ФАКУЛТЕТ ЗА ЕЛЕКТРОТЕХНИКА И ИНФОРМАЦИСКИ ТЕХНОЛОГИИ

ПОДАТОЧНИ СТРУКТУРИ И ПРОГРАМИРАЊЕ 2023/2024

Аудиториски вежби 2

Структури во C - вежби

Задача 1:

Да се напише програма која имитира поедноставена **facebook** апликација. Програмата чува податоци за вашиот **профил**: facebook име, град, датум на раѓање, низаа од пријатели, број на пријатели. **Датумот на раѓање** е од облик: ден, месец, година. За секој **пријател** познати се следниве информации: facebook име, датум на раѓање и порака за sид.

Да се напишат функции со кои може да ги промените основните податоци за вашиот профил: facebook име, град и датум на раѓање.

Да се напише **функција** со која прифаќате пријател и го додавате во низата од пријатели, но само доколку бројот на пријатели не го надминува максимално дозволениот број (5000 пријатели) и соодветно го менувате вашиот број на пријатели.

Да се напише **функција** со која бришете пријател (според facebook име) од низата на пријатели и соодветно го менувате вашиот број на пријатели.

Да се напише **функција** со која им честитате роденден (менувајќи го членот – порака за sид) на сите пријатели кои се родени на датумот кој се проследува како аргумент на функцијата.

Да се напише **главна програма** која ги тестира горенаведените функции.

