Projekt B, prvi dio

Pretpostavimo da igrate igru riječi poput “vješala” ili “kola sreće” gdje pogađate ili cijeli skriveni izraz ili neotkrivena slova izraza na engleskom jeziku. U nekom trenutku bi Vam dobro došao program koji bi na temelju otkrivenih slova izlistao sve moguće riječi koje odgovaraju otkrivenom uzorku. Na temelju izlistanih riječi biste pronašli slova koja bi bila potencijalno najbolja za otkrivanje skrivenih izraza.

Za ovaj modul su unaprijed zadane ulazne datoteke te dio izvornog koda, uključujući modul s main funkcijom. Datoteke koje su zadane zabranjeno je mijenjati (osim `algoritam.h`), a sav dodatni kod zapisuje se u datoteke opisane u nastavku.

Priložena binarna datoteka words.dat sadrži sortirani popis riječi. Stavke u datoteci su jednake duljine i definirane strukturom:

```
struct element {
	char word[51];
};
```

Dakle, nakon pokretanja programa, traži se unos “skrivenog” teksta, gdje podvlake (_) u unesenom tekstu označavaju neotkrivena slova. Vaš kod bi morao napraviti sljedeće:

1. za svaki ulazni uzorak, pretražiti datoteku words.dat na sljedeći način:
   a. Ako je poznat prvi znak, onda koristiti binarno pretraživanje za pronalazak početka niza riječi koje ga sadrže, a potom svaku riječ u nizu obići slijedno i pronaći sve riječi koje zadovoljavaju ostale otkrivene znakove
   b. Ako je prvi znak nepoznat, slijedno se pretražuje cijela datoteka za pronalazak riječi koje zadovoljavaju otkrivene znakove
2. ispisati sve riječi koje zadovoljavaju otkrivene znakove u datoteku output1.txt, pri čemu nijedan neotkriveni znak ne smije biti jednak nekom od otkrivenih. Rezultati za različite ulazne uzorke moraju biti odvojeni linijom koja sadrži sljedeći niz znakova “**********”

Tehnički detalji

Program se sastoji od tri modula:

- `glavni.c` - već je zadan te ga je potrebno proučiti.
- `nadji.c` - mora sadržavati implementaciju funkcija zadanih zaglavnom datotekom nadji.h,
- `algoritam.c` - slično mora sadržavati implementaciju funkcija iz algoritam.h.

i popratnih zaglavnih datoteka

- `nadji.h` i
- `algoritam.h`,
- `osnovni.h`.

U nadji.c potrebno je implementirati funkciju nadji za slijednu pretragu bloka podataka prototipa:

```
long nadji(char* uzorak, long pocetak_bloka, void* izvor);
```

koja kao argument prima:

1. trenutni tekst (uzorak) koji sadrži otkrivene i skrivene znakove,

2. adresu početka bloka u datoteci (odnosno adresu vodećeg zapisa bloka kako je vraća funkcija ftell),
3. pokazivač na izvor podataka. Pokazivač tipa void se mora unutar funkcije eksplicitno pretvoriti u tip pokazivača kojim funkcija manipulira. U slučaju 1. dijela zadatka tip je FILE* i riječ je o pokazivaču na tok datotečnih podataka.

Funkcija vraća adresu prvog zapisa koji odgovara uzorku ili -1 ako takvog zapisa nema u datoteci.

U algoritam.c potrebno je implementirati funkcije nadji_blok, nadji_blok_r i inicijaliziraj zadanih prototipa:

```
long nadji_blok_r(char znak, FILE *ulaz, long lijevo, long desno);
long nadji_blok(char znak, FILE* ulaz);
void inicijaliziraj (FILE *ulaz);
```

Funkcija nadji_blok_r prima pokazivač na ulazni tok podataka (datoteka words.dat) u kojem pronalazi adresu početne stavke bloka kojem pripada stavka s traženim prvim znakom. Funkcija implementira binarno pretraživanje datoteke rekurzivnim pozivanjem same sebe. Stoga su zadani argumenti lijevo i desno kojima se pri rekurzivnim pozivima prenose adrese lijevog i desnog graničnog bloka unutar kojih se nalazi traženi blok stavki.

Funkcija nadji_blok je omatajuća (wrapper) funkcija za nadji_blok_r: to znači da služi samo kako bi pozvala funkciju nadji_blok_r s ispravnim argumentima i proslijedila povratnu vrijednost funkcije nadji_blok_r. Dakle nadji_blok prima kao argumente ulazni tok podataka i traženi znak i vraća adresu početne stavke bloka kojem pripada stavka s traženim imenom, a implementirana je tako da poziva funkciju nadji_blok_r s adresom prvog i adresom posljednjeg bloka kao argumentima lijevo i desno. Razlog ovakve implementacije bit će jasniji nakon zadavanja drugog dijela projekta.

inicijaliziraj je funkcija koju main poziva prije početka pretraživanja i u kojoj se odrađuju pripremne radnje za pretragu: određivanje veličine bloka za pretragu na temelju veličine datoteke (odnosno broja stavki u datoteci) i veličine pojedine stavke.

