
Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Operativni sistemi 1 (SI2OS1, IR2OS1)
Nastavnik: prof. dr Dragan Milićev
Odsek: Softversko inženjerstvo, Računarska tehnika i informatika
Kolokvijum: Treći, jun 2019.
Datum: 19. 6. 2019.

Treći kolokvijum iz Operativnih sistema I

Kandidat: _____

Broj indeksa: _____ *E-mail:* _____

Kolokvijum traje 1,5 sat. Dozvoljeno je korišćenje literature.

Zadatak 1 _____/10 *Zadatak 3* _____/10
Zadatak 2 _____/10

Ukupno: _____/30 = _____% = _____/10

Napomena: Ukoliko u zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku, da je uokviri (da bi se lakše prepoznala prilikom ocenjivanja) i da nastavi da izgrađuje preostali deo svog odgovora na temeljima uvedene pretpostavke. Ocenjivanje unutar potpitanja je po sistemu "sve ili ništa", odnosno nema parcijalnih poena. Kod pitanja koja imaju ponuđene odgovore treba **samo zaokružiti** jedan odgovor. Na ostala pitanja odgovarati **čitko, kratko i precizno**.

1. (10 poena) Ulaz/izlaz

U nekom sistemu implementira se keš blokova za ulazne, blokovski orijentisane uređaje sa direktnim pristupom; pošto su uređaji ulazni, blokovi se mogu samo čitati. Za svaki takav uređaj sa datim identifikatorom pravi se jedan objekat klase `BlockIOCache`, inicijalizovan tim identifikatorom, koji predstavlja keš blokova sa tog uređaja. Keš čuva najviše `CACHESIZE` blokova veličine `BLKSIZE` u nizu `cache`. Svaki ulaz *i* u nizu `cacheMap` sadrži broj bloka na uređaju koji se nalazi u ulazu *i* keša. Kada učitava blokove, keš najpre redom popunjava svoje ulaze, dok ima neiskorišćenih ulaza; podatak član `numOfBlocks` govori o tome koliko je ulaza zauzeto (redom, prvih, od ulaza broj 0). Kada popuni sve ulaze, traženi blok koji nije u kešu učitava se na mesto bloka koji je najdavnije učitani (zamenjuje se taj blok učitanim blokom).

Na raspolaganju je operacija `ioRead` koja sa datog uređaja učitava blok sa zadatim brojem u bafer zadat poslednjim argumentom. Implementirati funkciju `BlockIOCache::read` koja treba da iz bloka za datim brojem `blk` niz bajtova počev od pozicije `offset` i dužine `sz` prepíše u bafer `buffer` koji je alocirao pozivalac. Pretpostaviti da su ovi argumenti ispravni i konzistentni (pozivalac je proverio njihovu ispravnost) i da se eventualne greške u operaciji `ioRead` obrađuju u njoj ili negde drugde (ignorirati ih). (Definicija klase `BlockIOCache` nije kompletna i može se dopunjavati po potrebi.)

```
typedef char byte; // Unit of memory
typedef long long BlkNo; // Device block number
void ioRead (int device, BlkNo blk, byte* buffer);

class BlockIOCache {
public:
    BlockIOCache (int device) : dev(device), numOfBlocks(0) {}
    void read (BlkNo blk, byte* buffer, size_t offset, size_t sz);

private:
    static const unsigned BLKSIZE = ...; // Block size in Bytes
    static const unsigned CACHESIZE = ...; // Cache size in blocks

    int dev;
    byte cache [CACHESIZE][BLKSIZE]; // Cache
    BlkNo cacheMap [CACHESIZE]; // Contents of the cache (block numbers)
    int numOfBlocks; // Number of used entries (blocks in the cache)
};
```

Rešenje:

2. (10 poena) Fajl sistem

U nekom binarnom fajlu zapisano je ogromno binarno stablo; stablo je potpuno balansirano, ima n nivoa i tačno $2^n - 1$ čvorova, tj. svaki čvor osim lista ima tačno dva deteta. U svakom čvoru nalazi se jedna jedinstvena vrednost tipa `int`, a stablo je sortirano (levi potomci su manji, a desni veći od svakog čvora). Stablo je zapisano u sadržaju fajla kao niz, tako da svaki element niza sadrži samo vrednost u odgovarajućem čvoru, dok su deca čvora koji odgovara elementu i implicitno određena i nalaze se u elementima $2i+1$ i $2i+2$ tog niza; koren je u elementu 0.

Korišćenjem dole datih sistemskih poziva standardnog C fajl interfejsa, implementirati funkciju

```
int binary_search (const char* filename, unsigned n, int x);
```

koja u binarnom fajlu sa datim imenom, u kome je zapisano stablo sa n nivoa u opisanom formatu, binarnom pretragom traži vrednost datu x i vraća 1 ako je pronađe, odnosno 0 ako je ne pronađe. Ignorirati sve potencijalne greške u sistemskim pozivima. Sledeće funkcije deklarirane su u zaglavlju `cstdio`:

- `std::FILE* std::fopen(const char* filename, const char* mode);`
otvara fajl sa zadatim imenom u zatom modalitetu; za čitanje je modalitet „r“, za upis „w“, a za otvaranje fajla u binarnom režimu treba dodati sufiks „b“ na modalitet;
- `int std::fclose(FILE*);`
zatvara dati fajl;
- `std::size_t fread(void* buffer, std::size_t size, std::size_t count, std::FILE* stream);`
iz datog fajla, počev od tekuće pozicije kurzora, u zadati bafer učitava najviše `count` objekata, svaki veličine `size` i pomera kurzor fajla iza pročitano sadržaja; vraća broj stvarno pročitanih objekata (manje od traženog u slučaju greške ili nailaska na kraj fajla);
- `int fseek(std::FILE* stream, long offset, int origin);`
pomera kurzor datog fajla na pomeraj (poziciju) zadatu parametrom `offset` (u bajtovima, odnosno jedinicama koje vraća operator `sizeof`), u odnosu na položaj zadat parametrom `origin`; za pomeraj u odnosu na početak fajla, ovaj parametar treba da bude jednak konstanti `SEEK_SET`.

Rešenje:

3. (10 poena) Fajl sistem

Za fajl opisan u zadatku 2. poziva se opisani potprogram za argument $n=12$ i vrednost x koja ne postoji u stablu, a koja je veća od vrednosti u svim čvorovima stabla. Veličina bloka na disku je 512B, a veličina jednog podatka tipa `int` je 32 bita. Fajl se alocira indeksiranim metodom, s tim da je indeks prvog nivoa u FCB koji se smatra učitanim u memoriju (učitava se prilikom otvaranja fajla).

a)(5) Ako je indeks u samo jednom nivou, koliko blokova sa sadržajem ukupno dohvata ovaj program?

b)(5) Ako je indeks u dva nivoa, s tim da je indeks prvog nivoa u FCB i učitani je u memoriju, a jedan indeks drugog nivoa zauzima jedan blok na disku i ima ulaze veličine 64 bita, koliko ukupno blokova dohvata (učitava) ovaj program, računajući i pristupe indeksima drugog nivoa?

Izvesti rezultat i obrazložiti postupak.

Rešenje: