Ovaj dokument nije službeni. Namjera mu je dati uvid u zadatke koji bi se MOGLI naći na međuispitu. Moguće je da će se u međuispitu javiti i lakši i teži zadaci od navedenih.

Raspršeno adresiranje

1. Jedan zapis datoteke organizirane po principu raspršenog adresiranja sadrži **matični broj studenta** (int), **ime i prezime** (50+1 znak) i **poštanski broj mjesta stanovanja** (int). Prazni se zapis prepoznaje po matičnom broju jednakom 0. Veličina bloka na disku je 2048. Očekuje se najviše 30000 zapisa, a tablica je dimenzionirana za 20% veći kapacitet. Kod upisa se primjenjuje metoda cikličkog preljeva. Ključ zapisa je matični broj, a transformacija ključa u adresu se obavlja zadanom funkcijom

   ```
   int adresa(int mbr);
   ```

   Napisati funkciju koja će za zadani indeks pretinca u preko argumenata vratiti koliko u pretincu ukupno ima zapisa, koliko od njih pripada tom pretincu i koliko ih je u zadani pretinac zapisano kao preljev. Funkcija preko imena (naredbom ***return***) treba vratiti 1 ako zadani pretinac postoji u datoteci, a 0 inače i treba imati prototip:

   ```
   int fun(FILE *f, int pretinac, int *uk, int *pripada, int *preljev);
   ```

2. Jedan zapis datoteke organizirane po principu raspršenog adresiranja sadrži **matični broj studenta** (int), **ime i prezime** (50+1 znak) i **datum rođenja** (8+1 znak formata *ddmmgggg*). Prazni se zapis prepoznaje po matičnom broju jednakom 0. Veličina bloka na disku je 1024. Očekuje se najviše 15000 zapisa, a tablica je dimenzionirana za 35% veći kapacitet. Kod upisa se primjenjuje metoda cikličkog preljeva. Ključ zapisa je matični broj, a transformacija ključa u adresu se obavlja zadanom funkcijom

   ```
   int adresa(int mbr);
   ```

   Napisati funkciju koja će zapisati novi zapis u datoteku. Funkcija treba imati prototip:

   ```
   int upisi(zapis novi, FILE *f);
   ```

   i treba vratiti 1 ako je zapis upisan u svoj pretinac, -1 ako je zapis upisan kao preljev, a 0 ako zapis uopće nije zapisan u datoteku (nije bilo mjesta).

3. Jedan zapis datoteke organizirane po principu raspršenog adresiranja sadrži **matični broj studenta** (int), **ime i prezime** (50+1 znak) i **datum rođenja** (8+1 znak formata *ddmmgggg*). Prazni se zapis prepoznaje po matičnom broju 0. Veličina bloka na disku je 1024. Očekuje se najviše 40000 zapisa, a tablica je dimenzionirana za 25% veći kapacitet. Kod upisa se primjenjuje metoda cikličkog preljeva. Ključ zapisa je matični broj, a transformacija ključa u adresu obavlja se gledanjem ostatka prilikom dijeljenja s brojem pretinaca.

   Napisati funkciju koja će u datoteci pronaći zapis o studentu sa zadanim matičnim brojem. Funkcija treba vratiti 0 ako se zapis ne nalazi u datoteci, 1 ako je zapis zapisan u svoj pretinac, a -1 ako je zapis zapisan kao preljev. Funkcija treba imati prototip:

   ```
   int fun(FILE *f, int mbr, zapis *stud);
   ```

4. Jedan zapis datoteke organizirane po principu raspršenog adresiranja sadrži **matični broj studenta** (int), **ime i prezime** (50+1 znak) i **datum rođenja** (8+1 znak formata *ddmmgggg*). Prazni se zapis prepoznaje po matičnom broju jednakom -1. Veličina bloka na disku je 2048. Očekuje se najviše 40000 zapisa, a tablica je dimenzionirana za 30% veći kapacitet. Kod upisa se primjenjuje metoda cikličkog preljeva. Ključ zapisa je matični broj, a transformacija ključa u adresu se obavlja zadanom funkcijom:

```
int adresa(int mbr);
```

Napisati funkciju koja će **za svaki** pretinac ispisati postotni udio zapisa koji su u njega upisani kao preljev. Funkcija treba imati prototip:

```
void postotniUdio(FILE *f);
```

5. Jedan zapis datoteke organizirane po principu raspršenog adresiranja sadrži **matični broj studenta** (int), **ime i prezime** (50+1 znak) i **datum rođenja** (8+1 znak formata *ddmmgggg*). Prazni se zapis prepoznaje po matičnom broju 0. Veličina bloka na disku je 512. Očekuje se najviše 20000 zapisa, a tablica je dimenzionirana za 30% veći kapacitet. Kod upisa se primjenjuje metoda cikličkog preljeva. Ključ zapisa je matični broj, a transformacija ključa u adresu obavlja se zadanom funkcijom:

```
int adresa(int mbr);
```