Решение:

```c
#include <stdio.h>

typedef struct Datum {
	int den;
	int mesec;
	int godina;
}Datum;

typedef struct Prijatel {
	char fbime[30];
	Datum datumPrijatel; /* променлива од една структура може да е член во друга структура */
	char poraka[100];
}Prijatel;

typedef struct Profil {
	char fbime[30];
	char grad[15];
	Datum fbdatum;
	Prijatel listaPrijateli[5000]; /* низа од променливи од една структура како член во друга структура */
	int brPrijateli;
}Profil;

/* во продолжение едноставни функции кои ги менуваат членовите во структурата Profil */

void smeniIme(Profil *prof) /* пренесувањето по покажувач штеди меморија! */
{
	printf("Vnesete go novoto facebook ime\n"); fflush(stdout);
	gets(prof->fbime);
	printf("Vaseto facebook ime e smeneto\n"); fflush(stdout);
}

void smeniGrad(Profil *prof)
{
	printf("Vnesete go imeto na vasiot grad\n"); fflush(stdout);
	gets(prof->grad);
	printf("Imeto na vasiot grad e smeneto\n"); fflush(stdout);
}
```

```
void smeniDatum(Profil *prof)
{
	printf("Vnesete den\n"); fflush(stdout);
	scanf("%d", &(prof->fbdatum.den));
	printf("Vnesete mesec\n");
	scanf("%d", &(prof->fbdatum.mesec)); fflush(stdout);
	printf("Vnesete godina\n"); fflush(stdout);
	scanf("%d", &(prof->fbdatum.godina));
	printf("Datumot na ragjanje e smenet\n"); fflush(stdout);
}

void dodadiPrijatel(Profil *prof, Prijatel novPrijatel)
{
	if (prof->brPrijateli > 5000) {
		printf("Ne mozete da dodadete prijatel\n"); fflush(stdout);
	}
	else
	{
		prof->listaPrijateli[prof->brPrijateli] = novPrijatel;
		/* даме нов пријател на последно место во низата listaPrijateli */
		(prof->brPrijateli)++;
		/* и соодветно го зголемуваме бројот на пријатели за еден */
		printf("Imate eden nov prijatel\n"); fflush(stdout);
	}
}

void brisiPrijatel(Profil *prof, char fbimePrijatel[])
{
	int i, j, najden = 0;
	for (i = 0; i < prof->brPrijateli; i++)
	{
		if (!strcmp(fbimePrijatel, prof->listaPrijateli[i].fbime))
		{
			/* сите пријатели после најдениот пријател ги поместуваме за едно место
во лево */
			for (j = i; j < prof->brPrijateli - 1; j++)
				prof->listaPrijateli[j] = prof->listaPrijateli[j + 1];
			(prof->brPrijateli)--;
			najden = 1;
			break;
			/* избришавме еден пријател */
		}
	}
	if (!najden) { /* или if (i == prof->brPrijateli) */
		printf("Vo vashata lista na prijateli ne postoi prijatel so ime %s \n",
fbimePrijatel); fflush(stdout);
	}
	else {
		printf("Prijatelot so ime %s e izbrisan od vasata lista na prijateli\n",
fbimePrijatel); fflush(stdout);
	}
}

void cestitaj(Profil *prof, Datum rodenden)
{
	int i;
	for (i = 0; i < prof->brPrijateli; i++)
	{ /* ги изминуваме сите пријатели во низата listaPrijateli и ги бараме оние кои се
родени на денот и месецот кој го пренесуваме во променливата rodenden */
		if ((rodenden.den == prof->listaPrijateli[i].datumPrijatel.den) &&
(rodenden.mesec == prof->listaPrijateli[i].datumPrijatel.mesec))
		{
			strcpy(prof->listaPrijateli[i].poraka, "Srekjen rodenden!!!");
			printf("Cestitavte rodenden na %s\n", prof->listaPrijateli[i].fbime);
fflush(stdout);
		}
	}
}
```

```
void pechatiProfil(const Profil *prof) /*пренос по покажувач за заштеда на меморија,
структурата Профил зафаќа многу меморија, проверете со sizeof*/
{
	int i;
	printf("Podatoci za profilot:\n"); fflush(stdout);
	printf("Ime: %s\n", prof->fbime); fflush(stdout);
	printf("Grad: %s\n", prof->grad); fflush(stdout);
	printf("Datum  na  ragjanje:  %-2d.%-2d.%-4d\n", prof->fbdatum.den, prof-
>fbdatum.mesec, prof->fbdatum.godina); fflush(stdout);
	printf("Prijateli:\n"); fflush(stdout);
	for (i = 0; i < prof->brPrijateli; i++) {
		printf("%-15s\n", prof->listaPrijateli[i].fbime); fflush(stdout);
	}
	printf("\n");
}

int main()
{
	Profil mojprofil = { "DonnaP", "Skopje", { 28,12,1986 }, { { "HarveyS", { 2,2,1986 },
"" }, { "RachelZ", { 20,5,1987 }, "" }, { "LouisL", { 15,3,1978 }, "" } }, 3 };

	Prijatel nov;
	pechatiProfil(&mojprofil);
	smeniIme(&mojprofil);
	pechatiProfil(&mojprofil);
	printf("Dodavam nov prijatel\n"); fflush(stdout);
	printf("Vnesete go imeto na vasiot nov prijatel\n"); fflush(stdout);

	scanf("%s", nov.fbime);
	printf("Vnesete go datumot (den, mesec i godina) na ragjanje na vashiot nov
prijatel\n"); fflush(stdout);
	scanf("%d%d%d", &(nov.datumPrijatel.den), &(nov.datumPrijatel.mesec),
&(nov.datumPrijatel.godina)); fflush(stdout);
	printf("Vnesete poraka za zid za vasiot nov prijatel\n"); fflush(stdout);
	scanf("%s", nov.poraka);
	dodadiPrijatel(&mojprofil, nov);
	pechatiProfil(&mojprofil);
	brisiPrijatel(&mojprofil, "RachelZ");
	pechatiProfil(&mojprofil);
	return 0;
}
```

