OS2019 - Domaći 1

Simulacija tastature

Cilj domaćeg zadatka je napraviti simulaciju tastature. Domaći se piše u C i assembly, i to za Linux 0.0.1. Kompletan domaći zadatak treba da podrži sledeće funkcionalnosti:

- Čitanje rasporeda karaktera po sken kodovima.
- Čitanje tabele mnemonika.
- Čitanje niza sken kodova i njihova interpretacija
    - Posebne funkcionalnosti za shift, ctrl i alt
- Prikaz odgovarajućih karaktera na ekranu.

Opis sistema

Kod svake tastature, pritisak i otpuštanje tastera se vezuje za jedinstveni broj - scan code (**sc** dalje u tekstu). Tastatura operativnom sistemu prosleđuje **sc** pri svakom pritisku ili otpuštanju tastera, operativni sistem ih tumači i prosleđuje dalje aplikacijama. U našoj simulaciji, posebni tasteri - shift, ctrl i alt će imati fiksirane **sc**. Za sve ostale tastere postoji spisak koji je naveden u datoteci scancodes.tbl koji navodi odgovarajući **sc** za sva "mala" i "velika" slova.

-	shift	ctrl	alt
down	200	201	202
up	300	301	302

Fiksirane **sc** vrednosti

Scan code 400 će u našem sistemu predstavljati kraj datoteke.

Pored toga, naš sistem će podržavati ispis proizvoljnog ASCII karaktera pri držanju alt, i ispisivanje unapred definisanih poruka pomoću prečica uz ctrl taster.

Program treba prvo da pročita dve datoteke koje opisuju konfiguraciju sistema - scancodes.tbl i ctrl.map. Nakon toga program počinje sa interaktivnim radom i korisnik unosi naziv datoteke (na primer: test1.tst) koja sadrži niz **sc** vrednosti. Ovi **sc** se tumače i odgovarajući rezultat se ispisuje na ekranu. Tastatura ima četiri režima rada:

- Bez kontrolnih tastera - ispisuju se "mala" slova pročitana iz scancodes.tbl.
- Pritisnuto shift - ispisuju se "velika" slova pročitana iz scancodes.tbl.
- Pritisnuto ctrl - unosi se mnemonik naveden u ctrl.map.
- Pritisnuto alt - unosi se ASCII kod.

Napomena: kod kontrolnog tastera alt se rezultat prikazuje tek kada se pročita **sc** za otpuštanje tog tastera (302).

Ulazni podaci

Ulazni podaci se navode preko tri odvojene datoteke:

- scancodes.tbl
- ctrl.map
- test1.tst - ime ove datoteke se unosi pri startu programa

scancodes.tbl

U našem zadatku, datoteka scancodes.tbl će da ima spisak karaktera koji treba da se vežu za **sc**, pritom, **sc** karaktera je dat implicitno - počinje od 0 i inkrementira se za 1 za svaki karakter. Spisak karaktera za "mala" i "velika" slova je odvojen znakom za novi red.

Primer datoteke:

```
abcd()1234
ABCD{}!"#$
```

Ovo bi bila datoteka koja opisuje sistem sa 10 **sc**. Karakter 'a' je opisan sa **sc** 0, karakter 'b' sa **sc** 1, i tako dalje, do karaktera '$' koji je opisan sa **sc** 9 dok je pritisnuto shift.

Maksimalan broj **sc** u sistemu je 128.

ctrl.map

Pored običnih slova treba podržati i mnemonike, ili tzv. prečice. Mnemonici su zadati u posebnoj datoteci - ctrl.map - koja je oblika:

```
3
q tekst koji se ispisuje kada se pritisne ctrl+q
d dobro jutro
Q neki treci tekst
```

Na početku datoteke se navodi broj N ($1 \leq N \leq 16$) koji predstavlja koliko ima mnemonika. Nakon toga sledi N linija gde prvi karakter predstavlja prečicu, a ostatak teksta je ono što treba ispisati kada se unese taj karakter uz pritisnuto ctrl. Tekst koji treba ispisati će biti maksimalne dužine 64 karaktera. Prečica se navodi kao ASCII karakter. Ako ASCII karaktera nema u scancodes.tbl, onda ne može da se pristupi mnemoniku.

*.tst

Kada je sistem uspešno konfigurisan, program traži od korisnika da navede ime neke .tst datoteke koja će sadržati niz **sc** vrednosti. Ove vrednosti se tumače prema prethodnoj konfiguraciji i rezultat tumačenja se ispisuje na ekran. Vrednost 400 predstavlja kraj datoteke.