Napisati funkciju koja će prebrojati koliko je zapisa u datoteci zapisano kao preljev. Funkcija treba imati prototip:

```
int fun(FILE *f);
```

6. Jedan zapis datoteke organizirane po principu raspršenog adresiranja sadrži **matični broj studenta** (int), **ime i prezime** (50+1 znak) i **datum rođenja** (8+1 znak formata *ddmmgggg*). Prazni se zapis prepoznaje po matičnom broju jednakom 0. Veličina bloka na disku je 1024. Očekuje se najviše 40000 zapisa, a tablica je dimenzionirana za 25% veći kapacitet. Kod upisa se primjenjuje metoda cikličkog preljeva. Ključ zapisa je matični broj, a transformacija ključa u adresu se obavlja zadanom funkcijom

```
int adresa(int mbr);
```

Napisati funkciju koja će vratiti indeks (redni broj) pretinca koji je najmanje iskorišten (u kojem ima najviše mjesta). Funkcija treba imati prototip:

```
int popunjeni(FILE *f);
```

7. **5.** Jedan zapis datoteke organizirane po principu raspršenog adresiranja sadrži **matični broj studenta** (int), **ime i prezime** (50+1 znak) i **datum rođenja** (8+1 znak formata *ddmmgggg*). Prazni se zapis prepoznaje po matičnom broju jednakom 0. Veličina bloka na disku je 1024. Očekuje se najviše 40000 zapisa, a tablica je dimenzionirana za 25% veći kapacitet. Kod upisa se primjenjuje metoda cikličkog preljeva. Ključ zapisa je matični broj, a transformacija ključa u adresu se obavlja zadanom funkcijom

```
int adresa(int mbr);
```

Napisati funkciju koja će vratiti broj do kraja popunjenih pretinaca. Funkcija treba imati prototip:

```
int popunjeni(FILE *f);
```

8. **5.** Jedan zapis datoteke organizirane po principu raspršenog adresiranja sadrži **matični broj studenta** (int), **ime i prezime** (50+1 znak) i **datum rođenja** (8+1 znak formata *ddmmgggg*). Prazni se zapis prepoznaje po matičnom broju jednakom 0. Veličina bloka na disku je 1024. Očekuje se najviše 40000 zapisa, a tablica je dimenzionirana za 25% veći kapacitet. Kod upisa se primjenjuje metoda cikličkog preljeva. Ključ zapisa je matični broj, a transformacija ključa u adresu se obavlja zadanom funkcijom

```
int adresa(int mbr);
```

Napisati funkciju koja će vratiti broj do kraja popunjenih pretinaca. Funkcija treba imati prototip:

```
int popunjeni(FILE *f);
```

Rekurzija

9. Napisati **rekurzivnu** funkciju koja će zadani niz znakova ispisati obrnutim redoslijedom. Npr. za niz ''Abeceda'', funkcija treba ispisati ''adecebA''. Funkcija mora imati prototip:

```
void ispisi_obrnuto(char *niz);
```

10. Napisati **rekurzivnu** funkciju koja će pronaći indeks najvećeg elementa u zadanom cjelobrojnom polju. Napisati drugu **rekurzivnu** funkciju koja će, koristeći prvu funkciju, urediti zadano polje po veličini. Druga funkcija treba raditi slijedeće:

- pronaći najveći element (koristeći prvu funkciju)
- zamijeniti ga s elementom na početku polja
- pozvati se rekurzivno za ostatak polja

Odrediti apriornu složenost prve i druge funkcije.

11. Napisati rekurzivnu funkciju koja će izračunati istu sumu reda kao i zadana funkcija `f`:

```
double f(int n) {
   double suma = 0;
   int i;
   for (i = 1; i < n; i++)
      suma += 1./(i * (i + 1) * (i + 2));
   return suma;
}
```

12. Niz brojeva definiran je rekurzivno na slijedeći način:

$f_0 = 1$,

$f_1 = 2$,

$f_n = (f_{n-1} + 1) * f_{n-2}$, za $n > 1$.

Napisati **rekurzivnu** funkciju koja će izračunati n-ti član niza. Odrediti apriornu složenost funkcije. Funkcija mora imati prototip:

```
long f(int n);
```

13. Napisati rekurzivnu funkciju za izračunavanje sume geometrijskog reda od n članova, $\sum_{j=0}^{n} aq^j$, ako je prototip funkcije:

```
float SumaGeometrijskogReda (float a, float q, int n);
```

Kolika je apriorna složenost te funkcije?

