

# **INSTYTUT TELEINFORMATYKI I AUTOMATYKI**

## **Wydział Cybernetyki WAT**

**Przedmiot: SYSTEMY OPERACYJNE**  
**SPRAWOZDANIE Z ĆWICZENIA LABORATORYJNEGO Nr 12**

**Temat ćwiczenia: SEMAFORY/PAMIĘĆ DZIELONA**

**Wykonał:**

Piotr Matyjek  
Grupa: **I6Y3S1**

**Ćwiczenie wykonane dnia**  
08.01.2018

**Prowadzący ćwiczenie**  
mgr. inż. Stanisław Matusiak

**Ocena:**

.....

## Opis rozwiązania

Został mi przydzielony numer zadania 2. W obu programach wykorzystywane są takie same funkcje do podnoszenia i opuszczania semafora. Statyczna struktura buf wykorzystywana jest w funkcjach „podnieś” i „opusc”. Zgodnie z nazwami służą one do odpowiednio podnoszenia i opuszczania semafora o określonym semid i semnum.

W programie „Matyjek\_czytelnik2.c” następnie są tworzone potrzebne zmienne i tworzony jest zestaw dwóch semaforów i ustawiane są tymczasowe wartości. Następnie tworzony jest obszar pamięci dzielonej o wielkości 120 bajtów i wskaźnik do tego obszaru przypisywany jest do zmiennej „buf\_shm”. Po czym jako pierwszy znak w tej tablicy jest ustawiany znak ‘1’. Zostało to zrobione, aby program na pewno wszedł w pętlę while z warunkiem `buf_shm[0]!='\0'`. Następnie tworzony program stara się opuścić pierwszy semafor, lecz dopóki program pisarza nie podniesie tego semafora, program czytelnika nie odczyta linijki z pamięci dzielonej. Gdy semafor zostanie podniesiony przez pisarza, czytelnik go opuszcza, kopiuje tablicę znaków do swojej tablicy, a następnie podnosi semafor na który oczekuje pisarz. Następnie wypisuje na ekran przeczytaną linijkę i zapętla się. Jeżeli przeczytanym znakiem było ‘\0’ program wychodzi z pętli i kończy pracę z kodem 0.

Program „Matyjek\_pisarz2.c” jest uruchamiany po utworzeniu wszystkich czytelników. Jest to bardzo ważny wymóg, gdyż gdyby dodać czytelnika po uruchomieniu pisarza, w pewnym momencie programy zawiesiłyby się na semaforach. Jest to spowodowane tym, że po uzyskiwaniu dostępu do zestawu semaforów, program pisarza ustawia początkową wartość pierwszego semafora, na liczbę procesów oczekujących na podniesienie semafora drugiego. Po uzyskaniu dostępu do pamięci współdzielonej, program otwiera swój plik źródłowy do odczytu i wchodzi w pętlę while wykonującą się dopóki jest coś do czytania w tym pliku źródłowym. Ustawiana jest zmienna i na wartość 0 i program wchodzi w kolejną pętlę while z warunkiem `i<lproc`, gdzie `lproc` jest liczbą procesów-czytelników. W tej pętli program opuszcza semafor pierwszy aż do wartości 0, a następnie wpisuje do pamięci dzielonej przeczytaną linijkę z pliku. W tym czasie czytelnicy czekają. Następnie w kolejnej pętli while pisarz zaczyna podnosić semafor drugi. Gdy semafor zostanie podniesiony, to jeden z czytelników natychmiast go opuszcza uzyskując dostęp do pamięci współdzielonej i czyta to co tam się znajduje i podnosi semafor pierwszy. W tym samym czasie pisarz wykonuje kolejną iterację pętli i znów podnosi semafor drugi i znów któryś z czytelników uzyskuje dostęp do pamięci współdzielonej.

To rozwiązanie jest dobre dla małej liczby procesów-czytelników, tak jak podano w zadaniu – dwóch. Gdyby była większa liczba czytelników mógłby pojawić się problem z tym, że czytelnik, który już przeczytał linijkę z pamięci „wepchnie się” przed innego czytelnika i tak naprawdę pierwszy czytelnik wyświetli na ekran dwa (lub więcej razy) tą samą linijkę, a inny czytelnik nigdy nie przeczyta tej linijki. Problem można rozwiązać wstawiając dodatkowe semafony, próbować synchronizować odczyt poprzez sterowanie procesami za pomocą sygnałów jednakże w przypadku dwóch, trzech, czy czterech czytelników ten problem nie występuje. Dla pewności można dodać funkcję `usleep` na koniec każdej pętli while w czytelniku, jednakże w czasie testowania działania programów nie wystąpił wyszczególniony wyżej błąd.

