ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА

КАТЕДРА СОФТУЕРНИ И ИНТЕРНЕТ ТЕХНОЛОГИИ

КУРСОВА РАБОТА

по

ПРИНЦИПИ НА ОПЕРАЦИОННИТЕ СИСТЕМИ

Изготвил:

/Георги Паскуали  
 фак. № /

## Задание

### Вариант 31

Да се състави програма на С/С++, която създава три процеса (А, В, С). А и Б комуникират чрез обща памет. Процес а записва общо 50 цели числа в общата памет. Процес B ги прочита и ги записва във файл. Процес Ц ги прочита от файла и визуализира на екрана.

## Основна идея

Првите два процеса се синхронизират от два семафора – empty и full. Първоначално initialise.c инициализира empty = 1, full = 0, след което се изпълняват двата паралелни процеса. При empty = 1, full = 0 не можем да влезем в критичната сек ция на writefile.c, така че ще влизаме в критичната секция на fillmem.c. След което там empty = 0, full = 1 и може да влезе в критичната секция на writefile.c. Така процесите се синхронизират – първият генерирайки числата, а вторият ги записва във файл.

## Source код

### initialise.c

#include<stdio.h>

#include"shared.h"

int main(){

sem\_t full, empty;

empty = sem\_init(63662);

full = sem\_init(62662);

sem\_set(full, 0);

sem\_set(empty, 1);

printf("Init process ended");

return 0 ;

}

### fillmem.c

#include<stdio.h>

#include"shared.h"

int main(){

sem\_t full, empty;

empty = sem\_init(63662);

full = sem\_init(62662);

int \*common,i,a;

common = (int \*)getmem(61662); // Allocate shared memory

for(i=1;i<50;i++){

PS(empty);

a = \*common;

a = a + 1;

\*common = a;

VS(full);

}

printf("fill process ended");

return 0;

}

### writefile.c

#include<stdio.h>

#include"shared.h"

#include<fcntl.h>

#include<sys/stat.h>

#include<sys/types.h>

#include<unistd.h>

int main(){

sem\_t full, empty;

empty = sem\_init(63662);

full = sem\_init(62662);

int \*common,i,a;

int bytes\_written;

common = (int \*)getmem(61662); // Allocate shared memory

int fd = open("myfile.txt", O\_RDWR | O\_CREAT, 0640);

if(fd == -1){

perror("open");

return 1;

}

for(i=1;i<50;i++){

PS(full);

printf("here");

bytes\_written = write(fd, common, sizeof(int));

if (bytes\_written == -1){

perror("write");

return 1;

}

VS(empty );

}

printf("write process ended");

}

### readfile.c

#include<stdio.h>

#include<fcntl.h>

main(){

int bytes\_read, i = 0;

int nums[50];

int fd = open("myfile.txt", O\_RDONLY);

if(fd == -1){

perror("read");

return 1;

}

while(bytes\_read = read(fd, &nums[i], sizeof(int))){

if (bytes\_read == -1){

perror("read");

return 1;

}

printf("number: %i", i);

i++;

}

}

### runprog.sh

#! /bin/bash

# run - a script for initialising the process

echo "After the ened of the process run ./readfile"

./initialise

./fill &

./write &

## Изпълними команди: bash runprog.sh