Malloc/realloc

14. Napisati funkciju koja će preko argumenata vratiti razliku dva skupa cijelih brojeva (sve elemente koji se pojavljuju u prvom skupu, ali se ne pojavljuju u drugom skupu). Unutar funkcije treba zauzeti memorijski prostor u koji će se zapisati razlika. Može se pretpostaviti da zadani skupovi nemaju duplih elemenata. Funkcija treba imati prototip:

```
int presjek(int p1[], int n1, int p2[], int n2, int **p3);
```

i preko imena treba vratiti broj članova razlike. Prilikom rješavanja ovog zadatka nije dopušteno korištenje funkcije *realloc* (koristiti isključivo *malloc*). Zadani skupovi nisu sortirani. Odrediti apriornu složenost funkcije.

15. Napisati funkciju koja za dane parametre *n*, a_0 i *d*, vraća novo polje od *n* elemenata (kreirano sa *malloc*) čije su vrijednosti članovi aritmetičkog niza ($a_i = a_{i-1} + d$). Prototip funkcije je:

```
int *KreirajAritNiz(int n, int a0, int d);
```

16. Napišite funkciju koja će preko parametara uzimati cjelobrojno polje A te donju i gornju granicu intervala te stvoriti novo cjelobrojno polje na način da ono sadrži sve brojeve iz polja A čija je vrijednost unutar intervala [a, b] (a i b uključeno). Ukoliko interval nije valjano zadan (npr a=10, b=5) funkcija vraća NULL preko `return` i ne rezervira nikakvu memoriju.

Npr za polje A = 1,22,-33,445, 900 i za a = 22, b = 899 mora se stvoriti novo polje C = 22, 445 (redoslijed elemenata u polju C nije bitan, a duplikate, ukoliko postoje, treba očuvati).

Novonastalo polje vratiti preko imena funkcije (naredbom return), a broj elemenata u polju preko parametra `int *Nc`.

```
int* napraviPolje(int *A, int Na, int a, int b, int* Nc)
```

17. Napisati funkciju čiji je prototip

```
char *brojeve_u_znakove(int *polje, int duljina)
```

koja će stvoriti niz znakova koji će sadržavati brojeve iz ulaznog polja napisane u obrnutom redoslijedu i to bez međusobnih razmaka te bez vodećih i pratećih praznina. Npr. ukoliko je ulazni polje sadržavalo brojeve 123, 15, 200, 84, 1021 i 7 funkcija će stvoriti novi niz koji će imati sadržaj "321510024812017". Funkcija mora vratiti pokazivač na novostvoreni niz znakova. Ukoliko je ulazni niz bio duljine 0, funkcija mora vratiti prazan niz. Možete pretpostaviti da su svi brojevi u polju pozitivni.

Datoteke

18. Neka je zadana slijedna formatirana datoteka u kojoj se nalaze podaci o albumima:

```
naziv       (char[30+1])
autor       (char[20+1])
prodano     (int)
```

(Podaci o jednom albumu zapisani su u jednom retku, odvojeni jednim razmakom. Pretpostaviti da `naziv` i `autor` ne sadrže praznine.)

Napisati funkciju koja će vratiti broj albuma za koje je prodano `granica` ili više primjeraka.

Prototip funkcije mora biti:

```
int veci_od(FILE *f, int granica);
```

19. Neka je zadana direktna neformatirana datoteka u kojoj se nalaze zapisi oblika:

```
struct natjecatelj{
      int redni_br;
      char prezime[20+1];
      int bodovi;
};
```

Zapisi su u datoteci sortirani po rednom broju (`redni_br`) i to tako da redni broj zapisa odgovara rednom broju natjecatelja. (Datoteka počinje rednim brojem 1. Ako je `redni_br = 0`, znači da je zapis prazan.)

Napisati funkciju koja vraća broj natjecatelja koji imaju više ili jednako `n` bodova. Paziti na prazne zapise!

Funkcija mora imati prototip:

```
int veci_od(FILE *f, int n);
```

20. Neka je dana formatirana datoteka `ispiti.txt` u kojoj se u svakom retku nalaze podaci o jednom izlasku na ispit i to u formatu:

```
Naziv predmeta#MBR#Ime i prezime#ocjena
```

`MBR` je osmeroznamenkasti prirodni broj, a `ocjena` jednoznamenkasti prirodni broj.

`Naziv predmeta` nije dulji od 100 znakova i može uključivati praznine. `Ime i prezime` nisu dulji od 50 znakova i mogu uključivati praznine.

Napisati program koji će u formatiranu datoteku `polozeni_ispiti.txt` prepisati samo one zapise kojima ocjena nije 1.

21. Neka je zadana formatirana datoteka `popis.txt` u kojoj se u svakom retku nalaze ime (najviše 40 znakova), prezime (najviše 40 znakova), spol (M ili Z) i broj godina, gdje su podaci međusobno odvojeni znakom $.

