CSGE602055 Operating Systems
CSF2600505 Sistem Operasi
Minggu 07: Synchronization

Rahmat M. Samik-Ibrahim

Universitas Indonesia

http://rms46.vlsm.org/2/207.html

REV094 7-NOV-2017

OS172 | INT TU/TH 13:00-15:00 | EXT TH 19:00-21:50

Minggu 00	29 Aug - 05 Sep 2017	Intro & Review
Minggu 01	07 Sep - 12 Sep 2017	IPR, SED, AWK, REGEX, & Scripting
Minggu 02	14 Sep - 19 Sep 2017	Protection, Security, Privacy, & C-language
Minggu 03	26 Sep - 30 Sep 2017	BIOS, Loader, Systemd, & I/O
Minggu 04	03 Okt - 07 Okt 2017	Addressing, Shared Lib, Pointer & I/O Programming
Minggu 05	10 Okt - 14 Okt 2017	Virtual Memory
Ming. UTS	15 Okt - 24 Okt 2017	
Minggu 06	26 Okt - 31 Okt 2017	Concurency: Processes & Threads
Minggu 07	02 Nov - 07 Nov 2017	Synchronization
Minggu 08	09 Nov - 14 Nov 2017	Scheduling & Network Sockets Programming
Minggu 09	16 Nov - 21 Nov 2017	File System & Persistent Storage
Minggu 10	23 Nov - 28 Nov 2017	Special Topic: Retreat
Cadangan	30 Nov - 09 Des 2017	
Ming. UAS	10 Des - 23 Des 2017	

Agenda I

Agenda II

Week 07: Synchronization

- Reference: (OSCE2e ch5) (UCB 7/8) (UDA P3L3/4) (OLD 04)
- The Critical Section Problem
- Race Condition
- Peterson's Solution
- Semaphores
- Classical Problems
  - Bounded-Buffer Problem
  - Readers and Writers Problem
  - Dining-Philosophers Problem
- Resource and Allocation Graph

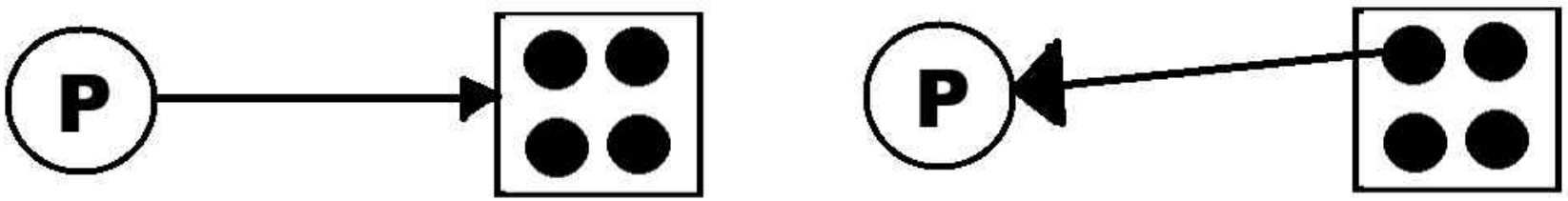

Figure: Request and Holding

Peterson's Solution

Process 0

```
   flag[0]=                    turn=

do {
    flag[0] = true
    turn = 1
    while (flag[1] && turn == 1)
      (do nothing);
    [CRITICAL SECTION];
    flag[0] = false
    [REMAINDER SECTION];
} while(true);
```

Process 1

```
   flag[1]=

do {
    flag[1] = true
    turn = 0
    while (flag[0] && turn == 0)
      (do nothing);
    [CRITICAL SECTION];
    flag[1] = false
    [REMAINDER SECTION];
} while(true);
```

Semaphore

- Dijkstra's Seinpalen (1963): Probeer (Try) en Verhoog (+1)
- Semaphore: Wait(S) and Signal(S)
- Linux System Calls: `sem_init()`, `sem_wait()`, and `sem_post()`

```
# Semaphore (Seinpalen)
# Wait (Probeer)

wait(S) {
   while (S <= 0)
      ; // busy wait
   S--;
}

# Signal (Verhoog)

signal(S) {
   S++;
}
```

Deadlock and Starvation

- Deadlock Characterization
  - Mutual exclusion
  - Hold and wait
  - No preemption
  - Circular wait
- Banker's Algorithm
- Deadlock Prevention
- Deadlock Avoidence
- How do Operating Systems handle Deadlocks?

IGNORE THE PROBLEM!
Pretending that deadlocks never occur
Just **RESET/REBOOT** it
This is how they **DO IT**!

