Operativni sistemi Septembar 1 11.09.2018.

Napraviti u **/home/ispit1** direktorijum u skladu sa indeksom, semestrom i asistentom kod koga slusate kurs. Na primer, student koji slusa kurs u prvom semestru kod Vlade, sa indeksom 101/2015, treba da napravi folder **v1_mi15101_s1**, a student sa indeksom 12/2015 koji slusa kurs kod Ognjena u drugom semestru treba da napravi folder **o2_mi15012_s1**. Za svaki zadatak napraviti odgovarajući **.c** fajl unutar ovog foldera (1.c, 2.c … 5.c).

Ispit se radi 3h. Svaki zadatak nosi po **20%** tj. **6 poena**. *Na izlaz za greske mozete ispisivati sta god zelite. Strogo se drzite navedenih formata ispisa za standardni izlaz!*

Zabranjeno je koristiti *system()* funkciju! Kod obavezno prevoditi sa opcijom -std=c99! Ukoliko zadatak zahteva dinamicku alokaciju, koriscenje staticke povlaci 0 poena!

1. Napisati program koji kao argumente komandne linije prima 2 argumenta, broj sekundi od Epohe i broj minuta za koliko treba pomeriti prosledjeno vreme u sekundama. Vreme ispisati na standardni izlaz u formatu DD/MM/YYYY HH:MM.

Pokretanje programa:	./1 1536493619 60	./1 1536493730 12	./1
Standardni izlaz:	09/09/2018 14:47	09/09/2018 14:00	----------
Exit kod:	0	0	1

2. Napisati program koji ume da pokrene i izvrsi bilo koji drugi program. Kroz argumente komandne linije program dobija putanju do drugog programa, ili njegov naziv, i listu argumenata koju je potrebno proslediti programu koji treba pokrenuti. Nakon izvrsavanja zeljenog programa, vas program ispisuje broj linija na izlazu pokrenutog programa ili „Neuspeh“ u slucaju neuspeha (do neuspeha dolazi kada pokrenuti program ne zavrsi sa exit kodom 0).

Pokretanje programa:	./2 cat 1.txt	./2 rm -r /ne_postoji	./2
Standardni izlaz:	7	Neuspeh	----------
Exit kod:	0	1	1
Komentar:	Datoteka 1.txt ima 7 linija	Direktorijum ne postoji	Nema arg. kom. linije

3. Napisati program koji uz pomoc niti izracunava *p*-normu matrice. Matrica realnih brojeva (koristiti double) se ucitava sa standardnog ulaza. Prvo **realan broj** *p*, pa ceo broj vrsta *m*, zatim ceo broj kolona *n* i u narednih *m* redova po *n* brojeva koji predstavljaju elemente matrice. Norma se izracunava po sledecoj formuli:

$$\|A\|_p = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^p \right)^{\frac{1}{p}}$$

Svaka nit izracunava zbir stepenovanih elemenata jedne vrste (kao i stepenovanje elemenate te vrste). Niti sinhronizovati pomocu muteksa (stititi globalni zbir svih vrsta). U main funkciji nakon obrade zavrsetka svih niti ispisati *p*-normu (stepenovati globalni zbir na 1/*p* i ispisati ga).

Standardni ulaz	**Standardni izlaz**
3 4 5 1 0 0 0 0 0 2 0 0 0 0 0 3 0 0	4.6416

0 0 0 4 0	

4. Napisati program koji koristi *poll()* funkciju da istovremeno motri vise FIFO fajlova cije se putanje prosledjuju kao argumenti komandne linije. Program iz FIFO fajlova cita sadrzaj sve dok strana koja pise ne zatvori konekciju (cita dok ima podataka). Ispisati **naziv** (ne putanju, bas naziv) FIFO fajla iz kog je najvise puta procitan karakter ***'a'*** kao i broj puta koliko je taj karakter procitan (dakle izlaz je **ime_fajla broj_pojavljivanja**).

Pokretanje programa:	./4 /tmp/f0 /tmp/f1 /tmp/f2	./4	./4 /tmp/nema /tmp/1
Standardni izlaz:	f1 11	----	----------
Exit kod:	0	1	1
Komentar:	Najvise pojavljivanja 'a' (11) je u /tmp/f1	----	fajl /tmp/nema ne postoji

5. Napisati program koji kao argumente komandne linije prima ime objekta *deljene memorije.* Potrebno je ucitati strukturu (**ARRAY_MAX** je 1024):

```
typedef struct {
      sem_t dataProcessingFinished;
      int array[ARRAY_MAX];
      unsigned arrayLen;
} OsInputData;
```

Nakon ucitavanja strukture u adresni prostor programa, program ceka na signal i ukoliko mu je poslat signal SIGUSR1 menja znak svim brojevima u nizu *array*, dok ako je u pitanju signal SIGUSR2 duplira vrednost svih elemenata niza *array* (mnozi ih sa 2). Nakon promene vrednosti niza *array* potrebno je uvecati semafor *dataProcessingFinished* (smatrati da je semafor dobro inicijalizovan). Ne raditi *shm_unlink*.

Pokretanje programa:	./5 /inmem	./5	./5 /nepostoji	./5 /somemem
Niz *array* pre:	3, 1, 2, 15, 81, 29	------	----------	7, 27, 48, 258, 343
Poslat signal:	SIGUSR1	------	----------	SIGUSR2
Niz *array* posle:	-3, -1, -2, -15, -81, -29	------	----------	14, 54, 96, 516, 686
Exit kod:	0	1	1	0

POSIX niti - dodatak

Sve funkcije za rad sa POSIX nitima vraćaju pozitivnu vrednost koda greške ako je do greske doslo, a nulu inace. Zadaci koji koriste ove funkcije se moraju linkovati sa **-lpthread**. Potpisi najbitnijih funkcija slede:

```
int pthread_create(pthread_t *thread, const pthread_attr_t *attr,
                   void *(*start_routine) (void *), void *arg);
int pthread_join(pthread_t thread, void **retval);
int pthread_mutex_init(pthread_mutex_t *mutex,const
          pthread_mutexattr_t *attr);
int pthread_mutex_destroy(pthread_mutex_t *mutex);
int pthread_mutex_lock(pthread_mutex_t *mutex);
int pthread_mutex_unlock(pthread_mutex_t *mutex);
```