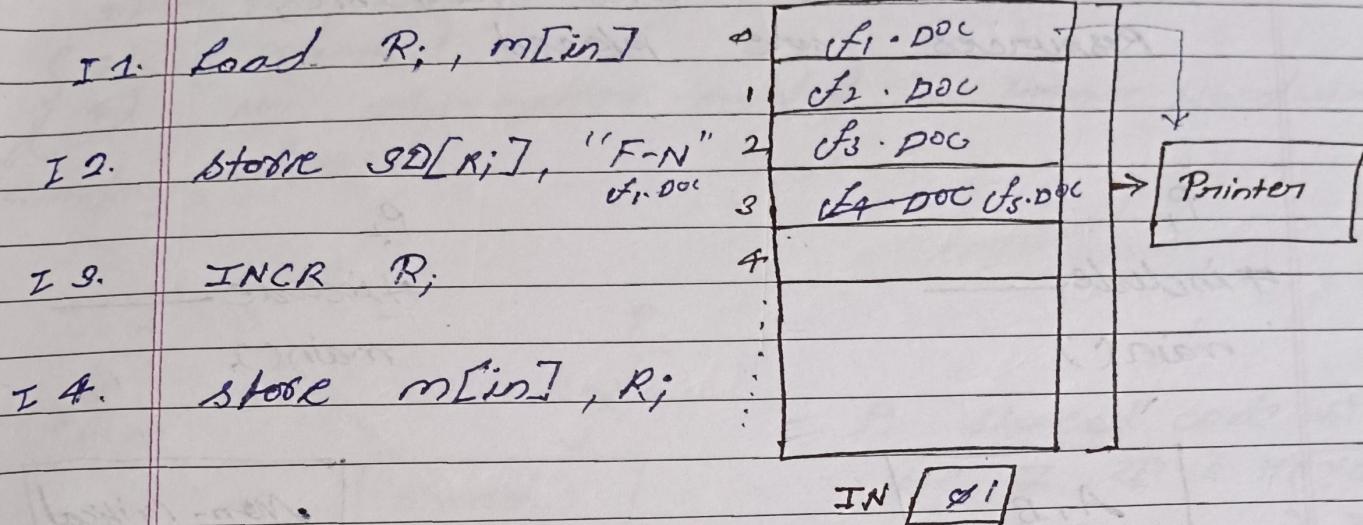
जब लोग इस बटर में 2 value
के लिए लेते हैं लेकिन वह 3 value है।

Here again race-condition हो सकती है
consumer producer की ओर लेता है।

Printer - spooler Problem

जब एक printer होता है और वह नेटवर्क
जिसके अन्तर्गत कई यूजर उपयोग करते हैं।

spooler directory



Case I:

$P_1: f_1 \cdot \text{DOC}$

R_i 1

case II:

P_1 \downarrow P_2 \uparrow
 $f_4 \cdot \text{DOC}$ $f_5 \cdot \text{DOC}$

f_4, f_5 एक spooler
directory में स्टेट
हो रहे हैं f_1, f_2
 f_3 घटे हैं एक spooler
directory में।

मत P_1 कोड ओपन

P_1, I_1, I_2, I_3 | P_2, I_1, I_2, I_3, I_4 | I_4

P_1 3

prompt prompt

281 के लिए डाटा 581

$f_4 \cdot \text{DOC}$ फाइल जल्दी से सेट

Critical section:- "it is part of the program where shared resources are accessed by various processes"

OR

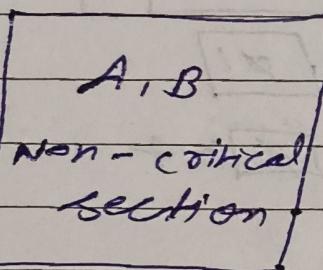
co-operative

place where shared variable, resources are placed

P₁

#include <...>

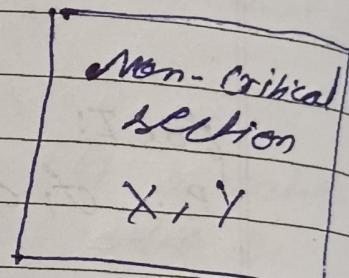
main()



P₂

#include <...>

main()



Entry section

Entry section

Count ++

Critical section

Count --

Critical section

Exit section

Exit section

P₁

Not lock a critical section if it isn't because it is or not share it

Problem At the start both of them exit

the two exit process synchronize

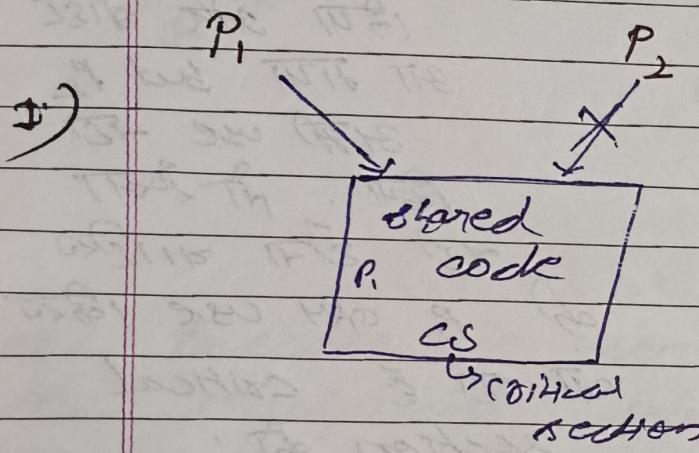
Program and execution state ही होना करते
जब process होते हैं।

Date _____
Page _____

Synchronisation Mechanism

4 conditions

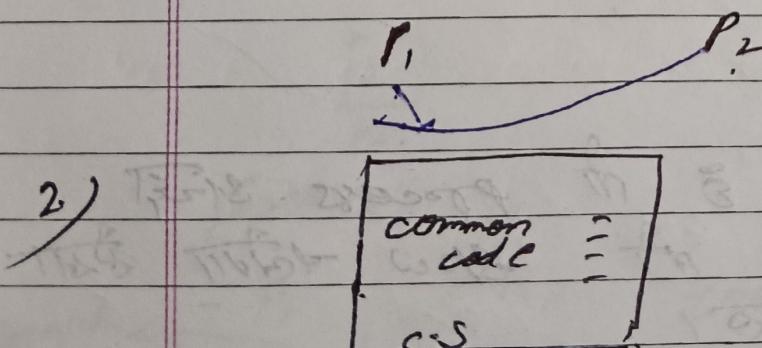
- 1) mutual exclusion & primary
 - 2) Progress
 - 3) Bounded wait.
 - 4) NO assumption related to ~~higher~~ Hardware Speed
- secondary



P₁ shared code की
use के लिए हमें प्री
critical section ही कै
से P₂ critical
section की वाली
एक्शन
सुन सकते हैं।

Mutual Exclusion:-

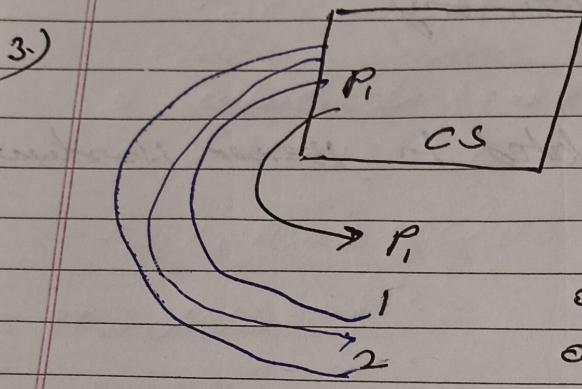
जब process critical
section की ओर आती है
तब उसे नहीं देता है।



P₁ जब भाग की है
तब P₂ की
entry section की ओर
प्रैक्टिक कोड लिया जाता है और
P₁ block हो जाता है।

जो ऐसा होता है कि
progress की ओर चाहिए

Progress:- Each process ~~is~~ at progress
at stop net result.



P_1 es की
 use किया गया
 आया but P_2 use
 करी किया

P_1 के बारे में
 किया जैसे गया
 आया but P_2
 के बारे में किया

किया नहीं किया

Bounded a.s.t:-

at let it critical
section at.

Two processes critical section की उपयोग की विधि करते हैं
 but प्रत्येक अपनी प्रोसेस की
 Q/R की critical section की उपयोग की विधि करते हैं

$P_1 - \infty$ } X
 $P_2 - 1$

4) processor IGNZ की प्रैस कैम्प विनियोग
वर्तमान में एक विनियोग संस्था है।

Lock variable in OS

Critical section solution using "lock"

do {

 acquire lock.

 CS (critical section)

 release lock

lock format
1 2 3
4 5 6

1. while (LOCK == 1);

2. LOCK = 1;

ENTRY
CODE

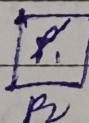
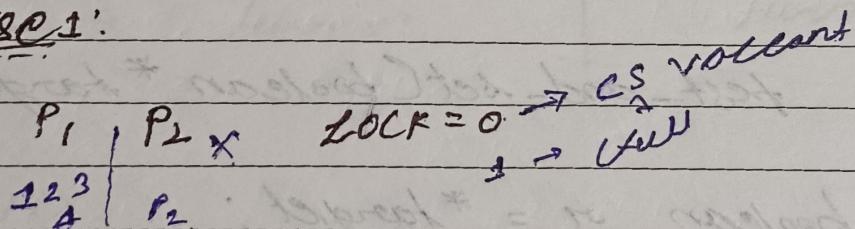
3. critical section

4. LOCK = 0

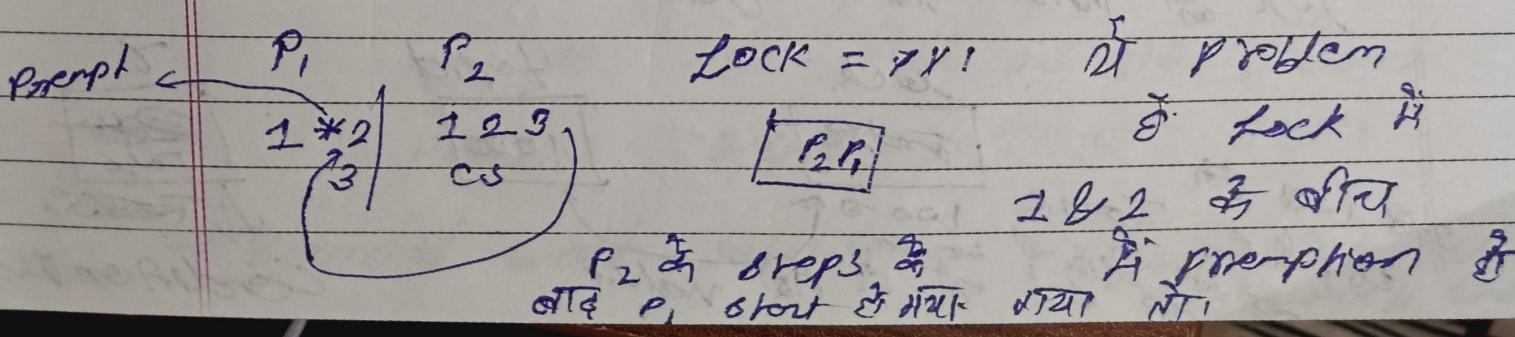
EXIT
CODE

- Execute in user mode
- Multiprocess solution
- NO mutual exclusion guarantee

Case 1:



Case 2:



Lock method & get problem of deadlock
resolve with the help of use of test & set instruction

Test & set Instruction

1. `while (LOCK == 1);`] \Rightarrow ~~critical section~~
 2. `LOCK = 1` \Rightarrow combined
 3. critical section \Rightarrow ~~critical section~~
 4. `LOCK = 0`

```
while( test-and-set (&lock) );
```

CS

~~lock = false~~

प्रेयर्स poesia
प्रेरणा प्रेरणा
Chance of
प्रेरणा.

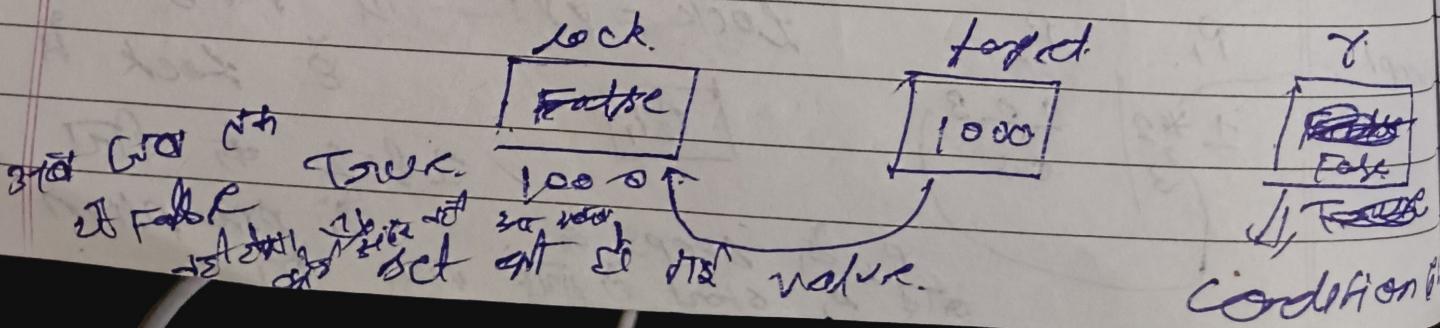
boolean test-and-set(boolean *target)