99-myutils.h

```
/*
 * (c) 2011-2016 Rahmat M. Samik-Ibrahim -- This is free software
 */

#define MAX_THREAD 256
#define BUFFER_SIZE  5
#define TRUE         1
#define FALSE        0

typedef struct {
   int   buffer[BUFFER_SIZE];
   int   in;
   int   out;
   int   count;
} bbuf_t;

void daftar_trit   (void* trit);      // mempersiapkan "trit"
void jalankan_trit (void);            // menjalankan dan menunggu hasil dari
                                      // "daftar_trit"
void beberes_trit  (char* pesan);     // beberes menutup "jalankan_trit"

void rehat_acak    (long max_mdetik); // istirohat acak "0-max_mdetik" (ms)

void init_buffer   (void);            // init buffer
void enter_buffer  (int entry);       // enter  an integer item
int remove_buffer  (void);            // remove the item

void init_rw       (void);            // init readers writers
int  startRead     (void);            // start reading
int  endRead       (void);            // end reading
void startWrite    (void);            // start writing
void endWrite      (void);            // end writing
```

99-myutils.c

```
/*
 * (c) 2011-2016 Rahmat M. Samik-Ibrahim -- This is free software
 * Feel free to copy and/or modify and/or distribute it,
 * provided this notice, and the copyright notice, are preserved.
 * REV01 Wed Nov  2 11:49:55 WIB 2016
 * REV00 Xxx Sep 30 XX:XX:XX UTC 2015
 * START Xxx Mar 30 02:13:01 UTC 2011
 */

#include <pthread.h>
#include <semaphore.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#include "99-myutils.h"
sem_t    mutex, db, empty, full, rmutex, wmutex;

/* TRIT ************************************************/
int       jumlah_trit = 0;

void*     trits  [MAX_THREAD];
pthread_t trit_id[MAX_THREAD];

void daftar_trit(void *trit) {
   if(jumlah_trit >= MAX_THREAD) {
      printf("\n ERROR MAX daftar_trit %d\n",jumlah_trit);
      exit(1);
   }
   trits[jumlah_trit++] = trit;
}
```

99-myutils.c (2)

```
void jalankan_trit(void){
   int ii;
   for (ii=0;ii<jumlah_trit;ii++) {
      if(pthread_create(&trit_id[ii], NULL, trits[ii], NULL)) {
         printf("\n ERROR pthread_creat: %d\n",ii);
         exit(1);
      }
   }
   for (ii=0;ii<jumlah_trit;ii++){
      if(pthread_join(trit_id[ii], NULL)) {
         printf("\n ERROR pthread_join: %d\n",ii);
         exit(1);
      }
   }
}
void beberes_trit(char* pesan) {
   if (pesan != NULL)
      printf("%s\n",pesan);
   pthread_exit(NULL);
}
```

99-myutils.c (3)

```
/* REHAT ***************************************************/
int  pertamax    = TRUE;

void rehat_acak(long max_mdetik) {
   struct timespec tim;
   long            ndetik;

   if (pertamax) {
      pertamax = FALSE;
      srandom((unsigned int) time (NULL));
   }
   ndetik      = random() % max_mdetik;
   tim.tv_sec  = ndetik   / 1000L;
   tim.tv_nsec = ndetik   % 1000L * 1000000L;
   nanosleep(&tim,NULL);
}
```

99-myutils.c (4)

```
/* BOUNDED BUFFER ****************************************/
bbuf_t buf;
void init_buffer(void) {
   buf.in    = 0;
   buf.out   = 0;
   buf.count = 0;
   sem_init   (&mutex, 0, 1);
   sem_init   (&empty, 0, BUFFER_SIZE);
   sem_init   (&full,  0, 0);
}

void enter_buffer(int entry) {
   sem_wait(&empty);
   sem_wait(&mutex);
   buf.count++;
   buf.buffer[buf.in] = entry;
   buf.in = (buf.in+1) % BUFFER_SIZE;
   sem_post(&mutex);
   sem_post(&full);
}

int remove_buffer(void) {
   int item;
   sem_wait(&full);
   sem_wait(&mutex);
   buf.count--;
   item = buf.buffer[buf.out];
   buf.out = (buf.out+1) % BUFFER_SIZE;
   sem_post(&mutex);
   sem_post(&empty);
   return item;
}
```

99-myutils.c (5)

```
/* READERS WRITERS ************************************/
int readerCount;
void init_rw(void) {
   readerCount = 0;
   sem_init   (&mutex,  0, 1);
   sem_init   (&rmutex, 0, 1);
   sem_init   (&wmutex, 0, 1);
   sem_init   (&db,     0, 1);
}

int startRead(void) {
   sem_wait(&mutex);
   if (++readerCount == 1 )
      sem_wait(&db);
   sem_post(&mutex);
   return readerCount;
}

int endRead(void) {
   sem_wait(&mutex);
   if (--readerCount == 0 )
      sem_post(&db);
   sem_post(&mutex);
   return readerCount;
}

void startWrite(void) {
   sem_wait(&db);
}
void endWrite(void) {
   sem_post(&db);
}
```

00-thread

```
/* (c) 2015-2017 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This is free software.
 * REV03 Wed Nov  1 15:17:08 WIB 2017
 * REV02 Tue Apr 18 15:28:19 WIB 2017
 * REV01 Wed Nov  2 11:49:30 WIB 2016
 * START Xxx Sep 30 XX:XX:XX UTC 2015
 */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <semaphore.h>
#include "99-myutils.h"
#define  LOOP0    10
#define  LOOP1   500
#define  LOOP2  1000
#define  LOOP3 10000

volatile int loop = LOOP0;
int          share;
```

00-thread (2)

```
void* thread1 (void* a) {
   int ii, jj, kk;
   printf("I am a thread no 1\n");
   sleep(1);
   share = 1000;
   while (loop > 0) {
      for (ii=0;ii<LOOP1;ii++) {
         for (jj=0;jj<LOOP2;jj++) {
              ;
         }
      }
      share++;
   }
}

void* thread2 (void* a) {
   int ii, jj, kk;
   printf("I am a thread no 2\n");
   sleep(1);
   share = 2000;
   while (loop > 0) {
      for (ii=0;ii<LOOP1;ii++) {
         for (jj=0;jj<LOOP2;jj++) {
              ;
         }
      }
      share--;
   }
}
```