Излез:

```
Podatoci za profilot:
Ime: DonnaP
Grad: Skopje
Datum  na  ragjanje:  28.12.1986
Prijateli:
HarveyS
RachelZ
LouisL

Vnesete go novoto facebook ime
MajaM
Vaseto facebook ime e smeneto
Podatoci za profilot:
Ime: MajaM
Grad: Skopje
Datum  na  ragjanje:  28.12.1986
Prijateli:
HarveyS
RachelZ
LouisL
Dodavam nov prijatel
Vnesete go imeto na vasiot nov prijatel
NovPrijatel
Vnesete go datumot (den, mesec i godina) na ragjanje na vashiot nov prijatel
13
5
1987
Vnesete poraka za zid za vasiot nov prijatel
Zdravo
Imate eden nov prijatel
Podatoci za profilot:
Ime: MajaM
Grad: Skopje
Datum  na  ragjanje:  28.12.1986
Prijateli:
HarveyS
RachelZ
LouisL
NovPrijatel
Prijatelot so ime RachelZ e izbrisan od vasata lista na prijateli
Podatoci za profilot:
Ime: MajaM
Grad: Skopje
Datum  na  ragjanje:  28.12.1986
Prijateli:
HarveyS
LouisL
NovPrijatel
```

Задача 2:

Да се напише програма која ќе симулира едноставно движење на лифт. За **лифтот** познати се: состојбата (стои, се движи нагоре или се движи надолу), спратот на кој моментално се наоѓа и бројот на патници кои се возат во него. За секој **патник** кој може да го повика лифтот познат е спратот на кој патникот тековно се наоѓа, како и спратот до кој сака да стигне.

Да се напише **функција** со која патник го повикува лифтот. Функцијата треба за време на целото возење на лифтот да ги менува: состојбата во која се наоѓа лифтот како и спратот на кој моментално се наоѓа. Кога лифтот ќе стигне до спратот на кој се наоѓа патникот, ако бројот на патници во него не е поголем од дозволениот (максимум 5 патници), да се испише соодветна порака, ако не се менува бројот на патници во лифтот и се повторува движењето до одредишниот спрат.

Во **главната програма** да се иницијализира лифт и патник и да се повика функцијата.

Решение:

```
#include<stdio.h>
#define MAXPAT 5
enum sostojba { stoi, gore, dolu };

struct Lift
{
	enum sostojba sostojba;
	int sprat;
	int brpatnici;
};
typedef struct Lift Lift;

struct Patnik
{
	int sprat;
	int kade;
};

void povikaj(struct Patnik *p, Lift *l)
{
	printf("Patnikot se naogja na sprat %d\n", p->sprat); fflush(stdout);

	if (l->sprat < p->sprat)
	{
		l->sostojba = gore;   /* движиме нагоре дури не стигнеме до спратот на кој се
наоѓа патникот */
		while (l->sprat < p->sprat)
		{
			(l->sprat)++;
			printf("Liftot e na %d sprat\n", l->sprat); fflush(stdout);
		}
	}
	else if (l->sprat > p->sprat)
	{
		l->sostojba = dolu;  /* движиме надолу дури не стигнеме до спратот на кој се
наоѓа патникот */
		while (l->sprat > p->sprat)
		{
			(l->sprat)--;
			printf("Liftot e na %d sprat\n", l->sprat); fflush(stdout);
		}
	}

	l->sostojba = stoi;
	printf("Liftot pristigna\n"); fflush(stdout);
	printf("Patnikot saka da prodolzhi na sprat %d\n", p->kade); fflush(stdout);
```

```
	if ((l->brpatnici) > MAXPAT) {
		printf("Liftot ne mozhe da primi povekje patnici\n"); fflush(stdout);
	}

	else
	{
		(l->brpatnici)++;
		if (l->sprat < p->kade)    /* патникот сака да продолжи нагоре */
		{
			l->sostojba = gore;
			while (l->sprat < p->kade)
			{
				(l->sprat)++;
				printf("Liftot e na %d sprat\n", l->sprat); fflush(stdout);
			}
		}
		else if (l->sprat > p->kade)    /* патникот сака да продолжи надолу */
		{
			l->sostojba = dolu;
			while (l->sprat > p->kade)
			{
				(l->sprat)--;
				printf("Liftot e na %d sprat\n", l->sprat); fflush(stdout);
			}
		}
		printf("Liftot pristigna\n"); fflush(stdout);
		l->brpatnici--;
		l->sostojba = stoi;
	}
}

int main()

{
	Lift l = { 0,0,4 };
	struct Patnik p = { 3,2 };
	povikaj(&p, &l);
	return 0;
}
```