Primer datoteke:

```
200
1
300
0
1
0
202
8
7
302
201
3
301
400
```

Uz konfiguracije navedene u prethodnim delovima zadatka, ispis za ovu datoteku će biti:

```
Baba dobro jutro
```

Tumačenje:

```
200         -     shift down
1           -     B
300         -     shift up
0           -     a
1           -     b
0           -     a
202         -     alt down
8           -     3
7           -     2
302         -     alt up (32 je razmak, tj. space)
201         -     ctrl down
3           -     ctrl + d je "dobro jutro"
301         -     ctrl up
400         -     kraj
```

Struktura zadatka

Kostur domaćeg zadatka se može skinuti sa github pomoću sledeće komande:

```
git clone https://github.com/RAFOperativniSistemi/OS2019/ --branch
vezbe3 vezbe3
```

Direktorijum: `vezbe3/linux-0.01/apps/domaci`

Tu se nalaze .c i .h datoteke sa komentarima koji navode gde treba implementirati koji deo funkcionalnosti zadatka, i da li treba da bude C ili assembly.

Program treba da radi po sledećem toku:

1. Čitanje konfiguracionih datoteka i popunjavanje globalnih nizova / tabela se izvršava jednom na početku programa.
2. Program pita korisnika za ime datoteke koja sadrži **sc** vrednosti.
3. Scan code vrednosti se čitaju jedna po jedna.
4. Parsiranje **sc** vrednosti se radi poptuno u assembly. C funkcija koja radi parsiranje ima potpis:
   `int process_scancode(int scancode, char* buffer)`
   a. Prvi argument predstavlja scancode koji se parsira.
   b. Drugi argument je bafer u koji treba smestiti rezultat parsiranja (tipično samo jedan karakter ćemo smestiti u bafer - više karaktera se smešta samo pri unosu mnemonika).
   c. Povratna vrednost predstavlja broj karaktera koji su upisani u bafer.
   d. Dozvoljeno je deklarisati jednu int promenljivu koja će biti povratna vrednost za funkciju. Osim toga, sva logika funkcije treba da je napisana u inline assembly bloku.
   e. Dozvoljeno je otvoriti više od jednog inline assembly bloka.
5. Nakon što se isparsira jedna **sc** vrednost, dobijeni bafer se ispisuje na terminalu, i čita se naredna vrednost iz datoteke.
6. Kada se datoteka pročita u celosti, vraćamo se na korak 2.
7. Ako se kao naziv datoteke unese komanda “exit”, program se završava.

Predaja i rokovi

Zadatak se predaje putem mail-a na bmilojkovic@raf.rs ili mveniger@raf.rs. Sve .c i .h datoteke smestiti u direktorijum koji se zove: “os_d1_ime_prezime_ind”.
Npr. “os_d1_branislav_milojkovic_rn3807”.
Arhivirati ovaj direktorijum (.zip) i arhivu poslati kao attachment uz mail.

U tekstu mail-a obavezno navesti:

- Ime i prezime
- Broj indeksa
- Grupa, po zvaničnom spisku

Subject mail-a mora da bude u obliku: “[OS] D1 ime_prezime_ind”.
Npr. “[OS] D1 branislav_milojkovic_rn3807”

Naziv arhive mora da bude u obliku: “os_d1_ime_prezime_ind.zip”
Npr. “os_d1_branislav_milojkovic_rn3807.zip”

Rok za predaju je:

- Ponedeljak, 25. mart 23:59:59 za grupe koje slušaju OS ponedeljkom
- Sreda, 27. mart 23:59:59 za grupe koje slušaju OS sredom
- Četvrtak, 28. mart 23:59:59 za grupe koje slušaju OS četvrtkom
- Petak, 29. mart 23:59:59 za grupe koje slušaju OS petkom

Rok je definisan po grupi kojoj student zvanično pripada. Za studente sa starijih godina se primenjuje najkasniji rok.

Neće se pregledati zadaci (tj. biće dodeljeno 0 poena) ako se desi bilo koje od:

- Sadržaj mail-a nije po navedenom obliku.
- Subject mail-a nije po navedenom obliku.fa
- Naziv arhive nije po navedenom obliku.
- Predaja se desi nakon navedenog roka.

Odbrana domaćih zadataka je obavezna. Termin za odbranu prvog domaćeg zadatka će biti subota 06. april. Odbrane će se vršiti po grupama. Grupe će biti formirane, objavljene u subotu 30. marta. Ako ste iz bilo kog razloga sprečeni da prisustvujete odbrani, obavezno to najavite što pre, kako bismo mogli da zakažemo vanredni termin za odbranu.

Svrha odbrane je da se pokaže autentičnost zadatka. Ovo podrazumeva odgovaranje na pitanja u vezi načina izrade zadatka, ili izvršavanje neke izmene nad zadatkom na licu mesta. U slučaju da odbrana nije uspešna, dodeljuje se -5 poena na domaćem zadatku.

Bodovanje

Zadatak se boduje na sledeći način:

- Parsiranje i ispis velikih i malih slova - 5 poena
- Parsiranje i ispis ASCII koda - 5 poena
- Parsiranje i ispis mnemonika - 5 poena