00-thread (3)

```
void* thread3 (void* a) {
   int ii, jj, kk;
   printf("I am a thread no 3\n");
   sleep(1);
   while (loop-- > 0) {
      for (ii=0;ii<LOOP3;ii++) {
         for (jj=0;jj<LOOP3;jj++) {
               ;
         }
      }
      printf("SHARE = %4.4d\n", share);
   }
}
void main(void) {
   daftar_trit   (thread1);
   daftar_trit   (thread2);
   daftar_trit   (thread3);
   jalankan_trit ();
   printf        ("I am MAIN\n");
   beberes_trit  ("Done...");
}
```

00-thread (4)

```
>>>>> $ 00-thread
I am a thread no 1
I am a thread no 2
I am a thread no 3
SHARE = 1994
SHARE = 1989
SHARE = 1985
SHARE = 1977
SHARE = 1966
SHARE = 1954
SHARE = 1944
SHARE = 1933
SHARE = 1923
SHARE = 1923
I am MAIN
Done...
>>>>> $ 00-thread
I am a thread no 2
I am a thread no 1
I am a thread no 3
SHARE = 0992
SHARE = 0985
SHARE = 0987
SHARE = 0994
SHARE = 0991
SHARE = 0982
SHARE = 0974
SHARE = 0967
SHARE = 0959
SHARE = 0959
I am MAIN
Done...
>>>>> $
```

01-thread

```
>>>>> $ cat 01-thread.c
/*
 * (c) 2015-2017 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This is free software.
 * REV02 Wed Nov  1 16:48:40 WIB 2017
 * REV01 Wed Nov  2 11:49:39 WIB 2016
 * START Xxx Sep 30 XX:XX:XX UTC 2015
 */

#include <stdio.h>
#include <stdlib.h>
#include <semaphore.h>
#include "99-myutils.h"

sem_t generik;
sem_t generik2;
```

01-thread (2)

```
void* thread1 (void* a) {
   sem_wait   (&generik);
   printf("THREAD1: I am second!\n");
   sem_post   (&generik2);
}

void* thread2 (void* a) {
   printf("THREAD2: I am first!\n");
   sem_post   (&generik);
}

void* thread3 (void* a) {
   sem_wait   (&generik2);
   printf("THREAD3: I am last!\n");
}

void main(void) {
   sem_init      (&generik,  0, 0);
   sem_init      (&generik2, 0, 0);
   daftar_trit   (thread1);
   daftar_trit   (thread2);
   daftar_trit   (thread3);
   jalankan_trit ();
   beberes_trit  ("Bye Bye Main...");
}

>>>>> $ 01-thread
THREAD2: I am first!
THREAD1: I am second!
THREAD3: I am last!
Bye Bye Main...
```

02-prodkon

```
>>>>> $ cat 02-prodkon.c
/*
 * (c) 2011-2017 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This is free software.
 * REV02 Wed Nov  1 16:50:50 WIB 2017
 * REV01 Wed Nov  2 11:20:30 WIB 2016
 * REV00 Xxx Sep 30 XX:XX:XX UTC 2012
 * START Xxx Mar 30 02:13:01 UTC 2011
 */

#include <stdio.h>
#include <stdlib.h>
#include "99-myutils.h"

#define P_REHAT 2000
#define K_REHAT 2000
int produk = 0;

void* Produsen (void* a) {
   printf("Produsen siap...\n");
   while (TRUE) {
      printf("P: REHAT ******\n");
      rehat_acak(P_REHAT);
      printf("P: PRODUKSI %d\n", produk);
      enter_buffer  (produk++);
   }
}
```

02-prodkon (2)

```
void* Konsumen (void* a) {
   printf   ("                          Konsumen siap...\n");
   while (TRUE) {
      printf("                          K: REHAT ******\n");
      rehat_acak(K_REHAT);
      printf("                          K: KONSUMSI %d\n", remove_buffer());
   }
}
int main(int argc, char * argv[])
{
   init_buffer();
   daftar_trit(Produsen);
   daftar_trit(Konsumen);
   jalankan_trit();
   beberes_trit("Selese...");
}
##################
>>>>> $ ./02-prodkon
Produsen siap...
P: REHAT ******
                        Konsumen siap...
                        K: REHAT ******
P: PRODUKSI 0
P: REHAT ******
                        K: KONSUMSI 0
                        K: REHAT ******
P: PRODUKSI 1
P: REHAT ******
P: PRODUKSI 2
P: REHAT ******
                        K: KONSUMSI 1
                        K: REHAT ******
```

03-readwrite

```
>>>>> $ cat 03-readwrite.c

/*
 * (c) 2011-2017 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This is free software.
 * REV02 Wed Nov  1 16:53:38 WIB 2017
 * REV01 Wed Nov  2 13:49:55 WIB 2016
 * REV00 Xxx Sep 30 XX:XX:XX UTC 2015
 * START Xxx Mar 30 02:13:01 UTC 2011
 */

#include <stdio.h>
#include <stdlib.h>
#include <semaphore.h>
#include "99-myutils.h"

extern sem_t    mutex, db, empty, full, rmutex, wmutex;

#define R_REHAT 4000
#define R_READ  4000
#define R_JUMLAH   4

#define W_REHAT 2000
#define W_WRITE 2000
#define W_JUMLAH   3

int reader_ID = 0;
int writer_ID = 0;
```