Primjer: `Ana Ivana$Ančić-Ivančić$Z$54`

Napisati program koji će otvoriti datoteku i zatim ispisati koliko ima ženskih osoba u sljedećim dobnim skupinama: do 18, 19-35, 36-59, 60 ili više.

22. Na disku se nalazi slijedna formatirana datoteka kino.txt, gdje su zapisi oblika

KKNffpppffpppKKNffpppffpppffpppKKNffppp...

gdje je:

KK	šifra kina 0 – 10
N	ukupan broj različitih filmova koji se prikazuju u kinu KK
ff	šifra filma od 0-20
ppp	broj prodanih karata 0-100 za film ff u kinu KK

Napisati program koji će ispisati najgledaniji film. Ukoliko više filmova ima najveću gledanost ispisati sve takve.

23. U slijednoj formatiranoj datoteci "ekipe.txt" nalazi se popis nogometnih ekipa na nastavničkom malonogometnom turniru na FER-u. Format datoteke je: redni_broj_ekipe#ime_ekipe\n

Primjer:

1#ZPM\n

2#ZEMRIS\n

3#ZVNE\n

4#ZESOI\n

Prvi podatak je redni broj ekipe, a drugi ime ekipe. Unaprijed se ne zna koliko ekipa ima u skupini, ali ih ne može biti više od 6.

U slijednoj formatiranoj datoteci "rezultati.txt" nalaze se rezultati odigranih utakmica. Format datoteke je: #redni_broj_ekipe1#redni_broj_ekipe2#broj_golova_ekipe1#broj_golova_ekipe2\n

Primjer:

1#2#7#2\n

3#4#3#3\n

2#3#4#7\n

4#1#2#6\n

3#1#5#5\n

2#4#3#6\n

Napisati program koji će učitati s tipkovnice naziv ekipe i ispisati koliko je prosječno golova davala po utakmici. Napomena: Možete pretpostaviti da ime ekipe ne sadrži praznine.

24. Slijedna formatirana datoteka sadrži podatke o studentima. Podaci su zapisani na slijedeći način:

dd.mm.gggg#ime prezime#JJJJJJJJJJJJ

dd.mm.gggg#ime prezime#JJJJJJJJJJJJ

gdje su:

dd.mm.gggg – dan, mjesec i godina rođenja

ime i prezime – ime i prezime studenta (najviše 50 znakova), student može imati više imena ili prezimena (npr. Ivana Brlić Mažuranić)

JJJJJJJJJJJJ – JMBG studenta (13 znakova)

Napisati funkciju koja će ispisati podatke o svim studentima koji su rođeni 29. veljače (bilo koje godine). Ako takvih studenata nema u datoteci, treba ispisati odgovarajuću poruku. Datoteku čitati koristeći **fscanf** funkciju. Funkcija mora imati prototip:

void funkcija(FILE *f);

Nama je Botički reka za hash totalno suprotno... Uglavnom da će biti 100%Naravno nece bit potrebno pisat cijeli kod hasha(sve one funkcije za spremanje zapisa, trazenje, ispis i main) već samo jedna takva.
a ta jedna moze eventualno bit:
1) neka statisticka funkcija (tipa prebroji preljeve u jednom pretincu, ILI za određeni pretinac ispisi postotak popunjenosti....itd itd)
2) jedna od funkcija **trazi** ili **upisi**....(dvi JAKO bitne funkcije, preporucan ako ih ne kuzite naucite napamet jer se doslovno moze desit da bas one budu,al isto tako nisu teske)

Za nasu grupu je stavija na ahyco neke primjere starih kolokvija za vjezbu
evo link http://rapidshare.de/files/17064408/...a-1MI.doc.html

I kao da provjezbamo ove zadatke...
Da naucimo napamet(skužimo) one definove na vrhu koda (ono M, C, N...)-kao da to nosi bodove i također definirana struktura.

Isto tako je kao vrlo bitno da se za **statisticke funkcije** koristi **for petlja**(jer trebamo napravit nesto za cijelu hash datoteku ili pak za cijeli pretinac), a za **funkcije tipa upisi ili trazi** OBAVEZNO **do while** petlja (zbog ciklickog upisa/trazenja)

Ne znam sad kolko san bila jasna.... ale eto 

E sta je jos reka....hm
jos sigurno jedna rekurzija, jedan malloc(pogledajte u ocin zadacima-ima primjera)
i jos kao zadnji zadatak mozda datoteka neka...ali to je reka da nije bas sigurno.
Zadnji sat nam je takoder natuknija da moze bit ispitivanje slozenosti. Tipa da za svaki zad odredimo jos i slozenost ili nesto tako....

Eto samo prenosim sve sta su nan rekli....

Nadan se da san pomogla nekima...