## Kod źródłowy programu

„matyjek\_pisarz2.c”

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/ipc.h>
#include<sys/sem.h>
#include<sys/shm.h>
#include <sys/stat.h>
#include<fcntl.h>

static struct sembuf buf;

void podnies (int semid, int semnum)
{
```

```

        buf.sem_num = semnum;
        buf.sem_op = 1;
        buf.sem_flg = 0;
        if(semop(semid, &buf, 1) == -1)
        {
            printf("blad przy podnoszeniu semafora\n");
            exit(1);
        }
    }

void opusc (int semid, int semnum)
{
    buf.sem_num = semnum;
    buf.sem_op = -1;
    buf.sem_flg = 0;
    if(semop(semid, &buf, 1) == -1)
    {
        printf("blad przy opuszczaniu semafora\n");
        exit(1);
    }
}

int main()
{
    int NR_semafora, NR_pamieci, NR_sem2;
    int semid, shmid, pid, pid2, semid2, i, lproc;
    FILE *fp1;
    char *buf_shm, tab[120];
    struct semid_ds buf3;
    struct shmid_ds buf4;

    //=====tworzenie semafora=====

    NR_semafora=ftok(".", 'G');

    semid = semget(NR_semafora, 2, IPC_CREAT|0666);
    if(semid==-1)
    {
        printf("err semafor projekt\n");
        exit(2);
    }

    lproc=semctl(semid,1,GETNCNT,0);

    if(semctl(semid, 0, SETVAL, lproc)== -1)
    {
        printf("blad nadania wartosci projekt\n");
        exit(3);
    }

    if(semctl(semid, 1, SETVAL, 0)== -1)
    {
        printf("blad nadania wartosci projekt\n");
        exit(3);
    }

    //=====koniec tworzenia semafora

```

```

//=====koniec tworzenia pamieci wspoldzielonej=====

        if((fp1=fopen("matyjek_pisarz2.c","r"))==NULL)
        {
            printf("cos nie teges\n");
            exit(1);
        }
lproc=semctl(semid,1,GETNCNT,0);
//printf("%d\n",lproc);
while((fgets(tab,120,fp1))!=NULL)
{
    i=0;
    while(i<lproc)
    {
        opusc(semid,0);
        i++;
    }
    strncpy(buf_shm, (char*)tab, 120);
    while(i)
    {
        podnies(semid,1);
        i--;
    }
    usleep(50000);

}

    while(semctl(semid,0,GETVAL,0))
    {
        opusc(semid,0);
        i++;
    }
buf_shm[0]='\0';
    while(i)
    {
        podnies(semid,1);
        i--;
    }
usleep(300000);

        semctl(semid, 0, IPC_RMID, &buf3);
        semctl(semid2, 0, IPC_RMID, &buf3);
        shmctl(shmid, IPC_RMID, &buf4);
        printf("Koncze prace\n");

return 0;
}

```

„Matyjek czytelnik2.c”

```

#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/ipc.h>
#include<sys/sem.h>
#include<sys/shm.h>
#include<sys/stat.h>
#include<fcntl.h>

static struct sembuf buf;

void podnies (int semid, int semnum)
{
    buf.sem_num = semnum;
    buf.sem_op = 1;
    buf.sem_flg = 0;
    if(semop(semid, &buf, 1) == -1)

```

```

        {
            printf("blad przy podnoszeniu semafora\n");
            exit(1);
        }
    }

void opusc (int semid, int semnum)
{
    buf.sem_num = semnum;
    buf.sem_op = -1;
    buf.sem_flg = 0;
    if(semop(semid, &buf, 1) == -1)
    {
        printf("blad przy opuszczaniu semafora\n");
        exit(1);
    }
}

int main()
{
    int NR_semafora, NR_pamieci, NR_sem2;
    int semid, shmid, pid, pid2, semid2,i;
    FILE *fpl;
    char *buf_shm, tab[120];
    struct semid_ds buf3;
    struct shmid_ds buf4;

    //=====tworzenie semafora=====

    NR_semafora=ftok(".", 'G');
    semid = semget(NR_semafora, 2, IPC_CREAT|0666);
    if(semid==-1)
    {
        printf("err semafor projekt\n");
        exit(2);
    }

    if(semctl(semid, 0, SETVAL, 1)== -1)
    {
        printf("blad nadania wartosci projekt\n");
        exit(3);
    }

    if(semctl(semid, 1, SETVAL, 0)== -1)
    {
        printf("blad nadania wartosci projekt\n");
        exit(3);
    }

    //=====koniec tworzenia semafora

    //=====tworzenie pamieci wspoldzielonej====
    NR_pamieci=ftok(".", 'P');
    shmid = shmget(NR_pamieci, 120, IPC_CREAT|0666);
    buf_shm=(char*)shmat(shmid, NULL, 0);
    //if(buf_shm[0]=='1')podnies(semid,0);
    buf_shm[0]='1';

    //=====koniec tworzenia pamieci wspoldzielonej====

```

```
while(buf_shm[0]!='\0')
{
opusc(semid,1);
strncpy(tab,buf_shm,120);
podnies(semid,0);
printf("%d   %s\n",getpid(),tab);
}
return 0;
}
```

## Wyniki uruchomienia

piotr@Piotr-Debian: ~/Dokumenty/shm2

Plik Edycja Widok Wyszukiwanie Terminal Pomoc

piotr@Piotr-Debian:~/Dokumenty/shm2\$ ./skrypt

piotr@Piotr-Debian:~/Dokumenty/shm2\$

piotr@Piotr-Debian: ~/Dokumenty/shm2

Plik Edycja Widok Wyszukiwanie Terminal Pomoc

piotr@Piotr-Debian:~/Dokumenty/shm2\$ ipcs

----- Kolejki komunikatów ---  
klucz id\_msg właściciel uprawn. bajtów komunikatów  
  
----- Segmenty pamięci dzielonej ----  
klucz id\_shm właściciel uprawn. bajtów podłączeń stan  
0x00000000 327680 piotr 600 524288 2 dest  
0x00000000 229377 piotr 600 4194304 2 dest  
0x00000000 458754 piotr 600 4194304 2 dest  
0x00000000 491523 piotr 600 524288 2 dest  
0x500118ae 524292 piotr 666 120 2  
  
----- Tablice semaforów -----  
klucz id\_sem właściciel uprawn. lsem  
0x470118ae 0 piotr 666 2  
  
piotr@Piotr-Debian:~/Dokumenty/shm2\$

piotr@Piotr-Debian: ~/Dokumenty/shm2

Plik Edycja Widok Wyszukiwanie Terminal Pomoc

piotr@Piotr-Debian:~/Dokumenty/shm2\$ ./matyjek\_czytelnik2.o

piotr@Piotr-Debian: ~/Dokumenty/shm2

Plik Edycja Widok Wyszukiwanie Terminal Pomoc

piotr@Piotr-Debian:~/Dokumenty/shm2\$ ./matyjek\_czytelnik2.o



<div>piotr@Piotr-Debian: ~/Dokumenty/shm2</div> <div> <div>Plik Edycja Widok Wyszukiwanie Terminal Pomoc</div> <pre> piotr@Piotr-Debian:~/Dokumenty/shm2\$ ./skrypt piotr@Piotr-Debian:~/Dokumenty/shm2\$ ./matyjek_pisarz2.o Koncze prace piotr@Piotr-Debian:~/Dokumenty/shm2\$ </pre> </div>	<div>piotr@Piotr-Debian: ~/Dokumenty/shm2</div> <div> <div>Plik Edycja Widok Wyszukiwanie Terminal Pomoc</div> <pre> 0x00000000 458754 piotr 600 4194304 2 dest 0x00000000 491523 piotr 600 524288 2 dest 0x500118ae 524292 piotr 666 120 2  ----- Tablice semaforów ----- klucz id_sem właściciel uprawn. lsem 0x470118ae 0 piotr 666 2  piotr@Piotr-Debian:~/Dokumenty/shm2\$ ipcs  ----- Kolejki komunikatów --- klucz id_msq właściciel uprawn. bajtów komunikatów  ----- Segmenty pamięci dzielonej ---- klucz id_shm właściciel uprawn. bajtów połączeń stan 0x00000000 327680 piotr 600 524288 2 dest 0x00000000 229377 piotr 600 4194304 2 dest 0x00000000 458754 piotr 600 4194304 2 dest 0x00000000 491523 piotr 600 524288 2 dest  ----- Tablice semaforów ----- klucz id_sem właściciel uprawn. lsem  piotr@Piotr-Debian:~/Dokumenty/shm2\$ </pre> </div>
<div>piotr@Piotr-Debian: ~/Dokumenty/shm2</div> <div> <div>Plik Edycja Widok Wyszukiwanie Terminal Pomoc</div> <pre> piotr@Piotr-Debian:~/Dokumenty/shm2\$ ./matyjek_czytelnik2.o 1890 #include&lt;stdio.h&gt;  1890 #include&lt;string.h&gt;  1890 #include&lt;stdlib.h&gt;  1890 #include&lt;unistd.h&gt;  1890 #include&lt;sys/types.h&gt;  1890 #include&lt;sys/ipc.h&gt;  1890 #include&lt;sys/sem.h&gt;  1890 #include&lt;sys/shm.h&gt;  1890 #include &lt;sys/stat.h&gt;  1890 #include&lt;fcntl.h&gt;  1890  1890 static struct sembuf buf; </pre> </div>	<div>piotr@Piotr-Debian: ~/Dokumenty/shm2</div> <div> <div>Plik Edycja Widok Wyszukiwanie Terminal Pomoc</div> <pre> piotr@Piotr-Debian:~/Dokumenty/shm2\$ ./matyjek_czytelnik2.o 1891 #include&lt;stdio.h&gt;  1891 #include&lt;string.h&gt;  1891 #include&lt;stdlib.h&gt;  1891 #include&lt;unistd.h&gt;  1891 #include&lt;sys/types.h&gt;  1891 #include&lt;sys/ipc.h&gt;  1891 #include&lt;sys/sem.h&gt;  1891 #include&lt;sys/shm.h&gt;  1891 #include &lt;sys/stat.h&gt;  1891 #include&lt;fcntl.h&gt;  1891  1891 static struct sembuf buf; </pre> </div>