03-readwrite (2)

```
void* Reader (void* a) {
   int  my_ID;

   sem_wait (&rmutex);
   my_ID  = reader_ID++;
   sem_post (&rmutex);

   printf   ("                EADER %d: SIAP  ******\n", my_ID);
   while (TRUE) {
      printf("                EADER %d: REHAT ******\n", my_ID);
      rehat_acak(R_REHAT);
      printf("                EADER %d: MAU  MEMBACA\n", my_ID);
      printf("                **** JUMLAH PEMBACA %d\n", startRead());
      printf("                EADER %d:=SEDANG==BACA\n", my_ID);
      rehat_acak(R_READ);
      printf("                EADER %d: SELESAI BACA\n", my_ID);
      printf("                **** SISA PEMBACA %d\n", endRead());
   }
}
```

03-readwrite (3)

```
void* Writer (void* a) {
   int  my_ID;

   sem_wait (&wmutex);
   my_ID  = writer_ID++;
   sem_post (&wmutex);

   printf   ("WRITER %d: SIAP  *******\n", my_ID);
   while (TRUE) {
      printf("WRITER %d: REHAT *******\n", my_ID);
      rehat_acak(W_REHAT);
      printf("WRITER %d: MAU   MENULIS\n", my_ID);
      startWrite();
      printf("WRITER %d:=SEDANG==NULIS\n", my_ID);
      rehat_acak(W_WRITE);
      endWrite();
      printf("WRITER %d: SELESAI NULIS\n", my_ID);
   }
}

int main(int argc, char * argv[])
{
   int ii;
   init_rw();
   for (ii = 0 ; ii < R_JUMLAH; ii++)
      daftar_trit(Reader);
   for (ii = 0 ; ii < W_JUMLAH; ii++)
      daftar_trit(Writer);
   jalankan_trit();
   beberes_trit("Selese...");
}
```

03-readwrite (4)

```
>>>>> $ 03-readwrite
                        READER 1: SIAP  ******
                        READER 1: REHAT ******
                        READER 0: SIAP  ******
                        READER 0: REHAT ******
WRITER 1: SIAP  *******
WRITER 1: REHAT *******
                        READER 3: SIAP  ******
                        READER 3: REHAT ******
                        READER 2: SIAP  ******
                        READER 2: REHAT ******
WRITER 2: SIAP  *******
WRITER 2: REHAT *******
WRITER 0: SIAP  *******
WRITER 0: REHAT *******
WRITER 2: MAU   MENULIS
WRITER 2:=SEDANG==NULIS
                        READER 3: MAU  MEMBACA
                        READER 1: MAU  MEMBACA
WRITER 2: SELESAI NULIS
WRITER 2: REHAT *******
                        ***** JUMLAH PEMBACA 2
                        READER 1:=SEDANG==BACA
                        ***** JUMLAH PEMBACA 1
                        READER 3:=SEDANG==BACA
WRITER 1: MAU   MENULIS
                        READER 1: SELESAI BACA
                        ***** SISA PEMBACA 1
                        READER 1: REHAT ******
WRITER 0: MAU   MENULIS
                        READER 3: SELESAI BACA
```

04-readwrite

```
>>>>> $ cat 04-readwrite.c
/*
 * (c) 2011-2017 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This is free software.
 * REV04 Mon Nov  6 20:20:29 WIB 2017
 * REV02 Fri Apr 28 10:06:07 WIB 2017
 * REV00 Xxx Sep 30 XX:XX:XX UTC 2015
 * START Xxx Mar 30 02:13:01 UTC 2011
 */

#include <stdio.h>
#include <stdlib.h>
#include <semaphore.h>
#include "99-myutils.h"

extern sem_t    mutex, db, empty, full, rmutex, wmutex;
sem_t           sync_er, sync_re, sync_ew, sync_we;

#define R_REHAT 1500
#define R_READ  1500
#define R_JUMLAH  2

#define W_REHAT 1500
#define W_WRITE 1500
#define W_JUMLAH  2

int reader_ID = 0;
int writer_ID = 0;
```

04-readwrite (2)

```
void* Reader (void* a) {
   int  my_ID;

   sem_wait (&rmutex);
   my_ID  = reader_ID++;
   sem_post (&rmutex);

   printf   ("                          READER %d: SIAP  ******\n", my_ID);
   while (TRUE) {
      sem_wait (&sync_er);
      printf("                          READER %d: REHAT ******\n", my_ID);
      rehat_acak(R_REHAT);
      printf("                          READER %d: MAU  MEMBACA\n", my_ID);
      printf("                          ***** JUMLAH PEMBACA %d\n", startRead());
      printf("                          READER %d:=SEDANG==BACA\n", my_ID);
      rehat_acak(R_READ);
      printf("                          READER %d: SELESAI BACA\n", my_ID);
      printf("                          ***** SISA PEMBACA %d\n", endRead());
      sem_post (&sync_re);
   }
}
```

04-readwrite (3)

```
void* Writer (void* a) {
   int  my_ID;

   sem_wait (&wmutex);
   my_ID  = writer_ID++;
   sem_post (&wmutex);

   printf   ("WRITER %d: SIAP  *******\n", my_ID);
   while (TRUE) {
      printf("WRITER %d: REHAT *******\n", my_ID);
      rehat_acak(W_REHAT);
      printf("WRITER %d: MAU   MENULIS\n", my_ID);
      startWrite();
      printf("WRITER %d:=SEDANG==NULIS\n", my_ID);
      rehat_acak(W_WRITE);
      endWrite();
      printf("WRITER %d: SELESAI NULIS\n", my_ID);
      sem_post (&sync_we);
      sem_wait (&sync_ew);
   }
}
```

04-readwrite (4)

```
void* Extra (void* a) {
   int ii;
   while (TRUE) {
      for (ii=0; ii<W_JUMLAH; ii++)
         sem_wait (&sync_we);
      for (ii=0; ii<R_JUMLAH; ii++)
         sem_post (&sync_er);
      for (ii=0; ii<R_JUMLAH; ii++)
         sem_wait (&sync_re);
      for (ii=0; ii<W_JUMLAH; ii++)
         sem_post (&sync_ew);
   }
}

int main(int argc, char * argv[])
{
   int ii;
   init_rw();
   sem_init   (&sync_er, 0, 0);
   sem_init   (&sync_re, 0, 0);
   sem_init   (&sync_ew, 0, 0);
   sem_init   (&sync_we, 0, 0);
   daftar_trit(Extra);
   for (ii = 0 ; ii < R_JUMLAH; ii++)
      daftar_trit(Reader);
   for (ii = 0 ; ii < W_JUMLAH; ii++)
      daftar_trit(Writer);
   jalankan_trit();
   beberes_trit("Selese...");
}
```

04-readwrite (5)

```
>>>>> $ 04-readwrite
                        READER 1: SIAP  ******
                        READER 0: SIAP  ******
WRITER 0: SIAP  *******
WRITER 0: REHAT *******
WRITER 1: SIAP  *******
WRITER 1: REHAT *******
WRITER 1: MAU   MENULIS
WRITER 1:=SEDANG==NULIS
WRITER 0: MAU   MENULIS
WRITER 0:=SEDANG==NULIS
WRITER 1: SELESAI NULIS
WRITER 0: SELESAI NULIS
                        READER 1: REHAT ******
                        READER 0: REHAT ******
                        READER 1: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 1
                        READER 1:=SEDANG==BACA
                        READER 1: SELESAI BACA
                        ***** SISA PEMBACA 0
                        READER 0: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 1
                        READER 0:=SEDANG==BACA
                        READER 0: SELESAI BACA
                        ***** SISA PEMBACA 0
WRITER 1: REHAT *******
WRITER 0: REHAT *******
WRITER 0: MAU   MENULIS
WRITER 0:=SEDANG==NULIS
```

05-alu

```
>>>>> $ cat 05-alu.c
/* (c) 2013-2017 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This is free software.
 * REV02 Wed Nov  1 17:16:35 WIB 2017
 * REV01 Wed Nov  2 13:50:33 WIB 2016
 * START Xxx Xxx XX XX:XX:XX UTC 2013
 */

#include <stdio.h>
#include <stdlib.h>
#include <semaphore.h>
#include "99-myutils.h"

#define      NThreads 4

sem_t        mutex,   switch1, switch2;
int          addvar1, addvar2, addresult;
int          subvar1, subvar2, subresult;
int          mulvar1, mulvar2, mulresult;
int          divvar1, divvar2, divresult;

void* add (void* a) {
   sem_post  (&switch1);
   sem_wait  (&switch2);

   sem_wait  (&mutex);
   printf("Add starts \n");
   addresult = addvar1 + addvar2;
   sem_post  (&mutex);
   sem_post  (&switch1);
}
```

05-alu (2)

```
void* subtract (void* a) {
   sem_post  (&switch1);
   sem_wait  (&switch2);

   sem_wait  (&mutex);
   printf("Subtract starts \n");
   subresult = subvar1 - subvar2;
   sem_post  (&mutex);
   sem_post  (&switch1);
}

void* multiply (void* a) {
   sem_post  (&switch1);
   sem_wait  (&switch2);
   sem_wait  (&mutex);
   printf("Multiply starts \n");
   mulresult = mulvar1 * mulvar2;
   sem_post  (&mutex);
   sem_post  (&switch1);
}

void* divide (void* a) {
   printf("Divide starts \n");
   sem_post  (&switch1);
   sem_wait  (&switch2);
   sem_wait  (&mutex);
   divresult = divvar1 / divvar2;
   sem_post  (&mutex);
   sem_post  (&switch1);
}
```

05-alu (3)

```
void* manager (void* a) {
   printf("Manager starts \n");

   for (int ii=0; ii< NThreads;ii++)
       sem_wait (&switch1);
   sem_wait  (&mutex);
   addvar1 = 5;
   addvar2 = 2;
   subvar1 = 7;
   subvar2 = 2;
   mulvar1 = 2;
   mulvar2 = 3;
   divvar1 = 4;
   divvar2 = 2;
   sem_post  (&mutex);

   for (int ii=0; ii< NThreads;ii++)
       sem_post (&switch2);
   for (int ii=0; ii< NThreads;ii++)
       sem_wait (&switch1);
   printf("Result: %d + %d = %d\n", addvar1, addvar2, addresult);
   printf("Result: %d - %d = %d\n", subvar1, subvar2, subresult);
   printf("Result: %d * %d = %d\n", mulvar1, mulvar2, mulresult);
   printf("Result: %d / %d = %d\n", divvar1, divvar2, divresult);
}
```