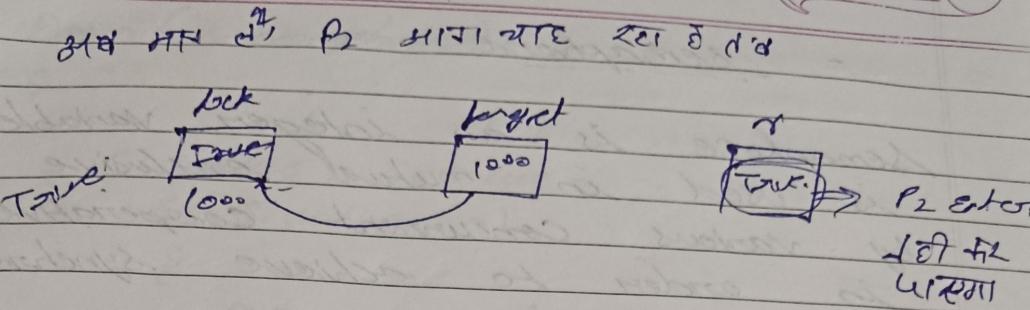
`bool can or = *target;`

* project = Tome;

return to:

lock is initial value false if





Twin Variable (strict Alteration)

- * 2 process solution
- * Run in user mode

int turn = 0

Process P ₀	Process "P,"
Entry code \rightarrow while (turn != 0); [critical section]	while (turn != 1); [critical section]
Exit code \rightarrow turn = 1	turn = 0;

[P₀] P, X
CS

\rightarrow mutual exclusion always satisfies

\rightarrow progress not satisfied

turn variable value not not

change not start not not second
process 3rd and off system

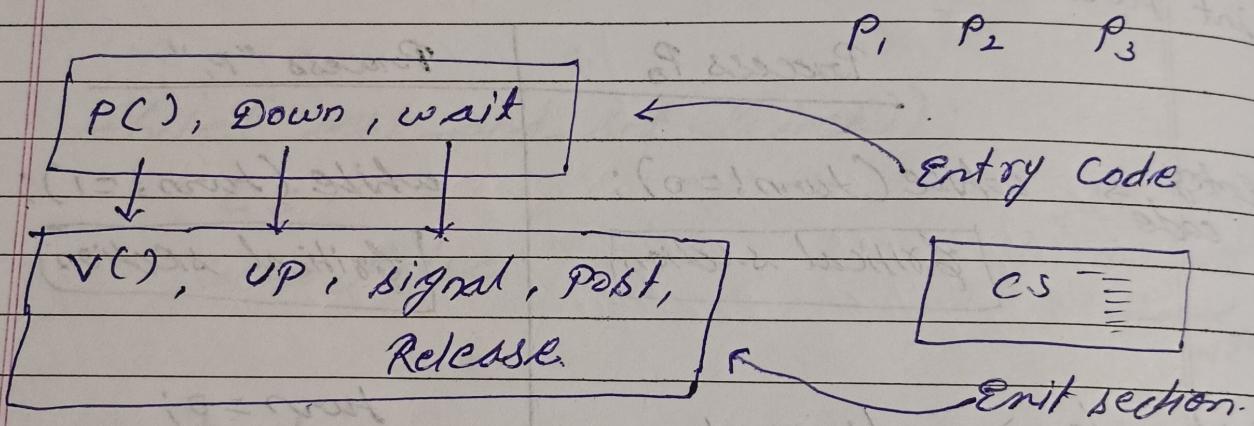
\rightarrow bounded wait satisfied. \rightarrow Independent of
software & hardware

Semaphore

Semaphore is an integer variable which is used in mutual exclusive manner by various concurrent cooperative processes in order to achieve synchronisation.

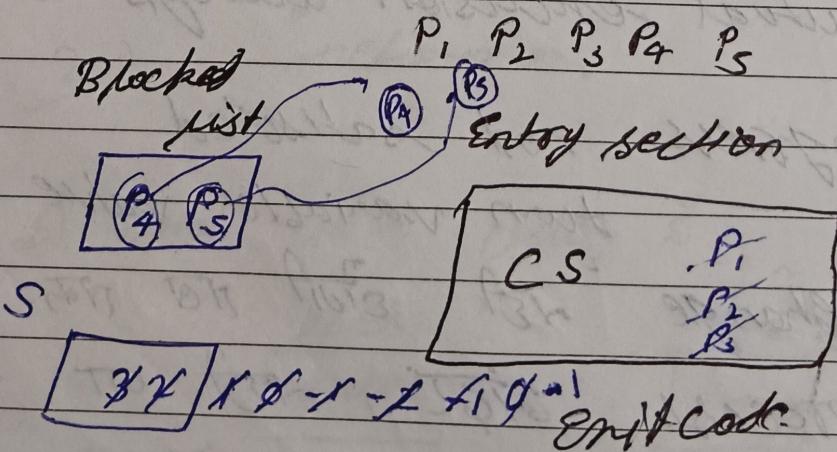
Semaphore

Counting Binary
 $(-\infty \text{ to } +\infty)$ $(0, 1)$



Entry code if $P(S)$ latest & it's corresponding $V(S)$ locked similarly $\text{Down} \rightarrow \text{UP}$

$S = -2$
 & if need
 2 processes
 blocked state
 $\Rightarrow 1$



Down (semaphores)

{

$s_value = s_value - 1;$
if ($s_value \leq 0$)

{

put process (PCB) in
suspended list, sleep();

}

else {

return;

}

UP (semaphore s)

{

$s_value = s_value + 1;$

if ($s_value \leq 0$)

{

select a process from
suspended list wake up(); } processes

}

}

If $s = 10$, now

many process can enter
in critical section

successfully.