Dodatno, main funkcija za ispis rezultata otvara izlazni tok podataka čije je ime zadano simboličkom konstantom DATOTEKA_IZLAZ te je u modulu algoritam.h potrebno definirati vrijednost te konstante (odnosno ime izlazne datoteke programa). Ime izlazne datoteke treba biti output1.rez.

Nakon što su implementirane sve tražene funkcije potrebno je pokrenuti program i provjeriti da su dobivene vrijednosti u očekivanom rasponu, a studenti se ohrabruju da eksperimentiranjem reimplementacijom funkcije inicijaliziraj provjere kako se program ponaša s drugačijim vrijednostima veličine bloka za pretragu.

Projekt B - 2.dio

Pretraživanje u prvom dijelu projekta je brže kada je prvi znak riječi otkriven zbog sortiranosti podataka po prvom znaku u datoteci i mogućnosti korištenja binarnog pretraživanja.

U drugom dijelu projekta potrebno je implementirati alternativni algoritam pretraživanja datoteke, korištenjem indeksa na blokove podataka za prvi otkriveni znak riječi što će omogućiti ubrzanje pretrage. Program će se sastojati od tri modula:

- glavni2.c - već je zadan te ga je potrebno proučiti.
- nadji-alt.c - mora sadržavati implementaciju funkcija zadanih zaglavnom datotekom nadji.h prilagođenih 2. dijelu zadatka
- funkcionalnost izgradnje i korištenja indeksa za pretragu bit će implementirana u novom modulu algoritam_alt.c.

Indeks je implementiran dvostruko povezanom listom čiji su elementi zadani strukturom:

```
struct at {
char word[51];
        long pocetak_bloka;
        struct at *sljed, *preth;
};
```

Svaki atom liste osim pokazivača na prethodni i sljedeći atom (od kojih je jedan NULL u prvom odnosno zadnjem atomu) sadrži adresu bloka u datoteci (odnosno adresu vodećeg zapisa u bloku). Na taj način svaki atom liste „pokazuje“ na jedan blok u datoteci. Atom također sadrži i ime vodećeg zapisa u bloku kako pri pretrazi podataka ne bi bilo potrebno za usporedbe dohvaćati ime iz datoteke što je najsporija operacija pri pretrazi (pristup disku je za više redova veličine sporiji od rada sa podacima u memoriji).

Također, koristi se polje pokazivača:

```
 struct at * pocetci [26];
```

gdje je n-ti element polja pokazivač na prvu riječ u listi s n-tim slovom abecede na zadanom mjestu.

Potrebno je napisati novu implementaciju funkcija inicijaliziraj i nadji_blok istih prototipa kao i u prvom dijelu projekta. U funkciji inicijaliziraj potrebno je kreirati listu čitanjem datoteke blok po blok i dodavanjem atoma u listu, sortirano uzlazno po znaku na prvoj otkrivenoj poziciji tražene riječi (dakle, kombinacija poziva strcmp i umetanja u listu na pravu poziciju, u nedostatku poznavanja sofisticiranijih algoritama za sortiranje), te popunjavanjem polja pocetci. Funkcija inicijaliziraj glavu i rep liste pohranjuje u globalnu varijablu indeks koju je potrebno definirati u modulu algoritam_alt.c kao:

```
struct {
        struct at *glava, *rep;
} indeks;
```

Funkcija nadji_blok treba biti implementirana operacijama na listi bez rada s datotekom, a listi pristupa korištenjem globalne varijable indeks. Funkcija zanemaruje ulazni tok podataka kojeg prima kao argument (smije se pozvati i s NULL pokazivačem).

U algoritam_alt.c je ponovno potrebno definirati vrijednost simboličke konstante DATOTEKA_IZLAZ, ali ovaj puta kao znakovni niz output2.rez.

Pri prezentaciji projekta očekuje se da student demonstrira rad obje verzije pretrage. Zato je potrebno jednom provesti prevođenje i povezivanje korištenjem datoteke algoritam.c iz prvog dijela projekta, a drugi puta prevođenje i povezivanje korištenjem datoteke algoritam_alt.c iz drugog dijela projekta. Rezultat izvođenja trebaju biti dvije jednako formatirane datoteke (output1.rez i output2.rez) s listama riječi koje zadovoljavaju zadani uzorak.