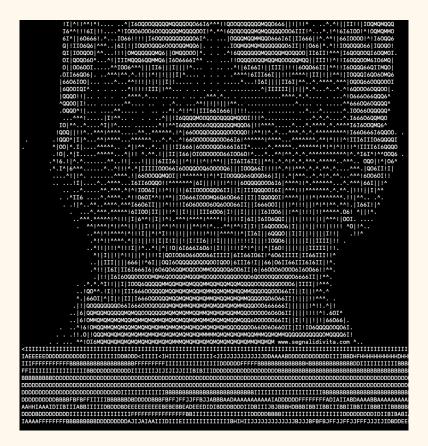
# Operating Systems Project 3 The Reader Writer Problem

By HONGGEUN JI [2019042633], Division of Computer Science, HYU ERICA 14.05.21



# Introduction

This report is about the third assignment given by Dr. HEEKUCK OH. The assignment was to understand the problems of synchronization and provide appropriate solutions. The reader preference problem is already dealt with in the lab. This report will only focus on the writer-preference and fair-reader-writer problem.

By using the POSIX synchronization, we can solve this preference problem and can fully understand the synchronization concept.

## How does it work? - The Writer Preference

#### 1. The Mutex "r avail"

This " $r_avail$ " mutex will be used to compete between readers and writers. As writers should precede the readers even if the writers arrive later, the first writer lock this mutex (get this mutex) to precede the other readers. The readers should get the " $r_avil$ " mutex later and this leads the readers to wait until the last writer unlocks this mutex.

## 2. The Mutex "r\_mutex" and "w\_mutex"

Both readers and writers should count themselves to choose when to lock or unlock the "rw\_mutex". However, counting should happen one at a time, and this means no overlap among the readers or writers. To do this, "r\_mutex" and "w mutex" will be used when they have to count up or down their number.

## 3. The Mutex "rw\_mutex"

The shared mutex when readers or writers run themselves. As the runner can overlap with other runners, only the first runner locks this mutex, and let the last mutex unlocks it. However, writers cannot overlap not only with the readers but also other writers. In this regard, the "rw\_mutex" will be used all the time whenever the writer wants to run itself.

# **Codes - The Writer Preference**

```
* Copyright 2020, 2021. Heekuck Oh, all rights reserved
*이 프로그램은 한양대학교 ERICA 소프트웨어학부 재학생을 위한 교육용으로 제작되었습니다.
*/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <pthread.h>
#include <unistd.h>
#include <fcntl.h>
#define N 8192
#define L1 75
#define L2 70
#define L3 70
#define RNUM 10
#define WNUM 3
face picture with a number of chars...
*/
* alive 값이 1이면 각 스레드는 무한 루프를 돌며 반복해서 일을 하고,
* alive 값이 0이 되면 무한 루프를 빠져나와 스레드를 자연스럽게 종료한다.
int alive = 1;
pthread_mutex_t rw_mutex;
pthread_mutex_t r_mutex;
pthread mutex tr avail;
pthread_mutex_t w_mutex;
int r cnt = 0;
int w_cnt = 0;
* Reader 스레드는 같은 문자를 N번 출력한다. 예를 들면 <AAA...AA> 이런 식이다.
*출력할 문자는 인자를 통해 0이면 A, 1이면 B,..., 등으로 출력하며, 시작과 끝을 <...>로 나타낸다.
* 단일 reader라면 <AAA...AA>처럼 같은 문자만 출력하겠지만, critical section에서 reader의
* 중복을 허용하기 때문에 reader가 많아지면 출력이 어지럽게 섞여서 나오는 것이 정상이다.
void *reader(void *arg)
```

```
int id, i;
    *들어온 인자를 통해 출력할 문자의 종류를 정한다.
   id = *(int *)arg;
   * 스레드가 살아 있는 동안 같은 문자열 시퀀스 <XXX...XX>를 반복해서 출력한다.
   while (alive) {
       pthread_mutex_lock(&r_avail); // reader available when writer allows
           pthread_mutex_lock(&r_mutex); // r_cnt cannot be accessed at the same
       if(++r_cnt == 1) pthread_mutex_lock(&rw_mutex); // first reader get rw_mutex for others
           pthread mutex unlock(&r mutex);
       pthread_mutex_unlock(&r_avail);
       * Begin Critical Section
       printf("<");</pre>
       for (i = 0; i < N; ++i)
          printf("%c", 'A'+id);
       printf(">");
       * End Critical Section
       pthread_mutex_lock(&r_mutex);
       if(--r_cnt == 0)
           pthread_mutex_unlock(&rw_mutex);
       pthread mutex unlock(&r mutex);
   pthread_exit(0);
* Writer 스레드는 어떤 사람의 얼굴 이미지를 출력한다.
*이미지는 세 가지 종류가 있으며 인자를 통해 식별한다.
* Writer가 critical section에 있으면 다른 writer는 물론이고 어떠한 reader도 들어올 수 없다.
*만일 이것을 어기고 다른 writer나 reader가 들어왔다면 얼굴 이미지가 깨져서 쉽게 감지된다.
void *writer(void *arg)
   int id, i;
   struct timespec req, rem;
   *들어온 인자를 통해 얼굴 이미지의 종류를 정한다.
```

}

{

```
id = *(int *)arg;
* 이미지 출력을 천천히 하기 위해 한 줄 출력할 때마다 쉬는 시간을 1 나노초로 설정한다.
req.tv_sec = 0;
req.tv_nsec = 1L;
* 스레드가 살아 있는 동안 같은 이미지를 반복해서 출력한다.
while (alive) {
    pthread_mutex_lock(&w_mutex); // w_cnt cannot be accessed at the same
    if(++w_cnt == 1) pthread_mutex_lock(&r_avail); // first writer suspends readers
    pthread_mutex_unlock(&w_mutex);
    pthread_mutex_lock(&rw_mutex); // writers cannot overlap each other
    * Begin Critical Section
    printf("\n");
    switch (id) {
        case 0:
            for (i = 0; i < L1; ++i) {
                printf("%s\n", t[i]);
                nanosleep(&req, &rem);
            break;
        case 1:
            for (i = 0; i < L2; ++i) {
                printf("%s\n", d[i]);
                nanosleep(&req, &rem);
            }
            break;
        case 2:
            for (i = 0; i < L3; ++i) {
                printf("%s\n", e[i]);
                nanosleep(&req, &rem);
            break;
        default:
    }
    * End Critical Section
    */
    pthread_mutex_unlock(&rw_mutex);
```

```
pthread mutex lock(&w mutex);
       if(--w cnt == 0) pthread mutex unlock(&r avail); // last writer allows the readers come in CS
       pthread_mutex_unlock(&w_mutex);
     }
    pthread_exit(0);
}
*메인 함수는 RNUM 개의 reader 스레드를 생성하고, WNUM 개의 writer 스레드를 생성한다.
* 생성된 스레드가 일을 할 동안 0.2초 동안 기다렸다가 alive의 값을 0으로 바꿔서 모든 스레드가
*무한 루프를 빠져나올 수 있게 만든 후, 스레드가 자연스럽게 종료할 때까지 기다리고 메인을 종료한다.
int main(void)
    int i;
   int rarg[RNUM], warg[WNUM];
   pthread_t rthid[RNUM];
   pthread_t wthid[WNUM];
    struct timespec req, rem;
   // stdout is now the txt file
   int output fd = open("writer prefer.txt", O CREAT|O TRUNC|O RDWR, 0666);
   dup2(output_fd, STDOUT_FILENO);
    pthread_mutex_init(&rw_mutex, NULL);
    pthread mutex init(&r mutex, NULL);
    pthread_mutex_init(&r_avail, NULL);
    pthread_mutex_init(&w_mutex, NULL);
    * Create RNUM reader threads
    for (i = 0; i < RNUM; ++i) {
       rarg[i] = i;
         if (pthread_create(rthid+i, NULL, reader, rarg+i) != 0) {
             fprintf(stderr, "pthread_create error\n");
             exit(-1);
   }
    * Create WNUM writer threads
    */
    for (i = 0; i < WNUM; ++i) {
       warg[i] = i;
       if (pthread_create(wthid+i, NULL, writer, warg+i) != 0) {
           fprintf(stderr, "pthread_create error\n");
           exit(-1);
       }
```

```
}
    * Wait for 0.2 second while the threads are working
    req.tv_sec = 0;
    req.tv_nsec = 200000000L;
    nanosleep(&req, &rem);
    * Now terminate all threads and leave
    alive = 0;
    for (i = 0; i < RNUM; ++i)
        pthread_join(rthid[i], NULL);
    for (i = 0; i < WNUM; ++i)
        pthread_join(wthid[i], NULL);
    pthread_mutex_destroy(&rw_mutex);
    pthread_mutex_destroy(&r_mutex);
    pthread_mutex_destroy(&r_avail);
    pthread_mutex_destroy(&w_mutex);
    exit(0);
}
```

# Compile and Run

Compile "writer\_prefer.c" with GCC. When you run the "writer\_prefer", you should see there is an output txt file named "writer prefer.txt".

```
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Sys
  main) 🔹 ls
fair_reader_writer.c writer_prefer.c
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Sys
  main) # gcc -o writer_prefer writer_prefer.c
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Sys
  main) ./writer_prefer
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Sys
 (main) # lslaf
total 2568
drwxr-xr-x@ 6 jihonggeun staff
                                    192 May 14 14:46 ./
drwxr-xr-x@ 9 jihonggeun staff
                                    288 May 14 14:46 ../
-rw-r--r-- 1 jihonggeun staff
                                 103535 May 14 09:47 fair_reader_writer.c
-rwxr-xr-x 1 jihonggeun staff
                                  67000 May 14 14:48 w
-rw-r--r- 1 jihonggeun staff 103742 May 14 09:46 writer_prefer.c
-rw-r--r- 1 jihonggeun staff 1029049 May 14 14:48 writer_prefer.txt
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Sys
 (main) ≰ 📗
```

#### The part of writer\_prefer.txt file -- (1)

```
^0|!^..
...
Writers never overlap
.6IIOQQMMMM6....
```

#### The part of writer prefer.txt file -- (2)

```
!^!6IO!I||66QI0066Q|!|III66I66III||||||||||16Q0666III|.^.
I.||^6|!^|!|!I^|!|!.^^!
                             [6][]^[][^!!!!!|0^!!^^.I
                            .^66||||!|!^!^^!||||1
                               66!.^^^
                                ^!!III!I||66|I^
                                ^06|I6|!!!|||I
MMMMMMMMOOOQQQGOGAMMMMQM||OQQQQMOOQQGIMQMMOOQQMMI6I6O|0660|6!6II6I!!6^.^..| |!|^!^!
MMMMQQQQOQQQQGOGAMQQMM|^!QQQQMMMMMQMMMMMMMMMMMMMOQGM60MO6QOIQM6Q|6QI|0^!6QM06!II|^I!
                                .QMQ06II||166II
                                ^QMMMMOI|I66Q06
\mathsf{MQOQQQQQQQQQQQQQQMM} | \dots | \mathsf{QMQMMMMQMQQQQMMMMQMQMQMMMMQMQMQMQMQMGQOO66} | \mathsf{16MMMO} | \mathsf{16} | \mathsf{n16} |
                                MMMMMMQQMMMMM
QQQQMQQQQMMMMOQOOQQMM^!!^...!!MMMMMQQOOQOQOOQMMMMMMMMMMMMMMMMMMOQ6I6II!!|MQOOQ6||I!|!
                                OMMQ666QQOQMQQI
QQ6III6000Q0006
                                QMO60660I066060QQ
                               IQMQ00066000Q606MM
                               0QQ0600600Q00660QMM
                          MMMMMQQQQM
                          MMMQMMMM6.0
                              000660QQ00QQ0066QMMM
IMMMMMMMMM QMO6IOQOQOOQQQO6IQQMMM
                          6QMMMMMMMMO^6IIIOOOOMQMQQOI6QMM
            MMQQMMMMMMMMMMMMMMMMMOI|666IMMMQ^.
                          ..|MMMMMMMMMQMQOO6OOQQMQQMOMOMQMMM
```

#### (writer precedes the Readers)

If you want to see the full text, please take a look inside the "writer prefer.txt"

## How does it work? - The Fair Readers Writers

#### 1. The Mutex "fcfs"

Every Reader and Writers should get this mutex that works similarly to the FCFS queue. After a reader or writer get this lock, the process of getting "r\_mutex" or "rw\_mutex" is quite similar to the Reader preference. As we agree to allow readers can overlap each other, only the first and last reader lock or unlock the "rw mutex".

This mutex may works as a queue since there is no priority between readers and writers. All they need to do is trying to get involved in the "fcfs" queue as soon as possible. Every reader and writer will compete to get this mutex lock, and this guarantees that there is no starvation.

## 2. The Mutex "r\_mutex"

Same as the previous one. When readers want to do their service, they need to count up or down the  $r\_cnt$ . This is still a critical problem since  $r\_cnt$  should be calculated one by one.

#### 3. The Mutex "rw\_mutex"

The shared mutex when readers or writers run themselves. As the runner can overlap with other runners, only the first runner locks this mutex, and the last mutex unlocks it. However, writers cannot overlap not only with the readers but also other writers. The "rw\_mutex" will be used whenever the writer wants to run itself.