05-alu (4)

```
void main(void) {
   sem_init      (&mutex,   0, 1);
   sem_init      (&switch1, 0, 0);
   sem_init      (&switch2, 0, 0);
   daftar_trit   (manager);
   daftar_trit   (add);
   daftar_trit   (subtract);
   daftar_trit   (multiply);
   daftar_trit   (divide);
   jalankan_trit ();
   beberes_trit  ("Done...");
}
```

```
>>>>> $ 05-alu
Manager starts
Divide starts
Add starts
Subtract starts
Multiply starts
Result: 5 + 2 = 7
Result: 7 - 2 = 5
Result: 2 * 3 = 6
Result: 4 / 2 = 2
Done...
>>>>> $
```

06-balap

```
>>>>> $ cat 06-balap.c
/*
 * (c) 2012-2017 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This is free software.
 * REV02 Wed Nov  1 17:22:23 WIB 2017
 * REV01 Wed Nov  2 11:20:30 WIB 2016
 * REV00 Xxx Sep 30 XX:XX:XX UTC 2015
 * START Xxx Mar 30 02:13:01 UTC 2012
 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <semaphore.h>
#include "99-myutils.h"

#define lamaRehat  250
#define jmlPembalap 12
sem_t mutex, start;

void* bandar (void* a) {
   for (int ii=0; ii<jmlPembalap; ii++)
      sem_wait (&start);
   sem_wait (&mutex);
   sleep(2);
   rehat_acak(lamaRehat);
   printf ("Bandar Siap!\n");
   fflush(NULL);
   sem_post (&mutex);
}
```

06-balap (2)

```
int idmaster = 1;
int juara = 1;
int menang = TRUE;
void* pembalap (void* a) {
   int id;
   sem_wait (&mutex);
   id = idmaster++;
   sem_post (&mutex);
   sem_post (&start);
   rehat_acak(lamaRehat);
   printf ("Pembalap %2.2d Siap!\n",id);
   fflush(NULL);
   rehat_acak(lamaRehat);
   rehat_acak(lamaRehat);
   sem_wait (&mutex);
   if (menang==TRUE) printf("HORE, pemain");
   else printf("Aduh, pemain");
   printf(" %2.2d juara %2.2d!\n",id,juara++);
   menang = FALSE;
   sem_post (&mutex);
}

void main(void) {
   sem_init (&mutex, 0, 1);
   sem_init (&start, 0, 0);
   daftar_trit (bandar);
   for (int ii=0; ii<jmlPembalap; ii++)
      daftar_trit (pembalap);
   jalankan_trit ();
   beberes_trit ("Selese...");
}
```

06-balap (3)

```
>>>>> $ 06-balap
Pembalap 06 Siap!
Pembalap 01 Siap!
Pembalap 02 Siap!
Pembalap 05 Siap!
Pembalap 04 Siap!
Pembalap 03 Siap!
Pembalap 08 Siap!
Pembalap 12 Siap!
Pembalap 10 Siap!
Pembalap 09 Siap!
Pembalap 11 Siap!
Pembalap 07 Siap!
Bandar Siap!
HORE, pemain 08 juara 01!
Aduh, pemain 02 juara 02!
Aduh, pemain 05 juara 03!
Aduh, pemain 12 juara 04!
Aduh, pemain 10 juara 05!
Aduh, pemain 11 juara 06!
Aduh, pemain 06 juara 07!
Aduh, pemain 01 juara 08!
Aduh, pemain 03 juara 09!
Aduh, pemain 09 juara 10!
Aduh, pemain 04 juara 11!
Aduh, pemain 07 juara 12!
Selese...
>>>>> $
```

07-sudokuSV

```
>>>>> $ cat 07-sudokuSV.c
/*
 * (c) 2015 M. Anwar Ma'sum and R.M. Samik-Ibrahim
 * (c) 2016-2017 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This is free software.
 * SSV: Sudoku Solution Validator
 * REV02 Wed Nov  1 18:04:38 WIB 2017
 * REV01 Wed Nov  2 11:20:30 WIB 2016
 */
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#include "99-myutils.h"
#define  V_THREADS 27

int   idSequence = 0;
sem_t mutex, sync;
char  result[3][9];
int   sudoku[9][9] = {   /* Check this 9x9 matrix */
   {5,3,4, 7,6,8, 9,1,2},
   {6,7,2, 1,9,5, 3,4,8},
   {1,9,8, 3,4,2, 5,6,7},

   {8,5,9, 6,7,1, 4,2,3},
   {4,2,6, 8,5,3, 7,9,1},
   {7,1,3, 9,2,4, 8,5,6},

   {9,6,1, 5,3,7, 2,8,4},
   {2,8,7, 4,1,9, 6,3,5},
   {3,4,5, 2,8,6, 1,7,9}
};
```

