Tema 5

Strukture i datoteke

Strukture

- Pravljenje složenih tipova
  - Od više primitivnih tipova (`int`, `double` itd.)
  - Od više složenih tipova (ranije formiranih struktura)
  - Kombinacije prethodna dva
- Ključna reč `struct`
- Pun naziv novog tipa: `struct novi_tip`
- Primereno za objedinjavanje zajedničkih osobina određene pojave ili objekta
  - Tačka (koordinate)
  - Geometrijska figura (broj, dužina stranica)
  - Student (broj indeksa, ime i prezime, prosek...)
  - Automobil (registarski broj, marka, tip, kubikaža...)

Primer 1

Predstavljanje tačke u dvodimenzionalnom prostoru, unos i ispis.

```c
#include <stdio.h>

struct tacka_st {
    double x;
    double y;
};

void unos_tacke(struct tacka_st *);
void ispis_tacke(struct tacka_st);

int main() {
    struct tacka_st t1, t2;

    unos_tacke(&t1);
    unos_tacke(&t2);

    ispis_tacke(t1);
    ispis_tacke(t2);
```

```
    return 0;
}

void unos_tacke(struct tacka_st *pt) {
    printf("Unesite x i y koordinatu tacke: ");
    scanf("%lf %lf", &(*pt).x, &(*pt).y);
}

void ispis_tacke(struct tacka_st t) {
    printf("(%.2lf, %.2lf)\n", t.x, t.y);
}
```

- Operator dereferenciranja (`*`) ima niži prioritet u odnosu na operator člana strukture (`.`)
- `(*pt).x` i `(*pt).y` su primeri čestih konstrukcija u C programima
  - Kraći zapis (uvek korišćen): `pt->x` i `pt->y`

Primer 2

Rad sa nizovima struktura.

```
#include <stdio.h>

#define MAX_SIZE 30

struct tacka_st {
    double x;
    double y;
};

void unos_tacaka(struct tacka_st *, int *);
void ispis_tacaka(struct tacka_st *, int);

int main() {
    struct tacka_st tacke[MAX_SIZE];
    int n;

    unos_tacaka(tacke, &n);
    ispis_tacaka(tacke, n);
```

```
    return 0;
}

void unos_tacaka(struct tacka_st *t, int *pn) {
    int i;

    do {
        printf("Unesite broj tacaka: ");
        scanf("%d", pn);
    } while(*pn <= 0 || *pn > MAX_SIZE);

    for(i = 0;i < *pn;i++) {
        printf("t[%d] = ", i);
        scanf("%lf %lf", &t[i].x, &t[i].y);
    }
}
```

```
void ispis_tacaka(struct tacka_st *t, int n) {
    int i;

    printf("[");
    for(i = 0;i < n;i++) {
        if(i > 0) {
            printf(", ");
        }
        printf("(%.2lf, %.2lf)", t[i].x, t[i].y);
    }
    printf("]\n");
}
```

Zadatak 1

Na osnovu koda iz prethodnog primera, učitati niz tačaka u ravni, maksimalno 30. Naći tačku koja je najbliža koordinatnom početku. Realizovati zadatak pomoću funkcija.

Primer zaglavlja funkcije koja nalazi tačku najbližu koordinatnom početku:

```
struct tacka_st najbliza_pocetku(struct tacka_st*, int);
```

- Za domaći uraditi isto za tačke u prostoru.

Zadatak 2

Onlajn striming platforma sadržaj koji nudi deli u posebne kategorije. Napraviti strukturu koja sadrži ime kategorije i trenutni broj gledalaca sadržaja. Dozvoliti korisniku da unese niz kategorija od najviše 30 elemenata. Sortirati kategorije po trenutnom broju gledalaca u opadajućem redosledu i ispisati ih na ekran terminala.

Primer vrednosti kategorija:

```
Games    403829
IRL      297405
Music    303699
Esports  498305
Creative 170493
```

Datoteke

- Sekvenca bajtova sačuvana na disku
  - Životni vek datoteke nije direktno uslovljen trajanjem programa
- Binarne i tekstualne
- Rad isključivo preko pokazivača na tip `FILE`
- Funkcije za rad sa datotekama nalaze se u `stdio.h` biblioteci
- Osnovne funkcije za rad sa datotekama
  - `fopen` - otvaranje datoteke
    - Režimi pristupa
  - `fclose` - zatvaranje datoteke
    - Pokazivač tipa `FILE` se ne postavlja na `NULL` vrednost!

Tekstualne datoteke

- Ektenzija `.txt`
- Sadržaj datoteke je isključivo tekst
- Funkcije za rad sa tekstualnim datotekama
  - Univerzalne funkcije za rad sa svim tipovima podataka
    - `fscanf` - formatirano čitanje sadržaja datoteke
    - `fprintf` - formatirani ispis sadržaja u datoteku
  - Specijalizovane funkcije za rad sa nizovima znakova
    - `fgets` - učitavanje određenog broja znakova iz datoteke u niz karaktera
    - `fputs` - ispis određenog broja znakova iz niza znakova u datoteku

Primer 1

Učitavanje iz i zapis u tekstualnu datoteku.