# Codes - The Fair Readers Writers

```
* Copyright 2020, 2021. Heekuck Oh, all rights reserved
*이 프로그램은 한양대학교 ERICA 소프트웨어학부 재학생을 위한 교육용으로 제작되었습니다.
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <pthread.h>
#include <unistd.h>
#include <fcntl.h>
#define N 8192
#define L1 75
#define L2 70
#define L3 70
#define RNUM 10
#define WNUM 3
face picture with a number of chars...
* alive 값이 1이면 각 스레드는 무한 루프를 돌며 반복해서 일을 하고,
* alive 값이 0이 되면 무한 루프를 빠져나와 스레드를 자연스럽게 종료한다.
int alive = 1;
pthread_mutex_t rw_mutex;
pthread mutex tr mutex;
pthread_mutex_t fcfs;
int r_cnt = 0;
* Reader 스레드는 같은 문자를 N번 출력한다. 예를 들면 <AAA...AA> 이런 식이다.
*출력할 문자는 인자를 통해 0이면 A, 1이면 B,..., 등으로 출력하며, 시작과 끝을 <...>로 나타낸다.
* 단일 reader라면 <AAA...AA>처럼 같은 문자만 출력하겠지만, critical section에서 reader의
* 중복을 허용하기 때문에 reader가 많아지면 출력이 어지럽게 섞여서 나오는 것이 정상이다.
void *reader(void *arg)
```

```
{
   int id, i;
   *들어온 인자를 통해 출력할 문자의 종류를 정한다.
   id = *(int *)arg;
    * 스레드가 살아 있는 동안 같은 문자열 시퀀스 <XXX...XX>를 반복해서 출력한다.
   while (alive) {
       pthread_mutex_lock(&fcfs); // reader goes to fcfs q and waits
           pthread_mutex_lock(&r_mutex); // r_cnt cannot be accessed at the same
              if(++r cnt == 1) pthread mutex lock(&rw mutex); // first reader get rw mutex for others
           pthread mutex unlock(&r mutex);
       pthread_mutex_unlock(&fcfs);
       * Begin Critical Section
       printf("<");</pre>
       for (i = 0; i < N; ++i)
       printf("%c", 'A'+id);
       printf(">");
       * End Critical Section
       pthread mutex lock(&r mutex);
       if(--r_cnt == 0) pthread_mutex_unlock(&rw_mutex); // last reader unlock the rw_mutex
                                                     // so that others can use it
       pthread mutex unlock(&r mutex);
   }
   pthread_exit(0);
}
* Writer 스레드는 어떤 사람의 얼굴 이미지를 출력한다.
*이미지는 세 가지 종류가 있으며 인자를 통해 식별한다.
* Writer가 critical section에 있으면 다른 writer는 물론이고 어떠한 reader도 들어올 수 없다.
*만일 이것을 어기고 다른 writer나 reader가 들어왔다면 얼굴 이미지가 깨져서 쉽게 감지된다.
void *writer(void *arg)
{
   int id, i;
   struct timespec req, rem;
   *들어온 인자를 통해 얼굴 이미지의 종류를 정한다.
```

```
id = *(int *)arg;
* 이미지 출력을 천천히 하기 위해 한 줄 출력할 때마다 쉬는 시간을 1 나노초로 설정한다.
req.tv_sec = 0;
req.tv_nsec = 1L;
* 스레드가 살아 있는 동안 같은 이미지를 반복해서 출력한다.
while (alive) {
    pthread_mutex_lock(&fcfs); // writer goes to fcfs q and waits
        pthread_mutex_lock(&rw_mutex); // when writer's turn in fcfs, get rw_mutex
    pthread_mutex_unlock(&fcfs); // complete pushing to q
    * Begin Critical Section
    printf("\n");
    switch (id) {
    case 0:
        for (i = 0; i < L1; ++i) {
            printf("%s\n", t[i]);
            nanosleep(&req, &rem);
        }
        break;
    case 1:
        for (i = 0; i < L2; ++i) {
            printf("%s\n", d[i]);
            nanosleep(&req, &rem);
        }
        break;
    case 2:
        for (i = 0; i < L3; ++i) {
            printf("%s\n", e[i]);
            nanosleep(&req, &rem);
        break;
    default:
    }
    * End Critical Section
    pthread_mutex_unlock(&rw_mutex); // release rw_mutex so that others can use it
pthread_exit(0);
```

}

```
*메인 함수는 RNUM 개의 reader 스레드를 생성하고, WNUM 개의 writer 스레드를 생성한다.
* 생성된 스레드가 일을 할 동안 0.2초 동안 기다렸다가 alive의 값을 0으로 바꿔서 모든 스레드가
*무한 루프를 빠져나올 수 있게 만든 후, 스레드가 자연스럽게 종료할 때까지 기다리고 메인을 종료한다.
int main(void)
   int i;
   int rarg[RNUM], warg[WNUM];
   pthread_t rthid[RNUM];
   pthread_t wthid[WNUM];
   struct timespec req, rem;
   // stdout is now the txt file
   int output_fd = open("fair_reader_writer.txt", O_CREAT|O_TRUNC|O_RDWR, 0666);
   dup2(output fd, STDOUT FILENO);
   pthread_mutex_init(&rw_mutex, NULL);
   pthread mutex init(&r mutex, NULL);
   pthread_mutex_init(&fcfs, NULL);
    * Create RNUM reader threads
   for (i = 0; i < RNUM; ++i) {
       rarg[i] = i;
       if (pthread_create(rthid+i, NULL, reader, rarg+i) != 0) {
           fprintf(stderr, "pthread_create error\n");
           exit(-1);
       }
   }
    * Create WNUM writer threads
   for (i = 0; i < WNUM; ++i) {
       warg[i] = i;
       if (pthread_create(wthid+i, NULL, writer, warg+i) != 0) {
           fprintf(stderr, "pthread create error\n");
           exit(-1);
       }
   }
    * Wait for 0.2 second while the threads are working
   req.tv_sec = 0;
   req.tv_nsec = 200000000L;
   nanosleep(&req, &rem);
```

```
*Now terminate all threads and leave

*/
alive = 0;
for (i = 0; i < RNUM; ++i)
    pthread_join(rthid[i], NULL);
for (i = 0; i < WNUM; ++i)
    pthread_join(wthid[i], NULL);

pthread_mutex_destroy(&rw_mutex);
pthread_mutex_destroy(&r_mutex);
pthread_mutex_destroy(&fcfs);

exit(0);
}
```

# Compile and Run

Compile "fair\_reader\_writer.c" with GCC. When you run the "fair\_read\_writer", you should see there is an output txt file named "fair reader writer.txt".

```
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Systems
  main) 🕯 ls
fair_reader_writer.c writer_prefer
                                           writer_prefer.c
                                                                writer_prefer.txt
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Syst
 (main) # gcc -o fair_reader_writer fair_reader_writer.c
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Syst
 (main) * ./fair_reader_writer
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Syst
 (main) * lslaf
total 3776
drwxr-xr-x@ 8 jihonggeun staff
                                      256 May 14 14:52 ./
                                      288 May 14 14:52 ../
drwxr-xr-x@ 9 jihonggeun
                          staff
                          staff
-rwxr-xr-x 1 jihonggeun
                                   66904 May 14 14:57 fair
-rw-r--r-- 1 jihonggeun staff
-rw-r--r-- 1 jihonggeun staff
                                  103535 May 14 09:47 fair_reader_writer.c
                                  547768 May 14 14:57 fair_reader_writer.txt
-rwxr-xr-x 1 jihonggeun
                          staff
                                  67000 May 14 14:48 writer_prefer*
                                 103742 May 14 09:46 writer_prefer.c
-rw-r--r-- 1 jihonggeun
                          staff
-rw-r--r-- 1 jihonggeun staff 1029049 May 14 14:48 writer_prefer.txt
(base) jihonggeun:~/Library/Mobile Documents/com~apple~CloudDocs/Study/HYU/2021/Operating Syst
 (main) 🔹 📗
```

