Strukture podataka i algoritmi 1
(zadatak - max 30 poena)

Avgust, 2022

U knjižari „IMI“ knjige se pakuju na police po oblastima. Za svaku oblast je poznat ID oblasti (ceo broj), naziv oblasti (niz karaktera), broj polica namenjenih za tu oblast (ceo broj) i koliko knjiga na jednu takvu policu može da stane (ceo broj). Police koje sadrže knjige iz iste oblasti su identične, ali se mogu razlikovati od polica za druge oblasti, pa stoga i broj knjiga po polici za različite oblasti može biti različit.
Za svaku knjigu u knjižari se zna naslov (niz karaktera), autor (niz karatera), ID oblasti kojoj knjiga pripada i broj primeraka u knjižari (ceo broj) i broj primeraka koji prodat tokom prošle nedelje (ceo broj). Knjige se pakuju tako da na police iz određene oblasti, moraju da stanu sve knjige koje pripadaju toj oblasti sa, ako je moguće, jednakim brojem primeraka. Ukoliko za neku knjigu nema dovoljan broj primeraka da se po broju izjednači sa drugim knjigama iz te oblasti, taj prostor može ostati nepopunjen. Primerci koji ne mogu da se smeste na policu, nalaze se u magacinu.
Deset najprodavanijih knjiga iz protekle nedelje smeštaju se na policu „Top 10“, a **ne** na policu iz oblasti. Svaku knjiga na polici „Top 10“ se nalazi u 10 primeraka ili manje u slučaju da u knjižari nema 10 primeraka knjige.
Učitati podatke o oblastima i knjigama. Rasporediti knjige po policama, uključujući i policu „Top 10“. Prikazati stanje u magacinu, ispsiujući za svaku knjigu naziv i broj primeraka u magacinu. Odrediti kojih knjiga nema u magacinu, da bi se formirao spisak za nabavku.
(3 poena main, <u>obavezno</u>)

Za rešavanje problema napisati sledeće funkcije:

a) Definisati sve potrebne <u>složene</u> tipove podataka neophodne za rešavanje opisanog problema.
(3 poena, obavezno)
b) Napisati funkciju **UcitajOblasti** koja iz datoteke *Oblasti.txt* učitava podatke o oblastima i formira listu/niz oblasti.
(3 poena, obavezno)
c) Napisati funkciju **NadjiOblast** koja za dat ID vraća <u>pokazivač</u> na odgovarajući oblast.
(2 poena)
d) Napisati funkciju **UcitajKnjige** koja iz datoteke *Knjige.txt* učitava podatke o knjigama i formira listu/niz knjiga.
(3 poena)
e) Napisati funkciju **Top10** koja određuje sadržaj pomenute police.
(4 poena)
f) Napisati funkciju **Rasporedi** koja za raspoređuje knjige po policama/oblastima i magacinu.
(5 poena)
g) Napisati funkciju **IspisMagacina** koja ispisuje sadržaj magacina.
(3 poena)
h) Napisati funkciju **Nabavka** koja ispsisuje podatke za knjige koje je potrebno nabaviti.
(4 poena)

Pri učitavanju podataka podrazumevati da su svi podaci korektno zadati.
Dozvoljeno je proširivanje struktura i definisanje novih.
Maksimalan broj poena se dobija samo u slučaju da postoji veza između struktura koje čuvaju oblasti i struktura koje čuvaju knjige, tj. ako postoje pokazivači “na jednu” ili “na obe strane”.

Zadatak se može rešavati korišćenjem povezanih lista, nizova ili kombinacijom.
Zadatak rešiti bez korišćenja globalnih promenljivih i bez unapred definisanih dužina korišćenih nizova (izuzetak može da bude pomoćni niz za učutavanje stringova)**.**

Rešenje studenta

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

#define alociraj_knjigu(x) (x = (Knjiga*)malloc(sizeof(Knjiga)))
#define alociraj_oblast(x) (x = (Oblast*)malloc(sizeof(Oblast)))

typedef struct knjiga {
	char *naslov_knjige;
	char *ime_autora;
	int ID_oblasti;
	int br_primeraka;
	int prodato_primeraka;

	struct knjiga *sledeca;
} Knjiga;

typedef struct oblast {
	int ID_oblasti;
	char *ime_oblasti;
	int br_polica; //broj polica po oblasti
	int br_knjiga; //broj knjiga koje staju na jednu policu
	int br_magacin; //broj razlicitih knjiga smestenih u magacin
	int br_razlicitih_knjiga; //broj razlicitih knjiga smestenih na police
	int prosek; //broj primeraka svake knjige koji treba da je na policama za ovu oblast

	Knjiga **magacin;
	Knjiga **knjige;

	struct oblast *sledeca;
} Oblast;

void UcitajOblasti(Oblast **p) {
	Oblast *nova;
	alociraj_oblast(nova);

	char pomocni[50];
	int duzina;

