CECS 326

Spring 2017

Haney Williams

Project 4

Phanna Chrin

Amanda Pan

**Given Problem:**

Write a program using C/C++ to do the following:

Create NS semaphores in a semaphore set. The values of the semaphores are provided via the command line arguments. Provide error checking to make sure the number of semaphore values is matched to the number of semaphores NS. semctl ropt NS value1 value2 . . . values where:

ropt: option to remove the semaphores: r for remove, n for not remove.

NS: number of semaphores

value1, value2, . . ., valueNS: values of the semaphores in the semaphore set.

The program should display the semaphore identifier, time it is created, and the list of the semaphores with their value. If not remove option is selected, the program should display error if the semaphores set has already been created.

**Problem Analysis:**

This project is to understand the functionality of semaphores and its structures. The program takes in arguments of the *semtl* command, *ropt* command, the number of semaphores to be created, and the number of values assigned to each semaphore. The program then generates a unique id via *ftok()* for the semaphore identifier, and creates an array for the number of semaphores generated. After that, each semaphore is assigned to a value entered by the user. The program sets the arguements (from the union structure) to the address of the storage location for the returned semid\_ds value and then set the arguments to the address of the initializing vector. It will print out the identifier, the time created, and the semaphores with their own values.

If the ropt command is r(remove), the program removes the semaphores.

**Program Flowchart:**

User Inputs arguments

Create sem\_array with the size of NS, and

Create semaphore.

Convert argument to unsigned short and store it into the sem\_array

Print identifier

Set the argment (the union) to the address of the storage location for the returned semid\_ds value

Return the current values of the semid\_ds for the indicated semaphore identifier

If ropt is r(remove), remove semaphores

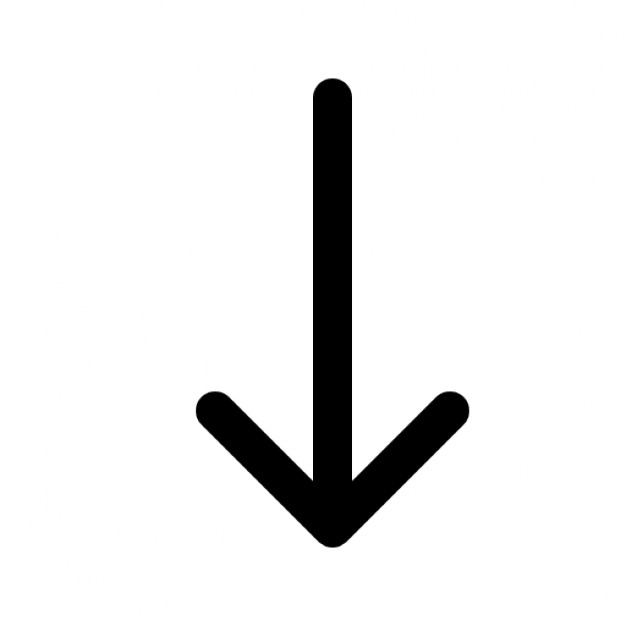
Print semaphore, and

Its value

Set values to each semaphore

Set the arguments (the union) to the address of the initializing vector

Print time created



**C Code:**

#include <stdio.h>

#include <sys/types.h>

#include <sys/ipc.h>

#include <sys/sem.h>

#include <unistd.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

//union has shared memory address

//struct doesn't

union semun

{

int val;

struct semid\_ds \*buf; // pointer to struct

ushort \*array; //pointer to array

};

int main(int argc, char \*argv[])

{

        int NS;

        int sem\_id, sem\_value;

        key\_t ipc\_key;

        struct semid\_ds sem\_buf;

        union semun arg;

        ipc\_key = ftok(".",'S');

        //Input Validation

        if (argc < 4)

        {

                printf("Must have at least 4 arguments\n");

                exit(1);

        }

        NS = atoi(argv[2]);

        if(strcmp(argv[1],"n") != 0 && strcmp(argv[1],"r") != 0)

        {

                printf("The first argument must be either 'n' or 'r'.\n");

                exit(1);

        }

        if(NS < 1)

      {

              printf("The second argument must be a positive integer.\n");

                exit(1);

        }

        if(argc - 3 != NS)

        {

              printf("Arguments not matched. Number of semaphores is %d.\n", NS);

                exit(1);

        }

        //create sem\_array with the size of NS

        ushort sem\_array[NS];

        //loop to save values into the semaphore array and make sure the values are positive integers

        for (int i = 3;i < argc; i++)

        {

                int k;

                sscanf(argv[i],"%i", &k);

                if (k < 0)

                {

                        printf("Arguments need to be positive integers\n");

                        exit(1);

                }

        }

        //create the semaphores

        if ((sem\_id = semget(ipc\_key, NS, IPC\_CREAT | 0666| IPC\_EXCL)) == -1)

{

        perror ("semget: IPC | 0666");

exit(1);

}

        //convert argument to unsigned short and store it into the sem\_array.

        for (int i = 0; i < NS; i++)

{

sem\_array[i] = (unsigned short)strtoul(argv[i+3], NULL, 0);

        }

        //print out the semaphore identifier number.

        printf("Semaphore identifier %d\n", sem\_id);

        //Set arg (the union) to the address of the storage location for

        //returned semid\_ds value

        arg.buf = &sem\_buf; //make the pointer in union semun point the address of sem\_buf

        //return the current values of the semid\_ds for the indicated semaphore identifier.

if (semctl(sem\_id, 0, IPC\_STAT, arg) == -1)

{

        perror ("semctl: IPC\_STAT");

        exit(2);

}

        //print out the time the semaphore set was created.

        printf ("Create %s", ctime(&sem\_buf.sem\_ctime));

        //Set arg (the union) to the address of the initializing vector

        arg.array = sem\_array;

        //set all the values for the individual semaphores.

        if (semctl(sem\_id, 0, SETALL, arg) == -1)

{

        perror("semctl: SETALL");

        exit(3);

        }

        //print out the values of the individual semaphores.

        for (int i = 0; i < NS; ++i)

{

                if ((sem\_value = semctl(sem\_id, i, GETVAL, 0)) == -1)

                {

                        perror("semctl : GETVAL");

                        exit(4);

}

printf ("Semaphore %d has value of %d\n",i, sem\_value);

        }

        //remove the semaphores if 'r' was selected.

        if(strcmp(argv[1], "r") == 0)

        {

                semctl(sem\_id, 0, IPC\_RMID, 0);

                if (semctl(sem\_id, 0, IPC\_RMID, 0) == -1)

                {

                        perror ("semctl: IPC\_RMID");

                        exit(5);

                }

        }

}

**Output:**