07-sudokuSV (2)

```
char validate(int iINIT,int iEND,int jINIT,int jEND) {
   int  ii, jj;
   char flag[9];

   for (ii = 0; ii < 9; ii++) flag[ii] = 'F';
   for (ii = iINIT; ii < iEND; ii++) {
      for (jj = jINIT; jj < jEND; jj++) {
         if (flag[sudoku[ii][jj]-1] == 'F')
             flag[sudoku[ii][jj]-1] =  'T';
         else
             return 'F';
      }
   }
   return 'T';
}

void *reporter (void *p) {
   int ii,jj;
   for (ii = 0; ii < V_THREADS; ii++)
      sem_wait(&sync);
   for (ii = 0; ii < 3; ii++) {
      if      (ii == 0) printf ("ROW Validators: ");
      else if (ii == 1) printf ("COL Validators: ");
      else              printf ("BOX Validators: ");
      for (jj = 0; jj < 9; jj++)
         printf("%c ", result[ii][jj]);
      printf("\n");
   }
}
```

07-sudokuSV (3)

```
void *sudokuValidator (void *param) {
    int my_ID, tmp0, tmp1;
    char check;

    sem_wait(&mutex);
    my_ID = idSequence++;
    sem_post(&mutex);

    if (my_ID < 9) {
        check = validate (my_ID, my_ID+1, 0, 9);
    } else if (my_ID < 18) {
        check = validate (0,9,my_ID%9,my_ID%9+1);
    } else {
        tmp0  = ((my_ID%9)/3)*3;
        tmp1  = ((my_ID%9)%3)*3;
        check = validate (tmp0,tmp0+3,tmp1,tmp1+3);
    }

    sem_wait(&mutex);
    result[(my_ID/9)][(my_ID%9)] = check;
    sem_post(&mutex);
    sem_post(&sync);
}
```

07-sudokuSV (4)

```
void main(void *v) {
   int ii, jj;
   printf("SSV: Sudoku Solution Validator\n\n");
   for (ii=0; ii<9; ii++) {
      for (jj=0; jj<9; jj++) {
         printf("%d ", sudoku[ii][jj]);
         if ((jj%3) == 2)
            printf(" ");
      }
      printf ("\n");
      if ((ii%3) == 2)
         printf("\n");
   }
   sem_init(&mutex,0,1);
   sem_init(&sync, 0,0);
   daftar_trit(reporter);
   for (ii = 0; ii < V_THREADS; ii++)
      daftar_trit(sudokuValidator);
   jalankan_trit();
   beberes_trit("Done...");
}
```

07-sudokuSV (5)

```
SSV: Sudoku Solution Validator

5 3 4  7 6 8  9 1 2
6 7 2  1 9 5  3 4 8
1 9 8  3 4 2  5 6 7

8 5 9  6 7 1  4 2 3
4 2 6  8 5 3  7 9 1
7 1 3  9 2 4  8 5 6

9 6 1  5 3 7  2 8 4
2 8 7  4 1 9  6 3 5
3 4 5  2 8 6  1 7 9

ROW Validators: T T T T T T T T T
COL Validators: T T T T T T T T T
BOX Validators: T T T T T T T T T

5 3 4  7 6 8  9 1 2
6 9 2  1 9 5  3 4 8
1 9 8  3 4 2  5 6 7

8 5 9  6 7 1  4 2 3
4 2 6  8 5 3  7 9 1
7 1 3  9 2 4  8 5 6

9 6 1  5 3 7  2 8 4
2 8 7  4 1 9  6 3 5
3 4 5  2 8 6  1 7 9
ROW Validators: T F T T T T T T T
COL Validators: T F T T T T T T T
BOX Validators: F T T T T T T T T
```

08-mainDadu

```
>>>>> $ cat 08-mainDadu.c
/*
 * (c) 2012-2017 Rahmat M. Samik-Ibrahim
 * http://rahmatm.samik-ibrahim.vlsm.org/
 * This is free software.
 * REV02 Wed Nov  1 18:16:14 WIB 2017
 * REV01 Wed Nov  2 11:20:30 WIB 2016
 * REV00 Xxx Sep 30 XX:XX:XX UTC 2015
 * START Xxx Mar 30 02:13:01 UTC 2012
 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <semaphore.h>
#include "99-myutils.h"

#define P_REHAT  400
#define K_REHAT 2000
#define WINpoint  12

sem_t mutex1;

int   idmaster=0;
int   winner=0;
```

08-mainDadu (2)

```
void* Dice (void* a) {
   int dadu;
   printf("The Dice is ready...\n");
   while (TRUE) {
      rehat_acak(P_REHAT);
      dadu=(random() % 6) + 1;
      printf("Dice value %d\n", dadu);
      enter_buffer  (dadu);
      if (winner !=0) {
         enter_buffer  (dadu);
         enter_buffer  (dadu);
         enter_buffer  (dadu);
         enter_buffer  (dadu);
         enter_buffer  (dadu);
         enter_buffer  (dadu);
         break;
      }
   }
}
```

08-mainDadu (3)

```
void* Player (void* a) {
   int id, prev=0, total=0;
   sem_wait (&mutex1);
   id=idmaster++;
   sem_post (&mutex1);
   printf   ("                          Player %d is ready...\n",id);
   while (total < WINpoint) {
      rehat_acak(K_REHAT);
      prev   = total;
      total += remove_buffer();
      if (winner !=0) break;
      printf("                          Player %d's points: %2d [plus %d] \n",
                                        id, total, total-prev);
   }
   if (winner != 1)
      printf("                          Player %d WINS!!!! (%d)\n", id, total);
   winner = 1;
   printf("                        Player %d EXIT\n", id);
}
```

