Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Operativni sistemi 1
Nastavnik: prof. dr Dragan Milićev
Odsek: Računarska tehnika i informatika, Softversko inženjerstvo
Kolokvijum: Prvi, avgust 2024.
Datum: 1. 9. 2024.

Prvi kolokvijum iz Operativnih sistema 1

Kandidat: _____

Broj indeksa: _____ *E-mail:*_____

Kolokvijum traje 90 minuta. Dozvoljeno je korišćenje literature.

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

Ukupno: _____/30 = _____% = _____/15

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)

Prevedeni *obj* fajl poseduje tabelu uvezenih simbola koja je zapisana u njemu na sledeći način. Na početku zapisa ove tabele nalazi se 32-biti neoznačen ceo broj (tip `uint32_t`) koji definiše ukupan broj uvezenih simbola koji slede u nastavku. Za svaki od tih simbola najpre je zapisan sam simbol, kao niz znakova (veličina znaka je 1 bajt) završen terminalnim znakom `'\0'`. Odmah iza tog znaka zapisan je 32-bitni pomeraj (*offset*) izražen u bajtovima u odnosu na početak sadržaja celog fajla na kom se nalazi prva 32-bitna ćelija prevedenog binarnog koda sa nerazrešenom apsolutnom adresom tog simbola. U toj ćeliji se nalazi 32-bitni pomeraj sledeće takve ćelije, na isti način, itd. To znači da su ove ćelije uvezane u jednostruku listu, gde vrednost 0 zapisana u ćeliji označava kraj liste.

Potrebno je implementirati funkciju `fillIn` koja se koristi u drugom prolazu linkera i koja treba da popuni ćelije sa nerazrešenim adresama. Sadržaju prevedenog fajla pristupa se datim funkcijama za čitanje i upis; one adresiraju ćelije sadržaja fajla datim pomerajem (u bajtovima) u odnosu na početak celog fajla. Parametar funkcije `fillIn` definiše pomeraj početka tabele uvezenih simbola. Na raspolaganju su sledeće funkcije:

- `uint32_t readUint32(uint32_t offset)`, `char readChar(uint32_t offset)`: čita i vraća jedan `unit32_t` odnosno `char` sa zadate pozicije (pomeraja u odnosu na početak fajla);
- `void writeUint32(uint32_t offset, uint32_t value)`: upisuje dati `unit32_t` na zadatu poziciju sadržaja fajla;
- `uint32_t getSymbolAddr(uint32_t offset)`: u tabeli simbola linkera pronalazi simbol koji je dat nizom znakova (završenih znakom `'\0'`) koji počinje na poziciji sa zadatim pomerajem i vraća njegovu adresu ako ga pronađe, u suprotnom vraća 0;
- `error(SYM_UNDEF,offs)`: ovaj poziv beleži grešku tipa „Nedefinisan simbol“ u tekućem *obj* fajlu, za simbol koji je dat nizom znakova (završenih znakom `'\0'`) koji počinje na poziciji sa zadatim pomerajem, koju će linker prijaviti na kraju.

```
void fillIn (uint32_t importTblOffset);
```

Rešenje:

2. (10 poena)

Neki sistem primenjuje kontinualnu alokaciju memorije za procese, s tim da tu memoriju alocira uvek u celim blokovima veličine `BLK_SIZE`, kako bi ublažio eksternu fragmentaciju. Alocira se uvek segment od najmanje dva bloka, pa i ispred prvog i iza poslednjeg alociranog segmenta ili više nema slobodnih blokova, ili postoji najmanje dva slobodna bloka. Evidencija slobodnih i zauzetih segmenata memorije vodi se u celobrojnom nizu `mmap` veličine `MAX_BLK`, tako što svaki element ovog niza odgovara po jednom bloku u prostoru za alokaciju procesa, redom. Za svaki alociran ili slobodan segment, u elementima niza `mmap` koji odgovaraju njegovom prvom i poslednjem bloku upisana je veličina tog segmenta izražena u blokovima, i to kao pozitivna vrednost ako je segment zauzet, odnosno kao negativna vrednost ako je segment slobodan.

Implementirati najpre funkciju `mergeWithNextFree` koja slobodan segment koji počinje u datom bloku spaja sa segmentom iza njega, ukoliko je i taj segment slobodan, a onda i funkciju `freeSeg` koja oslobađa zauzet segment koji počinje u datom bloku, uz spajanje sa segmentom iza i ispred njega, ukoliko su slobodni.

```
void mergeWithNextFree (int blk);
void freeSeg (int blk);
```

Rešenje:

3. (10 poena)

Neki sistem sa straničnom organizacijom memorije, sa učitavanjem stranica na zahtev (*demand paging*), ali bez zamene stranica (*page replacement*), ima veličinu stranice od 4 KB i koristi tehniku kopiranja pri upisu (*copy on write*). Neki roditeljski proces ima jedan logički segment za programski kod, veličine 560 KB, jedan logički segment za statičke podatke, veličine 1076 KB, i jedan logički segment za stek veličine 1 MB. Do trenutka pokretanja procesa deteta sistemskim pozivom *fork*, ovaj roditeljski proces je pristupao svim stranicama navedenih logičkih segmenata za kod i statičke podatke, a prilikom tog poziva stek mu je veličine 276 KB. Nakon uspešnog sistemskog poziva *fork*, povratkom iz poziva potprograma, proces roditelj najpre smanjuje veličinu svog steka do minimalne veličine od 128 KB, da bi je do kraja svog izvršavanja povećavao i do maksimalnih 560 KB; veličina steka ne izlazi iz tog opsega. Tokom tog perioda svog izvršavanja, on menja statičke podatke koji ukupno zauzimaju 16 stranica. Slično, proces dete, nakon dobijanja kontrole povratkom iz sistemskog poziva *fork*, najpre smanjuje svoj stek do minimalnih 200 KB, a potom povećava do maksimalnih 640 KB; tokom tog perioda svog izvršavanja, on menja statičke podatke koji ukupno zauzimaju osam stranica, od kojih se dve poklapaju sa stranicama koje je nakon poziva *fork* menjao roditelj.
Izračunati koliko maksimalno fizičke memorije zauzimaju ova dva procesa tokom celokupnog svog izvršavanja i obrazložiti postupak. Rezultat izraziti u broju okvira i u KB. Navesti i koliko fizčikih okvira zauzimaju ukupno njihovi segementi za kod, statičke podatke i stekove.

Odgovori:

Segmenti za kod ukupno zauzimaju _____ okvira.

Segmenti za statičke podatke ukupno zauzimaju _____ okvira.

Segmenti za stekove ukupno zauzimaju _____ okvira.

Procesi ukupno zauzimaju _____ okvira, odnosno _____ KB fizičke memorije.

Obrazloženje:

Za segmente za kod: _____

Za segmente za statičke podatke:_____

_____

_____

_____

_____

Za stekove:_____

_____

_____

_____

_____