#### The part of fair\_reader\_writer.txt file -- (1)

```
Readers sometimes precede the Writer
III66I666666666616661666666I
                      .666666666666000000000006
III66I66I66666666666I6I
                       I6III6666I6666666666I.
6161616666666666666
                        16666660600606000000
6161666666666666666666
                        666660060060606000
I666666666661!.....^^.
             ^!!!!!!..
                         16666600000000000
       . ^.^.. ^ ^6III|.
66616616666661^^.^.^^^
                         .660000600000000
       ...^^!^^.^..6066|.
              .||!^!^^^^
666666666661.^.^^^^ . .
                         .660000000000000
600000000006
606000000600
                          6600000006
                          0060000006
                          1006000006
00000006
                          6
                           100000006
                     600
                           00600000
۸Ì
                           00000006
                           00000006
                          Ш
                           60000000
                           00000000
                           00000000
06000000
                     6II660600000000.!
                           06000000
                     !6||IIII606060|60|.
                           00600000
                     !I||I^.0^|I0006!60I
.^. !|I6!006^^0I
.^^.!^^. .6I. ^0
                           60000000
                           00600006
                           60000006
                           0060000
                           060000
                            066006
                            060606
                            160600
...!60600000060006666666666660166016661611||!!!!\.^^!60066|!^^!|61|^^
..^!!006606000006006111161661666666660060661|||!!!|111006661||||11!.
..^!60616060000000661||11116166661106660066661||!!!|||00000666666611|!.
                            .66000
                      İ۷
                            60060
                            66606
```

#### The part of fair reader writer.txt file -- (2)

```
riter sometimes
 the Readers
 ^.|660||^||||1||6660QQQQMQQMQMMQMQMQMQMQMQMQMQMQQQQQ0606661||||||!!^^^. . . . . ||QQQQQQQQQ66666660|||||||!!!^!.^|!
```

(writer sometimes precede the readers)

#### The part of fair reader writer.txt file -- (3)

```
| IOO|! IOO6!000000006666666666666666660I6II06|!I66I|III||!|!!^^^^!
     |0011600!1000000000660666666666666666661611|161111661111|||!^.....^.
                                                                                                                                                                                                            60066
   06066
    ^^0061|60000000011|0666666060660660660600666066161||||!!!^^.
                                                                                                                                                                                                          .060606
    ^!||^I0000000006||!IQ66066060606666661006606616III|||!^^
                                                                                                                                                                                                       10600000
    106000000
  0000000000
                                                                                                                                                                                                6060000000
   ^!!!II000000
   000000|!|6666016||!^!^^!000060000000066616600000000066|!!^1
  000000|||6666016|||1.1.^1.000060000000606616600000000606||1.^.
0006||1.||III66III|||1.1.||1.1.||000000000060660000000666||1...^
006||1.1.||II61||6||||1.1.||1.1.||1.1.||1.0000000060660000006066||1...^
066||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.||1.|
                                                                                                                                      ^^ ^|^1006.^1||!.
                                                                                                                                                                                             ^.^!^.^^.|16
                                                                                                                                                                                        I|^^.^.^.
                                                                                                                                                 0000!.!611!^
                                                                                                                                                                                    ^16|^...^...
^^|61^^.^..
                                                                                                                                              . |60006^ |016 |! ^.
                                                                                                                                                 .1000|1000|.^
                                                                                                                                66
                                                                                                                                                  .1600660006!..
                                                                                                                                                                                   ..^|60!^..^^^^.
                                                                                                                            .60 . ^|.
                                                                                                                                                  |66I|00000006|!!.^..!0|!^^^^.^^!^^
                                                                                                                        ^100 ..!6!
                                                                                                                     !661^
                                                                                                                                                ^600610000000000|...^|66|^.^^^!^!.^
                                                                                                                                                |600|60000000000i...^!06I|!^^^..^^
                                                                                                                                      006!
                                                                                            ^!II|...
   6001
                                                                                                                                    .^|166001.
   606
                                                                                            v|!!vv..
                                                                                                                                    ||II600^
                                                                                                                          !!||||60!
^I!!!||I0
   060!
                                                                                                        ^^ 1!!!||10

^^ 1!!!||1601

.!^ ^\!||1600

.^\! ^!!!||1600

^^! ^!!!||1100

^ !!!||1100^
                                                                                                                                                                ..6006^^!!!!!!00|!^^..^!!|1
I0I0!^..^^||06I!!.^^!|!!
^60I6!!!!!^!|60I!!^.!!!!!
   660!
                                                    ^!!!^.
                                                                                  .16!|!^....
                                                     .^!!!!.^ . .^!6|6I|^^^^
   606!
                                                       ..^!!^!^!!!6!!!!!!|
   606!
                                                                                                                                                              ..^!!|II6II66I6II|!!
   106^
                                               ^. .^^!|I|||!||I||II
.!^. .^|II||I!|!|!!
^!...||||!!!||!!.
    60!
66^
    .064
                                                                                                 ijΛ
                                                                                                              .!!|||||600.
                                                                                                                                                               .!|I66 www.segnalidivita.com
  .^!^^....
                                                                                                                                                                                    ^..^!!^^..^.^..^!^...
```

(writers never overlap)