08-mainDadu (4)

```
int main(int argc, char * argv[]) {
   printf("The first player -- with more than %d points -- wins **** ****\n", WINpoint);
   sleep(1);
   sem_init (&mutex1, 0, 1);
   init_buffer();
   daftar_trit(Dice);
   daftar_trit(Player);
   daftar_trit(Player);
   daftar_trit(Player);
   daftar_trit(Player);
   daftar_trit(Player);
   jalankan_trit();
   beberes_trit("Done...");
}
```

08-mainDadu (4)

```
The first player -- with more than 12 points -- wins **** ****
The Dice is ready...
                        Player 0 is ready...
                        Player 2 is ready...
                        Player 3 is ready...
                        Player 4 is ready...
                        Player 1 is ready...
Dice value 3
                        Player 3's points:  3 [plus 3]
Dice value 5
Dice value 2
                        Player 4's points:  5 [plus 5]
Dice value 5
Dice value 2
Dice value 6
                        Player 3's points:  5 [plus 2]
                        Player 0's points:  5 [plus 5]
                        Player 0's points:  7 [plus 2]
                        Player 1's points:  6 [plus 6]
Dice value 5
                        Player 2's points:  5 [plus 5]
Dice value 2
                        Player 4's points:  7 [plus 2]
Dice value 5
                        Player 0's points: 12 [plus 5]
                        Player 0 WINS!!!! (12)
                        Player 0 EXIT
Dice value 5
                        Player 3 EXIT
                        Player 4 EXIT
                        Player 1 EXIT
                        Player 2 EXIT
Done
```

Rock Paper Scissors Lizard Spock

```
/*
 * (c) 2014-2016 Rahmat M. Samik-Ibrahim
 * -- This is free software
 * Feel free to copy and/or modify and/or
 * distribute it,  provided this notice, and
 * the copyright notice, are preserved.
 * REV01 Wed Nov  2 11:20:30 WIB 2016
 * REV00 Xxx Sep 30 XX:XX:XX UTC 2015
 * START Xxx Oct 19 XX:XX:XX UTC 2014
 */
```

RPSLS 1

```
// *Rock*Paper*Scissors*Lizard*Spock*
// Invented by Sam Kass and Karen Bryla
// Rock crushes Scissors
// Rock crushes Lizard
// Paper covers Rock
// Paper disproves Spock
// Scissors cut Paper
// Scissors decapitate Lizard
// Lizard eats Paper
// Lizard poisons Spock
// Spock vaporizes Rock
// Spock smashes Scissors
```

RPSLS 2

```
#include <semaphore.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <unistd.h>
#include "99-myutils.h"
#define nPlayers 2
#define nWeapons 5
int     playerSEQ=1;
int     myWeapon[nPlayers+1];
sem_t   mutex, sync1, sync2;

// (0=Rock) (1=Paper) (2=Scissors) (3=Lizard) (4=Spock)
char* weaponName[nWeapons]= {
   "Rock", "Paper", "Scissors", "Lizard", "Spock"
};
```

RPSLS 3

```
// '-' = draw  'v' = win  'x' = lose
char weaponTable[nWeapons][nWeapons] = {
   {'-','x','v','v','x'},
   {'v','-','x','x','v'},
   {'x','v','-','v','x'},
   {'x','v','x','-','v'},
   {'v','x','v','x','-'}
};
void waitPlayers() {
   for (int ii=0; ii < nPlayers; ii++)
      sem_wait(&sync1);
}
void postPlayers() {
   for (int ii=0; ii < nPlayers; ii++)
      sem_post(&sync2);
}
```

RPSLS 4

```
void* playerThread (void* a) {
   int      playerID;
   sem_wait (&mutex);
   playerID=playerSEQ++;
   sem_post (&mutex);
   printf("Player[%d]: READY\n",playerID);
   sem_post (&sync1);
   sem_wait (&sync2);
   myWeapon[playerID] = rand() % nWeapons;
   printf("Player[%d]: %s\n",
      playerID, weaponName[myWeapon[playerID]]);
   sem_post (&sync1);
}
```

RPSLS 5

```
void* refereeThread (void* a) {
   waitPlayers();
   printf("Referee:   ALL READY!\n");
   postPlayers();
   waitPlayers();
   char result =
      weaponTable[myWeapon[1]][myWeapon[2]];
   if (result == '-')
      printf("Referee:   DRAW!\n");
   else if (result == 'v')
      printf("Referee:   Player[1] WINS!\n");
   else
      printf("Referee:   Player[2] WINS!\n");
}
```

RPSLS 6

```
void main() {
   // randomize with a time seed
   srand(time(NULL));
   sleep(1);
   // init semaphore mutex = 1 syncx = 0
   sem_init (&mutex, 0, 1);
   sem_init (&sync1, 0, 0);
   sem_init (&sync2, 0, 0);
   // register and execute threads
   daftar_trit (refereeThread);
   for (int ii=0; ii<nPlayers; ii++)
      daftar_trit (playerThread);
   jalankan_trit ();
   beberes_trit  ("Goodbye...");
}
```

tba

```
tba
```

The End

- This is the end of the presentation.