Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Ispitni rok: jun 2026.
. 7. 2026.

Ispit iz predmeta „Operativni sistemi 1“

Ime i prezime: _____

Broj indeksa: _____ Broj poena: _____/30

Ispit traje 90 minuta. Nije dozvoljeno korišćenje literature.

1.(3) Dva tipa korisničkog interfejsa operativnog sistema su _____

_____.

Ako se korisnički interfejs izvršava kao proces, a ne kao deo kernela, obično se naziva _____.

2.(3) Ako na spisku fajlova koje linker treba da poveže u a) *exe* b) *lib* neki *obj* fajl uvozi neki simbol koji nijedan fajl ne definiše (izvozi), da li će linker prijaviti grešku? Odg: a) _____ b) _____

3.(3) Koje od navedenih informacija sadrži deskriptor fizičkog segmenta (upisati „1“), a koje deskriptor stranice (upisati „2“) u tabelama preslikavanja kod segmentno-stranične organizacije virtuelne memorije?

Veličina segmenta: _____ Biti prava pristupa: _____ Broj okvira: _____

Veličina segmenta izražava se u _____.

4.(3) Šta je osnovni nedostatak FIFO algoritma zamene stranica kod virtuelne memorije? Kratko obrazložiti.

Odgovor:

5.(3) Šta radi sistemski POSIX sistemski poziv *waitpid* i čemu služi njegov drugi parametar koji je tipa `int*`?

Odgovor:

6.(3) Korišćenjem bibliotečnih funkcija `setjmp` i `longjmp` napisati telo funkcije `dispatch` za promenu konteksta u školskom jezgru.

Odgovor:

7.(3) Precizno objasniti značenje procesorske instrukcije tipa *test and set*, kao i način na koji se ona koristi za međusobno isključenje kritičnih sekcija kernela kod multiprocesorskih sistema.

Odgovor:

8.(3) Bajtovi sa nekog ulaznog uređaja stižu u naletima tj. paketima promenljive dužine. Potrebno je obezbediti interfejs ka ostatku sistema koji bi pristigle bajtove mogao uzimati u paketima fiksne veličine. Ukratko, ali precizno objasniti kako to treba izvesti.

Odgovor:

9.(3) Objasniti zašto je tekuća pozicija za čitanje i upis sadržaja fajla informacija svojstvena (lokalna) svakom procesu, a ne deljena za sve procese koji su otvorili isti fajl.

Odgovor:

10.(3) FAT je keširana u operativnoj memoriji (niz `fat`), a slobodni blokovi se ulančavaju na isti način kao i fajlovi, s tim da je glava te liste `freeBlkHead`; kraj liste označava se ulazom sa vrednošću 0. Implementirati funkciju kernela `getFreeBlk` koja treba da pronađe i vrati slobodan blok i izbaci ga sa spiska slobodnih.

```
typedef uint32_t BlkNo;
extern BlkNo fat[];
extern BlkNo freeBlkHead;
BlkNo getFreeBlk ();
```

Rešenje: