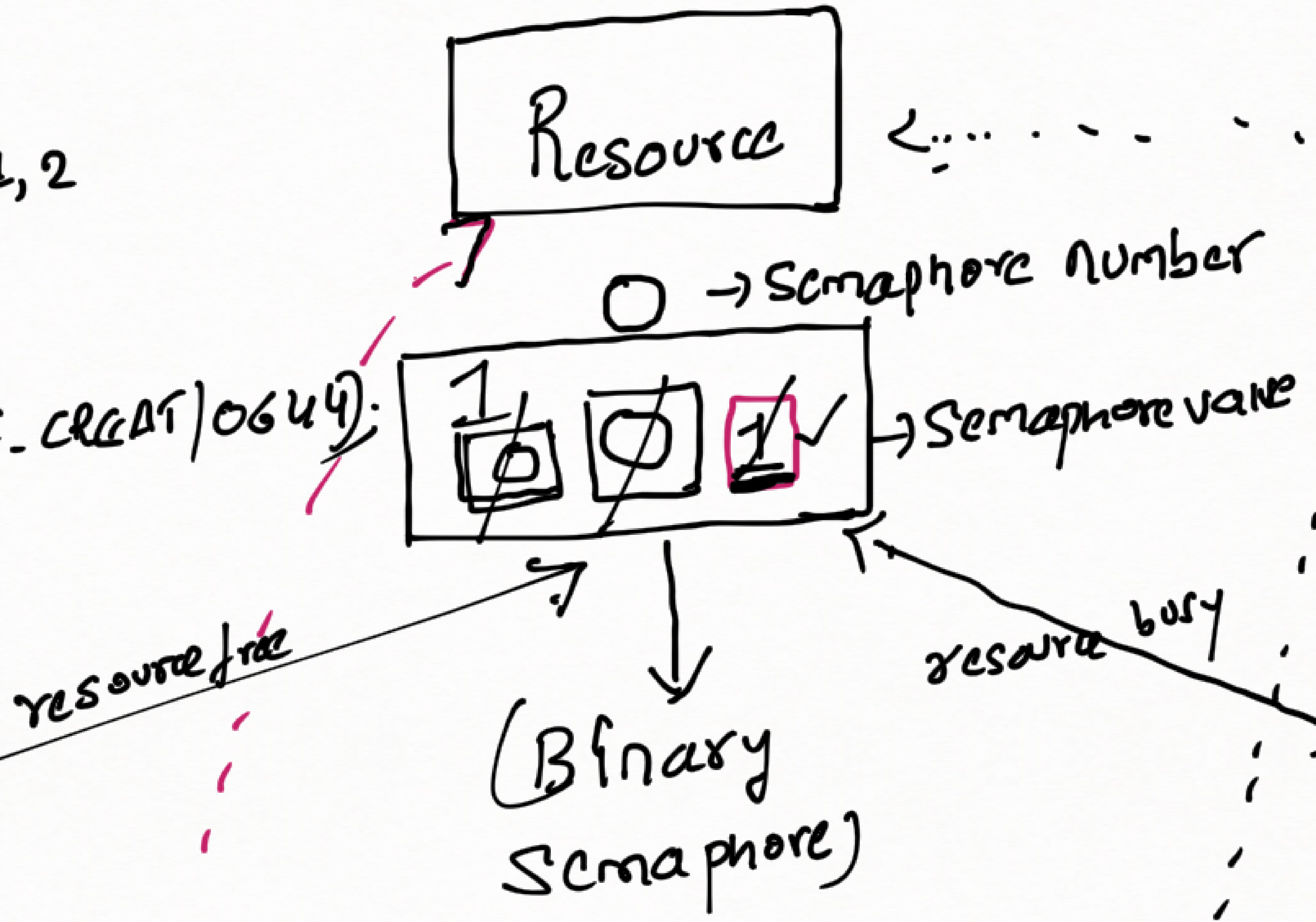
Date: 24-11-21 ✓

0, 1, 2

P1

```
id = semget(5, 3, IPC_CREAT|0644);
struct sembuf v;
v.sem_num = 0;
v.sem_op = 0;   ← wait for zero)
v.sem_flg = 0;
printf("Before semop\n");
✓ semop(id, &v, 1);
✓ semctl(id, 0, SETVAL, 1);
≡ } → critical section
```

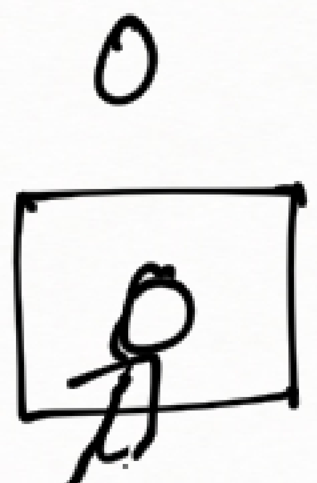


```
semctl(id, 0, SETVAL, 0);
```

P2

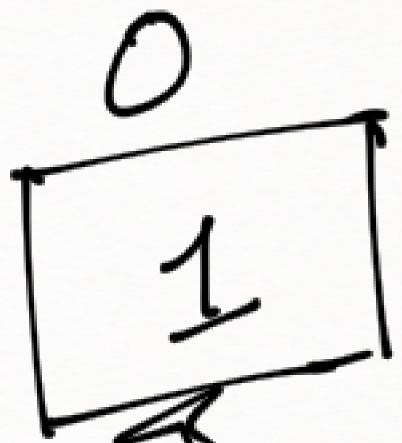
```
id = semget(5, 3, IPC_CREAT|0644);
struct sembuf v;
v.sem_num = 0;
v.sem_op = 0;   ← (wait for zero)
v.sem_flg = 0;
printf("Before semop..\n");
semop(id, &v, 1);
semctl(id, 0, SETVAL, 1);
≡ } critical section code
semctl(id, 0, SETVAL, 0)
```

Resource free



v. sem-op = 0;
&
Semaphore value zero

Resource busy



V. Sem. op = 0
&
Semaphore value 1

→ If any process want to access a resource

(i) process will Test resource if free (or) busy

(ii) If free will access / If busy will wait

(iii) after resource access done release the source

→ In the Semaphore value is either Zero (or) one then that type of semaphores are caled as binary semaphores.