Излез:

```
Patnikot se naogja na sprat 3
Liftot e na 1 sprat
Liftot e na 2 sprat
Liftot e na 3 sprat
Liftot pristigna
Patnikot saka da prodolzhi na sprat 2
Liftot e na 2 sprat
Liftot pristigna
```

Задача 3:

Да се напише програма која опишува првенство во ракомет. За првенството познати се **репрезентациите** кои учествуваат (име на репрезентација, низа од играчи, вкупен број на голови по репрезентација), а за секој **играч** познати се следниве информации: име, презиме, позиција (лево крило, десно крило, лев бек, десен бек, пивот, центар, голман) и постигнати голови.

Да се напише **функција** која го пресметува вкупниот број на голови за секоја репрезентација (познат е бројот на голови на секој играч во репрезентацијата).

Да се напише **функција** која ќе избере најдобри играчи на првенството, во секоја од 7-те позиции и ќе ги прикаже истите.

Во **главната програма** да се иницијализираат неколку репрезентации и да се повикаат функциите.

Решение:

```
#include <stdio.h>
#define IGR 7
#define REP 4 /*заради поедноставеност на програмата */
enum pozicija { lkrilo, dkrilo, lbek, dbek, centar, pivot, golman };

struct Igrach
{
	char prezime[30];
	enum pozicija pozicija;
	int golovi; /* број на голови на играч */
};

struct Reprezentacija
{
	char ime[30];
	struct Igrach igrachi[IGR];
	int golovi; /* вкупен број на голови на репрезентација */
};

void golovi(struct Reprezentacija r[])
{
	int i, j;
	for (i = 0; i < REP; i++)
	{
		r[i].golovi = 0;
		for (j = 0; j < IGR; j++)
			r[i].golovi += r[i].igrachi[j].golovi;

	}
	for (i = 0; i < REP; i++)
		printf("Reprezentacijata %s postigna %d golovi \n", r[i].ime, r[i].golovi);
}

void izberiNajdobri(struct Reprezentacija r[])
{
	struct Igrach najdobri[7]; /* на секоја позиција во низата ќе сместуваме најдобар играч во соодветната позиција. Пример: najdobri [0] = левокрило, najdobri [1] = деснокрило, итн.*/

	int i, j;
	char *pozicija[] = { "levokrilo", "desnokrilo", "levbek", "desenbek", "centar",
"pivot", "golman" };

	for (i = 0; i < IGR; i++)
		najdobri[i].golovi = -1; /* на почеток головите на најдобрите во секоја од 7-те позиции ги иницијализираме со негативна вредност. */

	for (i = 0; i < REP; i++) /* ги изминуваме сите репрезентации и секој играч во нив */
	{
```

```
		for (j = 0; j < IGR; j++)
		{
			switch (r[i].igrachi[j].pozicija)
			{
			case lkrilo: {if (r[i].igrachi[j].golovi > najdobri[lkrilo].golovi)
				najdobri[lkrilo] = r[i].igrachi[j];
				break; }
			case dkrilo: {if (r[i].igrachi[j].golovi > najdobri[dkrilo].golovi)
				najdobri[dkrilo] = r[i].igrachi[j];
				break; }
			case lbek: {if (r[i].igrachi[j].golovi > najdobri[lbek].golovi)
				najdobri[lbek] = r[i].igrachi[j];
				break; }
			case dbek: {if (r[i].igrachi[j].golovi > najdobri[dbek].golovi)
				najdobri[dbek] = r[i].igrachi[j];
				break; }
			case centar: {if (r[i].igrachi[j].golovi > najdobri[centar].golovi)
				najdobri[centar] = r[i].igrachi[j];
				break; }
			case pivot: {if (r[i].igrachi[j].golovi > najdobri[pivot].golovi)
				najdobri[pivot] = r[i].igrachi[j];
				break; }
			case golman: {if (r[i].igrachi[j].golovi > najdobri[golman].golovi)
				najdobri[golman] = r[i].igrachi[j];
				break; }
			default: break;
			}
		}
	}
	for (i = 0; i < IGR; i++)
		printf("Najdobar igrach na pozicija %s: %s so postignati %d golovi\n",
pozicija[i], najdobri[i].prezime, najdobri[i].golovi);
}

int main()
{
	struct Reprezentacija r[REP] = { { "Makedonija",{ { "Alushevski",0,80 },{
"Markovski",1,30 },{ "Mirkulovski",2,70 },{ "Mojsovski",3,60 },{ "Lazarov",4,90 },{
"Markoski",5,65 },{ "Ristovski",6,80 } },0 },{ "Srbija",{ { "Aa",0,90 },{ "Ab",1,30 },{
"Av",2,30 },{ "Ag",3,30 },{ "Ad",4,30 },{ "Agj",5,30 },{ "Ae",6,90 } },0 },{ "Svedska",{ {
"Ba",0,90 },{ "Bb",1,20 },{ "Bv",2,85 },{ "Bg",3,20 },{ "Bd",4,20 },{ "Bgj",5,90 },{ "Be",6,90
} },0 },{ "Germanija",{ { "Va",0,30 },{ "Vb",1,40 },{ "Vv",2,90 },{ "Vg",3,90 },{ "Vd",4,20
},{ "Vgj",5,70 },{ "Ve",6,70 } },0 } };

	golovi(r);
	izberiNajdobri(r);
	return 0;
}
```

