```
     sleep(2); // give readers chance
   }
return NULL;
}

int main() {
    // pthread_t r[5], w[2];
    int num_readers = 5;
    int num_writers = 2;

    pthread_t *r = malloc(num_readers *
sizeof(pthread_t));
    pthread_t *w = malloc(num_writers *
sizeof(pthread_t));

    int r_ids[5] = {1, 2, 3, 4, 5};
    int w_ids[2] = {1, 2};

    // initialize semaphores
    sem_init(&x, 0, 1);
    sem_init(&wsem, 0, 1);

    // create threads
    for (int i = 0; i < 5; i++)
        pthread_create(&r[i], NULL, reader, &r_ids[i]);

    for (int i = 0; i < 2; i++)
        pthread_create(&w[i], NULL, writer, &w_ids[i]);

    // join threads (they loop infinitely)
    for (int i = 0; i < 5; i++)
        pthread_join(r[i], NULL);

    for (int i = 0; i < 2; i++)
        pthread_join(w[i], NULL);

    // cleanup (not normally reached)
    sem_destroy(&x);
    sem_destroy(&wsem);

    return 0;
}
```

CODE:

```
manasvi@manasvi:/mnt/c/Users/bhute/Desktop/oslab/Ass 4$ ./four
Reader 1 is reading shared_data = 0
Reader 5 is reading shared_data = 0
Reader 3 is reading shared_data = 0
Reader 4 is reading shared_data = 0
Reader 2 is reading shared_data = 0
```

```
Writer 1 is writing shared_data = 1
Writer 2 is writing shared_data = 2
Reader 1 is reading shared_data = 2
Reader 4 is reading shared_data = 2
Reader 5 is reading shared_data = 2
Reader 3 is reading shared_data = 2
Reader 2 is reading shared_data = 2
Writer 1 is writing shared_data = 3
```