Ex

10 process

Successful operation on HLLB in C.S
successfully 21/07/11

Q $s = 10$ After 4 V operation what will be the value of s

10 9 8 7 6 5 4 3 2 1

Ans $s = 8$

Q $s = 17$ 5 P operation then 3 V operation then 1 P operation
 $-5 + 3 - 1 = -3$

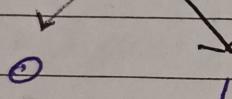
$$17 - 3 = 14$$

$\boxed{s = 14}$ Answer

P & V decrements
 at $\Sigma 1$

V & increment of
 $\Sigma 1$

Binary Semaphore



For Down
 $s = 0$ \rightarrow Net blocked
 $s = 1$ \rightarrow Net unblocked

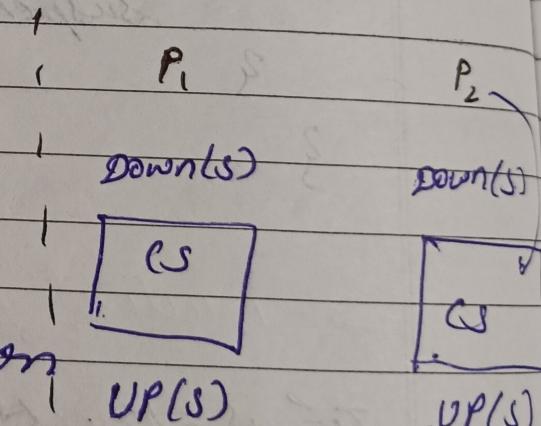
For Down

$s = 10$

successful operation

$s = \emptyset 0$ block

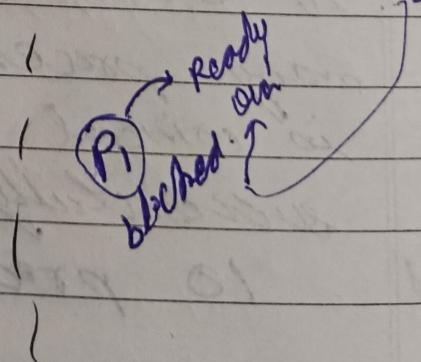
unsuccessful operation



For up

$s = \emptyset 1$

$s = \emptyset 0$



Down (semaphore s)

{

if (s.value == 1)

{

s.value = 0;

}

else

{

Block this process and place in
suspend list, sleep().

}

}

UP (semaphore s)

{

if (suspend list is Empty)

{

s.value = 1;

}

else

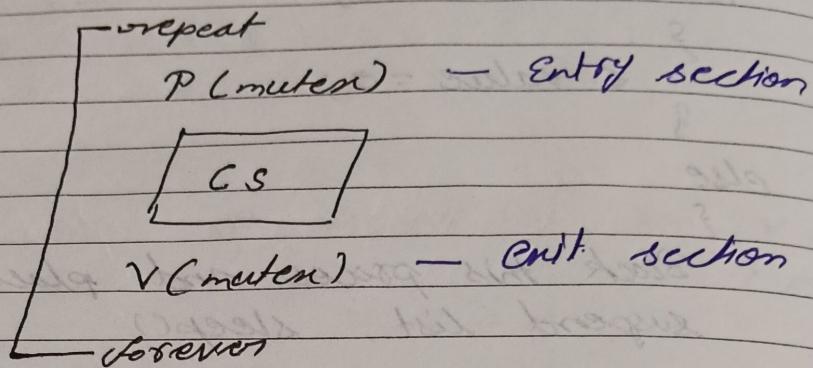
{

Select a process from suspend list
and wake up().

