Datoteke

Če želimo rezultate programa trajno shraniti, da bi si jih lahko ogledali večkrat (brez ponovnega zagona programa), jih moramo shraniti na datoteko. Na datoteki si tudi lahko pripravimo podatke, ki jih bo obdelal naš program (rezultati meritev, ...).

Ločimo dve vrsti datotek: znakovne in dvojiške. Razlikujeta se v načinu, kako so podatki zapisani. V primeru znakovnih datotek so podatki zapisani v obliki, kot jih vidimo na zaslonu, torej z znaki (od tod tudi ime). V primeru dvojiških datotek pa so podatki shranjeni v obliki, kot so zapisani v pomnilniku. Uporabnikom prijaznejše so znakovne datoteke, saj jih lahko odpremo in spreminjamo s katerimkoli urejevalnikom znakovnih datotek (Notepad, TextPad, ...), tako da se bomo tukaj omejili samo na delo z znakovnimi datotekami.

Odpiranje

Pred prvim posegom na datoteko (branje, pisanje) moramo datoteko odpreti. V ta namen uporabimo funkcijo `fopen`, ki za parametra dobi niza. Prvi je ime datoteke, ki jo želimo odpreti, drugi pa določa, na kakšen način jo želimo odpreti. Ime datoteke je lahko relativno glede na mapo, kjer se nahaja program, ali pa absolutno (z oznako diska in celo potjo). Za ločila med mapami lahko uporabljamo nagibnico `\` ali poševnico `/`. Če uporabljamo nagibnice, jih moramo pisati po dve skupaj. Drugi parameter je sestavljen iz enega ali več znakov. Običajno določimo način, kako želimo odpreti datoteko (`r` za branje, `w` za pisanje ali `a` za dodajanje) in vrsto datoteke (`t` za znakovno ali `b` za dvojiško). Funkcija vrne datotečni kazalec (`FILE *`), ki ga potem uporabljamo pri vseh nadaljnih posegih na datoteko. Če datoteke ni bilo mogoče odpreti, vrne `NULL`. Če datoteko odpremo za branje, mora ta že obstajati, sicer pride do napake. Če datoteko odpremo za pisanje in ta še ne obstaja, jo ustvari, sicer pa pobriše njeno vsebino. Odpiranje datoteke za dodajanje deluje podobno kot za pisanje, le da ne pobriše morebitne vsebine (dodajamo lahko samo na konec).

```
FILE *f = fopen("datoteka.txt", "rt");
```

Pisanje na znakovno datoteko

Za pisanje imamo na razpolago funkcijo `fprintf`, ki deluje podobno kot `printf`. Razlika je samo v tem, da ji moramo kot prvi parameter podati datotečni kazalec, ki pove, na katero datoteko želimo pisati. V pomoč sta tudi funkciji `fputs` in `fputc`, ki sta namenjeni pisanju niza oziroma znaka na datoteko.

```
fprintf(f, "%d + %d = %d\n", a, b, a + b);
fputs(niz, f);
fputc(znak, f);
```

Branje z znakovne datoteke

Za branje imamo na razpolago funkcijo `fscanf`, ki deluje podobno kot `scanf`. Razlika je samo v tem, da ji moramo kot prvi parameter podati datotečni kazalec, ki pove, s katere datoteke želimo brati. V pomoč sta tudi funkciji `fgets` in `fgetc`, ki sta namenjeni branju niza oziroma znaka z datoteke.

```
int n = fscanf(f, "%d:%d", &ura, &min);
char *str = fgets(niz, MAX, f);
int znak = fgetc(f);
```

Funkcija `fscanf` dobi za prvi parameter datotečni kazalec, sledi pa formatni niz, ki določa, koliko in kakšne vrste podatkov želimo prebrati (poznamo že `%d` za celo število, `%lf` za realno število, `%c` za znak in `%s` za niz (besedo)). V formatnem nizu lahko uporabimo več formatnih

določil, če so podatki ločeni s kakšnim posebnim znakom, pa vmes napišemo še ta znak. Če bi na primer želeli prebrati uro v obliki `hh:mm`, bi to napisali tako, kot v primeru zgoraj. Funkcija vrne število uspešno prebranih podatkov oziroma celoštevilsko vrednost `EOF`, če ni mogla prebrati ničesar.

Funkcija `fgets` je namenjena branju vrstice. Za parametre dobi niz (tabelo znakov), kamor naj shrani prebrano vrstico, dolžino te tabele, da ne prebere več, kot lahko shrani, in datotečni kazalec, ki pove, iz katere datoteke naj bere. V tabelo zapiše tudi morebitni prebrani znak za konec vrstice in znak za konec niza. Funkcija vrne kazalec na začetek prebranega niza, oziroma kazalec `NULL`, če ni mogla prebrati ničesar.

Funkcija `fgetc` je namenjena branju znaka. Za parameter dobi datotečni kazalec, ki pove, iz katere datoteke naj bere. Vrne kodo prebranega znaka (`int`) oziroma celoštevilsko vrednost `EOF`, če ni mogla prebrati ničesar.

Zapiranje

Ko datoteke ne potrebujemo več, jo moramo zapreti, sicer lahko izgubimo del njene vsebine.

```
fclose(f);
```