<div>piotr@Piotr-Debian: ~/Dokumenty/shm2</div> <div> <div>Plik Edycja Widok Wyszukiwanie Terminal Pomoc</div> <pre> piotr@Piotr-Debian:~/Dokumenty/shm2\$ ./skrypt piotr@Piotr-Debian:~/Dokumenty/shm2\$ ./matyjek_pisarz2.o Koncze prace piotr@Piotr-Debian:~/Dokumenty/shm2\$ </pre> </div>	<div>piotr@Piotr-Debian: ~/Dokumenty/shm2</div> <div> <div>Plik Edycja Widok Wyszukiwanie Terminal Pomoc</div> <pre> 0x00000000 458754 piotr 600 4194304 2 dest 0x00000000 491523 piotr 600 524288 2 dest 0x500118ae 524292 piotr 666 120 2  ----- Tablice semaforów ----- klucz id_sem właściciel uprawn. lsem 0x470118ae 0 piotr 666 2  piotr@Piotr-Debian:~/Dokumenty/shm2\$ ipcs  ----- Kolejki komunikatów --- klucz id_msq właściciel uprawn. bajtów komunikatów  ----- Segmenty pamięci dzielonej ---- klucz id_shm właściciel uprawn. bajtów połączeń stan 0x00000000 327680 piotr 600 524288 2 dest 0x00000000 229377 piotr 600 4194304 2 dest 0x00000000 458754 piotr 600 4194304 2 dest 0x00000000 491523 piotr 600 524288 2 dest  ----- Tablice semaforów ----- klucz id_sem właściciel uprawn. lsem  piotr@Piotr-Debian:~/Dokumenty/shm2\$ </pre> </div>
<div>piotr@Piotr-Debian: ~/Dokumenty/shm2</div> <div> <div>Plik Edycja Widok Wyszukiwanie Terminal Pomoc</div> <pre> 1890 static struct sembuf buf;  1890  1890 void podnies (int semid, int semnum)  1890 {  1890     buf.sem_num = semnum;  1890     buf.sem_op = 1;  1890     buf.sem_flg = 0;  1890     if(semop(semid, &amp;buf, 1) == -1)  1890     {  1890         printf("blad przy podnoszeniu semafora\n");  1890         exit(1);  1890     } </pre> </div>	<div>piotr@Piotr-Debian: ~/Dokumenty/shm2</div> <div> <div>Plik Edycja Widok Wyszukiwanie Terminal Pomoc</div> <pre> 1891 static struct sembuf buf;  1891  1891 void podnies (int semid, int semnum)  1891 {  1891     buf.sem_num = semnum;  1891     buf.sem_op = 1;  1891     buf.sem_flg = 0;  1891     if(semop(semid, &amp;buf, 1) == -1)  1891     {  1891         printf("blad przy podnoszeniu semafora\n");  1891         exit(1);  1891     } </pre> </div>

Poniżej powiększyłem okna konsoli czytelników, aby nie robić za dużo zrzutów ekranu







```
1890      shmctl(shmid, IPC_RMID, &buf4);
1890      printf("Koncze prace\n");
1890      return 0;
1890  }
1890
1890
piotr@Piotr-Debian:~/Dokumenty/shm2$
```

```
1891      shmctl(shmid, IPC_RMID, &buf4);
1891      printf("Koncze prace\n");
1891      return 0;
1891  }
1891
1891
piotr@Piotr-Debian:~/Dokumenty/shm2$
```

## PODSUMOWANIE

Program został napisany w języku C i uruchomiony w środowisku systemu Linux Debian ver. 9. Programy wykonują się poprawnie, jak widać na zamieszczonych powyżej przykładach wywołania programów. Program został oddany do sprawdzenia na zajęciach laboratoryjnych i wtedy też został przesłany kod źródłowy programu na wskazany e-mail.