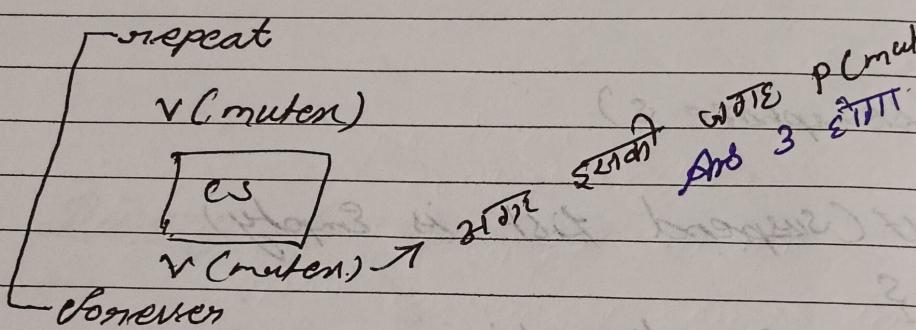
- limit $(x+1)$

Date _____
Page _____

2 Each process $P_i \{ i=1 \text{ to } 9 \}$ execute the following code

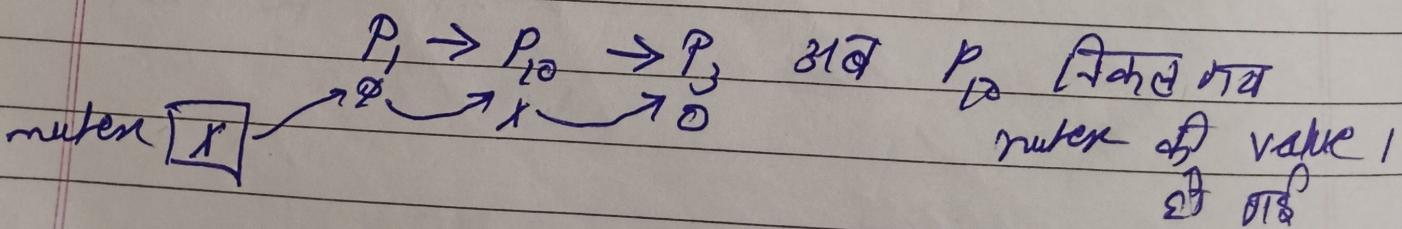


process P_0 execute the following code



what is minimum number of process that may present in critical section at any point of time?

Ans V का नहीं होता कि CS के बाहर होने के लिए वही स्थिति होती है कि P_1 to P_9 नहीं होते हैं CS के लिए वही स्थिति होती है कि P वाली वैल्यु वाली होती है।



P₁ अंदर गया फिर P₁₀ फिर P₂ अंदर गया, फिर P₁₀ बाहर
गया फिर P₃ अंदर गया, फिर P₁₀ अंदर गया
फिर P₄ अंदर गया फिर P₁₀ बाहर गया ---
so on
CS

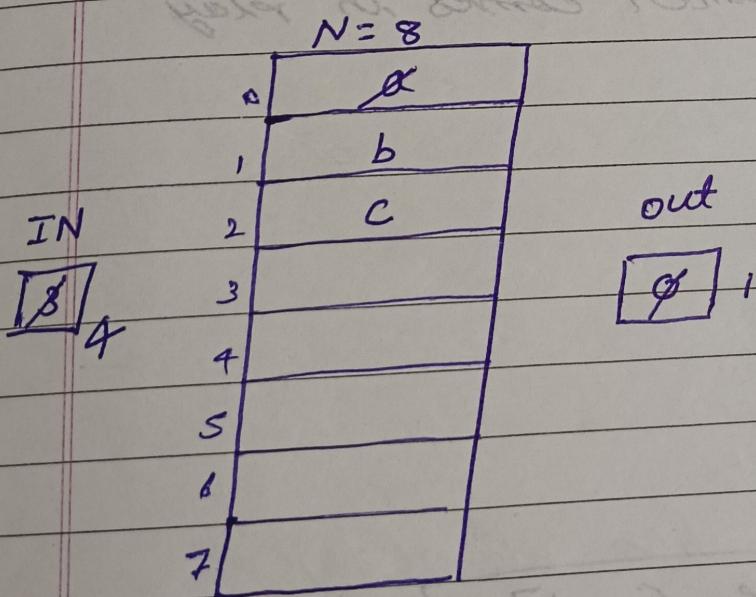
$P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}$

Ans [10]

Solution of Producer-Consumer Problem

Counting Semaphore \rightarrow Full = 0 = No. of filled slots
 \rightarrow Empty = N \rightarrow No. of empty slots

Let at any point of time.



Empty = 5 & 4

Full = 3 & 3

S = 1

Produce-item (item b)

- 3) down (Empty);
- 2) down (s);

$\text{Buffer[IN]} = \text{itemp};$
 $\text{IN} = (\text{IN}+1) \bmod n;$

- 3) up (s)
- 4) up (full);

now, producing d

$$s = 1010$$

$$\text{IN} = (3+1) \bmod 8 = 4$$

producer

critical section

at dist 3 ft away

now, consumer comes in play

consumer

$$\begin{array}{r} \diagup \\ \diagdown \end{array} = \text{full}$$

$$\begin{array}{r} \diagup \\ \diagdown \end{array} = 1 = 2$$

- 1) down (full)
- 2) down (s)

$\text{itemC} = \text{Buffer[out]};$
 $\text{out} = (\text{out}+1) \bmod n$

- 3) up (s)
- 4) up (empty)

previous example if it sequentially has
no pre-emption or no debt
now assume

$$N = 8$$

	a
1	b
2	c
In	
3	
4	out
5	
6	
7	

1.0 | 1

empty = \$ 4^s producer prompt
full = \$ 2.3 consumer come

$$S = 1 \otimes V \otimes 1$$

old P.O.

consumer

other not

new C.S.