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

#define MAX_SIZE 30

typedef struct tacka_st {
    double x;
    double y;
} TACKA;


FILE *safe_fopen(char *, char *);
void ucitaj_tacke(FILE *, TACKA *, int *);
void ispisi_koordinate(FILE *, TACKA *, int);
```

```c
int main(int argc, char **argv) {
    TACKA tacke[MAX_SIZE];
    int n;
    FILE *pin, *pout;

    if(argc != 3) {
        puts("Primer poziva programa: ./a.out in.txt out.txt");
        exit(EXIT_FAILURE);
    }

    pin = safe_fopen(argv[1], "r");
    ucitaj_tacke(pin, tacke, &n);
    fclose(pin);

    pout = safe_fopen(argv[2], "w");
    ispisi_koordinate(pout, tacke, n);
    fclose(pout);

    return 0;
}
```

```
FILE *safe_fopen(char *name, char *mode) {
    FILE *pf = fopen(name, mode);

    if(pf == NULL) {
        printf("File %s could not be opened.\n", name);
        exit(EXIT_FAILURE);
    }

    return pf;
}

void ucitaj_tacke(FILE *pin, TACKA *t, int *pn) {
    int i = 0;

    while(fscanf(pin, "%lf %lf", &t[i].x,
                                 &t[i].y) != EOF) {
        i++;
    }

    *pn = i;
}
```

```c
void ispisi_koordinate(FILE *pout, TACKA *t, int n) {
    int i;

    for(i = 0;i < n;i++) {
        fprintf(pout, "(%.2lf, %.2lf)\n", t[i].x, t[i].y);
    }
}
```

Primer 1 - dodatna pojašnjenja

- `typedef` za preimenovanje tipa (zarad kraćeg zapisa)
- Funkcija `exit` deo je biblioteke `stdlib.h`
  - Za razliku od `return` okončava program iz bilo koje funkcije
  - Konstante definisane u `stdlib.h`
    - `EXIT_SUCCESS` - uspešno izvršen program
    - `EXIT_FAILURE` - neuspešno izvršen program
- Pomoćna funkcija `safe_fopen`
  - Provera da li je uspešno otvorena datoteka je neophodna
  - U slučaju neuspešno otvorene datoteke, `fopen` vraća `NULL`
    - Primer: ne postoji fajl koji se otvara u režimu za čitanje `"r"`

Primer 1 - poziv programa i rezultat

Za poziv programa:

```
./a.out tacke.txt koordinate.txt
```

Gde je sadržaj ulazne datoteke `tacke.txt`:

```
1 2
2 3
3 4
4 5
```

Na osnovu kog je dobijen sadržaj izlazne datoteke `koordinate.txt`:

```
(1.00, 2.00)
(2.00, 3.00)
(3.00, 4.00)
(4.00, 5.00)
```

Zadatak 1

Napisati program koji iz tekstualne datoteke učitava `n` elemenata strukture tipa `auto_st` koja se sastoji iz sledećih polja:

- Marka automobila (jedna reč, do 20 karaktera)
- Kubikaža (prirodan broj)
- Godište (prirodan broj)

Ime ulazne tekstualne datoteke i kubikaža zadaju se kao argumenti komandne linije. Na osnovu zadatih informacija pronaći najnoviji auto sa kubikažom ne većom od zadate i rezultat upisati u tekstualnu datoteku, koja ima naziv oblika `<marka_automobila>.txt`.

Zadatak 2

Napisati program koji kao argument komandne linije prima jednu reč i celobrojnu vrednost. Šifrirati reč pomoću zadatog broja koristeći Cezarovo šifriranje i upisati celobrojnu vrednost i, bez razmaka, dobijeni rezultat u tekstualni fajl `rec.txt`. U slučaju da program ne može da otvori zadatu izlaznu datoteku, izaći iz programa sa kodom greške `1`.

Zadatak 3

Proširiti `Zadatak 6` ("Igra vešala") iz teme "Funkcije" da čita i dekodira reč iz ulazne datoteke, čije ime je zadato argumentom komandne linije. Doraditi da program koristi dekodiranu reč kao zadatu, umesto string konstante, što je bio slučaj u prethodnoj verziji zadatka.

Uraditi sledeće obrade kategorija grešaka u programu:

- Ukoliko ulazna datoteka ne postoji, izaći iz programa sa kodom greške `1`
- U slučaju da pročitani string ne počinje brojem, ispisati poruku o grešci prilikom dekodiranja i izaći iz programa sa kodom greške `2`