	//ucitavanje oblasti top10 kao prve u listi oblasti
	nova->sledeca=NULL;
	nova->ID_oblasti=-10;
	char top10naziv[]="Top 10 ";
	duzina=strlen(top10naziv);
	top10naziv[duzina-1]='\0';
	nova->ime_oblasti=(char*)malloc(sizeof(char)*duzina);
	strcpy(nova->ime_oblasti,top10naziv);
	nova->br_polica=1;
	nova->br_razlicitih_knjiga=10;
	nova->br_knjiga=0;
	nova->br_magacin=0;
	nova->knjige=(Knjiga**)malloc(sizeof(Knjiga));
	nova->magacin=(Knjiga**)malloc(sizeof(Knjiga));
	*p=nova;
	FILE *f;
	f=fopen("Oblasti.txt","r");

	while(!feof(f)) {
		alociraj_oblast(nova);
		nova->sledeca=NULL;

		fscanf(f,"%d", &nova->ID_oblasti);
		fgetc(f);

		fgets(pomocni, 50, f);
		duzina=strlen(pomocni);
		pomocni[duzina-1]='\0';

		nova->ime_oblasti=(char*)malloc(sizeof(char)*duzina);
```

```
                strcpy(nova->ime_oblasti,pomocni);

                fscanf(f,"%d",&nova->br_polica);
                fscanf(f,"%d", &nova->br_knjiga);

                nova->magacin=(Knjiga**)malloc(sizeof(Knjiga));
                nova->br_magacin=0;

                nova->knjige=(Knjiga**)malloc(sizeof(Knjiga));
                nova->br_razlicitih_knjiga=0;
                nova->prosek=0;

                Oblast *pom=*p;
                while(pom->sledeca)
                        pom=pom->sledeca;
                pom->sledeca=nova;

        }
        fclose(f);
}

Oblast *NadjiOblast(int ID, Oblast *p) {
        Oblast *pom=p;
        while(pom) {
                if(pom->ID_oblasti==ID)
                        return pom;
                pom=pom->sledeca;
        }
        return pom;
}

void UcitajKnjige(Knjiga **p) {
        Knjiga *nova;
        int duzina;

        FILE *f;
        f=fopen("Knjige.txt","r");

        while(!feof(f)) {
                char pomocni[50];

                alociraj_knjigu(nova);
                nova->sledeca=NULL;

                fgets(pomocni, 50, f);
                duzina=strlen(pomocni);
                pomocni[duzina-1]='\0';
                nova->naslov_knjige=(char*)malloc(sizeof(char)*duzina);
                strcpy(nova->naslov_knjige, pomocni);

                fgets(pomocni,50,f);
                duzina=strlen(pomocni);
                pomocni[duzina-1]='\0';
                nova->ime_autora=(char*)malloc(sizeof(char)*duzina);
                strcpy(nova->ime_autora,pomocni);

                fscanf(f,"%d",&nova->ID_oblasti);
                fscanf(f,"%d",&nova->br_primeraka);
                fscanf(f, "%d", &nova->prodato_primeraka);
                fgetc(f);

                if(*p == NULL)
                        *p=nova;
                else {
                        Knjiga *pom=*p;
                        while(pom->sledeca)
                                pom=pom->sledeca;
                        pom->sledeca=nova;
                }
        }
        fclose(f);
}
```

```
void DodajElement(Knjiga **p, Knjiga *nova) {
	if(*p==NULL)
		*p=nova;
	else {
		Knjiga *pom=*p;
		while(pom->sledeca)
			pom=pom->sledeca;
		pom->sledeca=nova;
	}
}

void Top10(Knjiga **p, Oblast **o) {
	Knjiga *maks, *maks_pr, *pom, *pom_pr;
	Knjiga**pp=p;
	Knjiga *q=NULL;

	//soritranje liste knjiga u opadacujem poretku, po broju prodatih primeraka od prethodne nedelje
	while(*pp) {
		maks=*pp;
		maks_pr=NULL;
		pom_pr=*pp;
		pom=pom_pr->sledeca;

		while(pom) {
			if(pom->prodato_primeraka>maks->prodato_primeraka) {
				maks=pom;
				maks_pr=pom_pr;
			}
			pom_pr=pom;
			pom=pom->sledeca;
		}
		if (maks_pr==NULL)
			*pp=maks->sledeca;
		else
			maks_pr->sledeca=maks->sledeca;
		maks->sledeca=NULL;
		DodajElement(&q,maks);
	}
	*p=q;

	//smestanje knjiga u oblast top10 i stampanje smestnih knjiga u oblast top10
	int i=1;
	Oblast *pomo=*o;
	pom=*p;
	printf("\n\nTOP 10 KNJIGA:\n\n");
	while(i<=10) {
		printf("\t%d. knjiga\n",i);
		printf("\t\tNaslov knjige: %s\n",pom->naslov_knjige);
		printf("\t\tIme knjige: %s\n", pom->ime_autora);

		pom->ID_oblasti=-10; //za knjige koje su u top10, stavlja ID top10 oblasti, umesto ID-a
oblasti kojoj bi knjiga pripadala po zanru
		pomo->knjige=(Knjiga**)realloc(pomo->knjige, sizeof(Knjiga)*i);
		pomo->knjige[i]=pom;