Not old producer
C.S. is bigger next

AT 4/27/11

problem solved.

Reader-writer Problem

Reader, writer i database mt use or kese

J^R, J^W, J^R, J^W

Database

so Database ab size kse read or kare
R, w, time w, bit marni h
problem create h gikarast

but R-W \rightarrow problem

W-R \rightarrow problem

W-W \rightarrow problem

R-R \rightarrow No problem

इनके remove करने के
लिए C++ code लियांगे।

int nc = 0;

Binary semaphore mutex = 1;

Binary semaphore db = 1;

void Reader(void)

{

 while(true)

{

 down(mutex);

 nc = nc + 1;

 if(nc == 1) then down(db);

 up(mutex);

DB

 down(mutex)

$rc = rc - 1;$
if ($rc == 0$) then $up(db);$

$up(mutex)$

Process-data

{
}

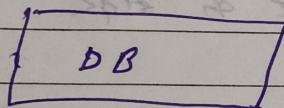
void writer(void)

{

while (true)

{

$down(db);$



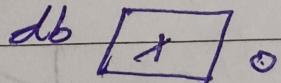
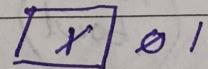
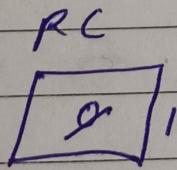
$up(db);$

{
}

Case I

Reader R₁ comes first

mutex



Case II

w-R problem

$db = 10$

mutex = 10

$rc = 0$

reader blocked

RC & writer waiting

$db = 0$ at block

at wait 1

$\lim_{x \rightarrow 0} (x+1)^{\cot x}$

Date _____
Page _____

case II

$w - w$

$db = 1^o$ → blocked writer w_2 after $31/10/2021$

case IV

R-R

marked = $1 \oplus 1$

$rc = 012$

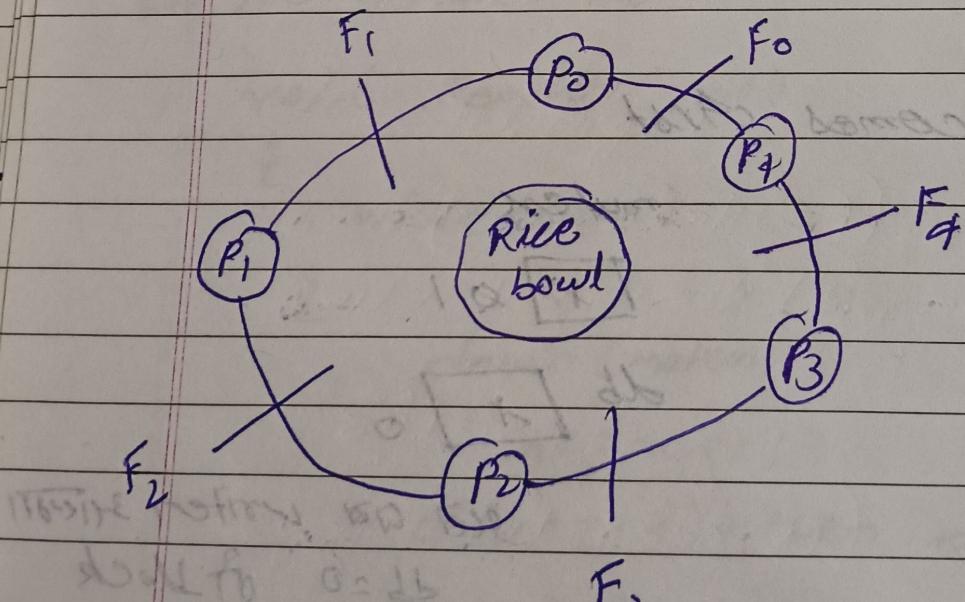
$db = 1^o$

R_1 $31/10/2021$

R_2

$4A$ $31/10/2021$ Database is size

Dining Philosophers Problem



5 philosophers

5 Forks

void philosopher(void)

white (true)

Thinking();
 take_fork(i); \leftarrow left fork
 take_fork($(i+1) \cdot N$); \rightarrow right fork
 EAT(); \nearrow No. of Forks
 put_fork(i);
 Put_fork($(i+1) \cdot N$);

3

3

case I

P_0

लोक भिट्ठे को रख देंगा भिट्ठे को

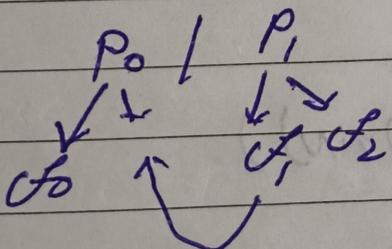
1

$f_1 f_2$

philosophers philosopher state serial manner ⁹

It's not code perfectly written.

case 2.