Излез:

```
Reprezentacijata Makedonija postigna 475 golovi
Reprezentacijata Srbija postigna 330 golovi
Reprezentacijata Svedska postigna 415 golovi
Reprezentacijata Germanija postigna 410 golovi
Najdobar igrach na pozicija levokrilo: Aa so postignati 90 golovi
Najdobar igrach na pozicija desnokrilo: Vb so postignati 40 golovi
Najdobar igrach na pozicija levbek: Vv so postignati 90 golovi
Najdobar igrach na pozicija desenbek: Vg so postignati 90 golovi
Najdobar igrach na pozicija centar: Lazarov so postignati 90 golovi
Najdobar igrach na pozicija pivot: Bgj so postignati 90 golovi
Najdobar igrach na pozicija golman: Ae so postignati 90 golovi
```

Задача 4:

Да се напише структура за претставување на **квадрат** зададен со темето горе лево и должината на страната.

Да се напише **функција** за поместување на квадратот (поместувањето може да се врши само паралелно со координатните оски). Растојанијата по x и y оската за кои треба да се помести квадратот се пренесуваат како аргументи на функцијата.

Решение:

```
#include <stdio.h>

typedef struct Tochka {
	float x;
	float y;
}Tochka;

typedef struct Kvadrat {
	Tochka teme;
	float strana;

}Kvadrat;

void pomesti(Kvadrat *k, float x, float y) {
	(k->teme).x += x;
	(k->teme).y += y;
}

int main()
{
	Kvadrat k;
	float i, j;

	printf("Vnesete teme i strana\n"); fflush(stdout);
	scanf("%f %f %f", &k.teme.x, &k.teme.y, &k.strana);
	printf("Za kolku da se pomesti kvadratot? \n Po X-oskata:"); fflush(stdout);
	scanf("%f", &i);
	printf("Po Y-oskata:"); fflush(stdout);
	scanf("%f", &j);
	pomesti(&k, i, j);
	printf("Pomesten za (%.2f,%.2f) sega kvadratot e na pozicija: (%.2f,%.2f)\n", i, j,
k.teme.x, k.teme.y); fflush(stdout);
	return 0;
}
```

Излез:

```
Vnesete teme i strana
2
3
5
Za kolku da se pomesti kvadratot?
Po X-oskata:5
Po Y-oskata:5
Pomesten za (5.00,5.00) sega kvadratot e na pozicija: (7.00,8.00)
```