		//ako je knjiga oblasti top10<10, toliko ih smestamo u oblast, u suprotnom smestamo ih 10 u
oblast, a ostatak cuvamo za magacin
		if(pom->br_primeraka<10) {
			pomo->br_knjiga+=pom->br_primeraka;
			printf("\t\tOvih knjiga je %d\n",pom->br_primeraka);
			pom->br_primeraka=0;
		} else {
			pomo->br_knjiga+=10;
			printf("\t\tOvih knjiga je 10.\n");
			pom->br_primeraka-=10;
		}
		i++;
		pom=pom->sledeca;
	}
	printf("\n\n\n_____________________________________________________\n\n\n");
}

void Rasporedi(Knjiga **k, Oblast **o) {
```

```
	Knjiga *pomk=*k;
	Oblast *pomo;
	int br_prosek;
	int i=0;

	//pakuje u magacin preostale knjige iz oblasti top 10 i prebrojava razlicite vrste knjiga po
oblastima
	while(pomk) {
		pomo=NadjiOblast(pomk->ID_oblasti,*o);
		if((pomk->ID_oblasti==-10)) {
			if(pomk->br_primeraka>0) {
				pomo->br_magacin++;
				pomo->magacin=(Knjiga**)realloc(pomo->magacin,sizeof(Knjiga)*pomo-
>br_magacin);
				pomo->magacin[i]=pomk;
				i++;
			}
		} else {
			pomo->br_razlicitih_knjiga++;
	}
		pomk=pomk->sledeca;
	}

	//kako je poznat broj razlicith knjiga po oblastima, racuna koliko primeraka svake knjige treba da
bude najvise na polici u oblasti
	pomo=*o;
	while(pomo!=NULL) {
		if(pomo->br_razlicitih_knjiga>0)
			pomo->prosek=pomo->br_polica*pomo->br_knjiga/pomo->br_razlicitih_knjiga;
		else
			pomo->prosek=0;
		pomo=pomo->sledeca;
	}

	//rasporedjuje knjige koje nisu u top10 po oblastima i pakuje visak u magazin
	pomk=*k;
	i=0; //indeks niza za knjige
	while(pomk) {
		if(pomk->ID_oblasti!=-10) {
			pomo=NadjiOblast(pomk->ID_oblasti,*o);
			if(pomk->br_primeraka > pomo->prosek ) {
				pomk->br_primeraka-=pomo->prosek;
				pomo->br_magacin++;
				pomo->magacin=(Knjiga**)realloc(pomo->magacin,sizeof(Knjiga)*pomo-
>br_magacin);
				pomo->magacin[pomo->br_magacin-1]=pomk;
			} else if(pomk->br_primeraka>0){
				i++;
				pomo->knjige=(Knjiga**)realloc(pomo->knjige,sizeof(Knjiga)*i);
				pomo->knjige[i-1]=pomk;
			}
		}
		pomk=pomk->sledeca;
	}
}

void IspisMagacina(Oblast *p)
{
	Oblast *pom=p;
	int i;
	printf("STANJE U MAGACINU PO OBLASTIMA:\n\n");
	while(pom)
	{
		if(pom->br_magacin)
		{
			printf("\tOblast: %s\n",pom->ime_oblasti);
			for(i=0; i<pom->br_magacin; i++)
			{
				printf("\tAutor knjige: %s\n",pom->magacin[i]->ime_autora);
				printf("\tNaziv knjige: %s\n",pom->magacin[i]->naslov_knjige);
				printf("\tOvih knjiga u magacinu je:%d\n\n",pom->magacin[i]->br_primeraka);
			}
		}
```

```
		pom=pom->sledeca;
	}
	printf("\n\n\n_____________________________________________________\n\n\n");
}

void Nabavka(Knjiga *k, Oblast *o)
{
	Knjiga *pomk=k;
	int i, indikator=0;
	Oblast *pomo;
	printf("NABAVKA:\n\n");
	while(pomk)
	{
		pomo=NadjiOblast(pomk->ID_oblasti,o);
		for(i=0; i<pomo->br_magacin; i++)
		{
			if(pomo->magacin[i]==pomk)
				indikator=1;
		}
		if(indikator==0)
			printf("\tKnjigu %s, autora %s, treba nabaviti.\n\n",pomk->naslov_knjige,pomk-
>ime_autora);
		else
			indikator=0;
		pomk=pomk->sledeca;
	}
}
int main() {
	Oblast *o=NULL;
	Knjiga *k=NULL;

	UcitajOblasti(&o);
	UcitajKnjige(&k);


	Top10(&k,&o);
	Rasporedi(&k, &o);
	IspisMagacina(o);
	Nabavka(k,o);



}
```