P_0 , P_1 राया P_1 का P_1 का
राया और P_1 का राया लिया
अब P_0 wait कर रहा है ए
को लिया

\$ 1st & 2nd philosopher संतोष ने कहा है कि समय
time के लिए वाले condition arise करता है।

Solutions of problem

We use 5 semaphore

$s[i] \rightarrow$ live semaphore

s_0, s_1, s_2, s_3, s_4
 ↓ ↓ ↓ ↓ ↓
 ; ; ; ; ;

(P₀) → execute task 2
 2 semaphore diff.
 s_0, s_1

P₁: s_1, s_2

P₂: s_2, s_3

P₃: s_3, s_4

P₄: s_4, s_0

void philosopher(void)

{

white(chow)

{

Thinking();

wait(take-fork(s_i));

wait(take-fork(s_{i+1}), red N))

EAT();

Signal(put_fork(i))

Signal(put_fork((i+1) % N))

}

3rd GT problem

ET 2nd ET

4th ET 3rd ET

ET

P_i block

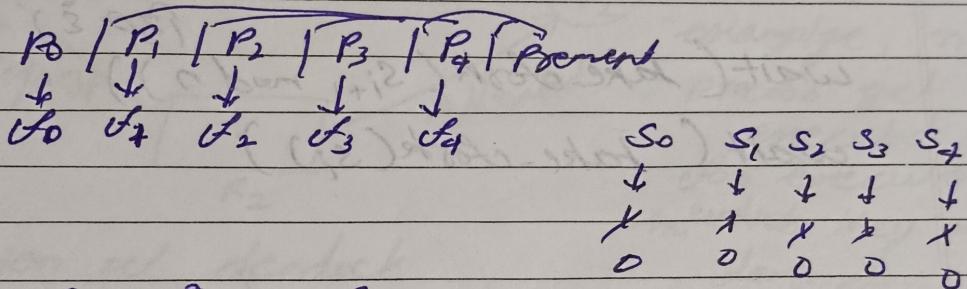
ET after

प्राणी P₁ P₂ आज्ञा आ लकड़ी है same
 हम पर महले दूत ने लकड़ी because
 तो वीच कुछ common नहीं है।

अनेक दर्शकों के अनुसार philosopher ही एकत्र ही लेते हैं लेकिन उनका विचार अपेक्षा नियन्त्रित नहीं होता।

deadlock condition will arise if CPU

Po आजका leaf वाला clock 351211 372
Prompt द्वारा अब P, 371211 को क्या करें



25 बड़ी की एक night
एल फूक मिले ही भी block हो

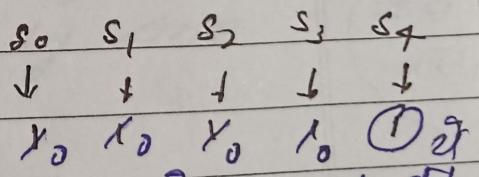
GREGORY

so, deadlock condition arise.

selection for this

P_q of sequence $\{b_n\}$ of \mathbb{R}^n

P_4 : $S_0 \quad S_4$



भिन्न वाले philosophers द्वारा सब उद्देश्यों की विभिन्नता

गलत look 30 सेकंडों फिर right तरीका look

but last वाला पहले night वाला दूसरा

कृष्ण लोह विक्री

P_0	s_0	s_1
P_1	s_1	s_2
P_2	s_2	s_3
P_3	s_3	s_4
P_4	s_0	s_4

Blocked.

mandatory at $\frac{1}{2}$ $\frac{1}{2}$
~~for~~ for lost
 philosopher at $\frac{1}{2}$ $\frac{1}{2}$
 sequence change
 at $\frac{1}{2}$ $\frac{1}{2}$
 $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$
 $\frac{1}{2}$ $\frac{1}{2}$

For N^{th} philosopher (मिसका sequence change
 $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$)

wait(take-clock($s_{i+1} \bmod n$)))

wait(take-clock(s_i))

process resource hold

किए हैं तो $\frac{1}{2}$ $\frac{1}{2}$ resource
 hold किए हैं तो $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$
 resource at demand at