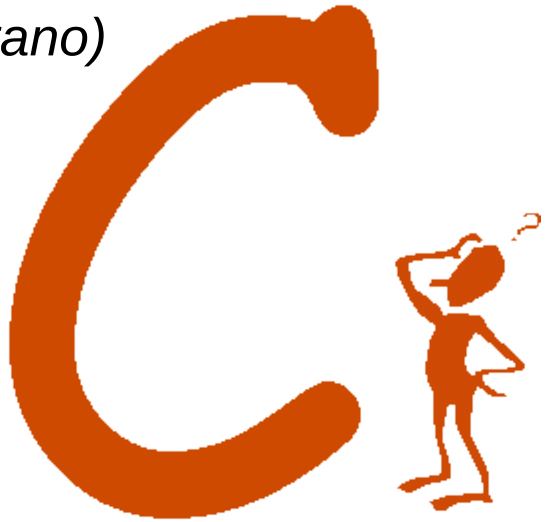
Programski jezik C

Delo z datotekami (formatirano, neformatirano)
knjižnice,
C-jev predprocesor,
krmiljenje zaslona



Vhodno izhodne operacije

V datoteki stdio.h so definirani:

kazalci na datoteke (FILE): stdin, stdout, stderr
NULL (ki je enak 0)
EOF (ki je enak -1)
FILE (ki je typedef za podatkovno strukturo)

Funkcije s standardnim vhodom, izhodom:

int printf (format [,arg, arg,..arg])	Formatiran izpis na standardni izhod
int scanf (format [,kazalec, kazalec, ..])	Formatirano branje s standardnega vhoda
int getchar( )	Branje znaka s standardnega vhoda
int putchar (int)	Izpis znaka na standarni izhod
~~char *gets(char str[80])~~	~~Branje niza s standardnega vhoda~~ (nevarna, nevarna)
char *puts(char str[80])	Izpis niza na standardni izhod

Delo z datotekami

Odpiranje datoteke
Primer:
Odprimo za branje datoteko z imenom **datoteka**.

FILE *fd ; fd = fopen("datoteka", "r");

Najprej smo morali definirati kazalec fd, ki ga bomo kasneje uporabljali pri vseh operacijah z odprto datoteko. Temu kazalcu pravimo "opisnik datoteke" (file descriptor). Drugi parameter v klicu funkcije fopen je v danem primeru "r", zato bo datoteka odprta za branje.

Seznam vseh možnosti

"r" Odpri datoteko za branje
"w" Odpri datoteko za pisanje. Če je še ni, jo tvori, če je že, jo povozi
"a" Odpri datoteko za pisanje. Če je še ni, jo tvori, sicer dodajaj na konec.
"r+" Datoteka bo odprta za branje in pisanje
"w+" Datoteka bo odprta za branje in pisanje
"a+" Datoteka bo odprta za branje in pisanje

Zapiranje datoteke
fclose (fd);

Branje iz datoteke... Zapis v datoteko

Pri navedbi možnih funkcij predpostavimo, da smo deklarirali dva kazalca na datoteki (fp1 in fp2), en niz (s) in eno spremenljivko tipa char:

```
FILE  *fp1, *fp2;   /* kazalca na datoteki */char *s          /* kazalec na niz */int c;int size, n;
```

Seznam funkcij za branje in zapis v datoteko:

c = getc (fp1);	**/* To je makro */**
c = fgetc(fp1);	**/* To je funkcija */ Branje znaka iz datoteke (podobno getchar())**
ungetc(c,fp1)	**Vrne znak na standardni vhod**
fgets (s,n,fp1);	**Branje (največ n) znakov iz datoteke v niz s**
fscanf (fp1, format, &arg1, &arg2,..);	**Formatirano branje, podobno kot scanf()**
fread (s, size, n, fp1);	**Neformatirano (binarno) branje n elementov,velikosti size, pomnjenih v polju s. Funkcija vrne število prebranih elementov.**
putc (ch, fp2)	**Zapis znaka v datoteko**
fputs (s,fp2);	**zapis niza v datoteko**
fprintf (fp2,format, arg1, arg2,..);	**Formatiran zapis v datoteko**
fwrite (s, size, n, fp2);	**Neformatirani (binarni) zapis n elementov velikosti size. s kaže na polje elementov. Funkcija vrne število zapisanih elementov.**

Primer: Kopiranje ene datoteke v drugo

```
#include < stdio.h >      /* Kopiranje datoteke */

void main() {
        FILE *fp1,  *fp2;
        int ch;
        if ((fp1= fopen("inpFile", "r")) ==NULL) {
            fprintf(stderr,"Datoteke ne morem odpreti");
            exit (1);
        }
        fp2 = fopen ("outFile", "w");
        while ( (ch = getc(fp1)) != EOF) putc(ch, fp2);
        fclose( fp1); fclose (fp2);
        exit (0);
}
```

Branje datoteke (znak po znak) DEMO

Branje –pisanje (znak po znak) DEMO

Branje –pisanje (vrstico za vrstico) DEMO

Branje (besedo za besedo) DEMO

Naključno branje in pisanje datoteke

Pri naslednjih funkcijah uporabljamo spremenljivko tipa "long integer", ki predstavlja odmik (in torej položaj v datoteki), merjen v bytih

```
FILE *fp;
 long odmik;      /* v datoteki */
 int odkod ;      /* od kod se steje odmik  */
```

Možne funkcije:

rewind (fp); Postavi odmik na začetek datoteke
fseek (fp, odmik, odkod);
Drugi parameter -odmik -pove, od kod dalje bo sledili branje ali zapis v datoteko. Tretji parameter ima lahko eno od vrednosti:
SEEK_SET .... Začetek datoteke
SEEK_CUR.....Trenutni položaj v datoteki
SEEK_END......Konec datoteke

odmik = ftell (fp); Dobimo trenutni položaj (odmik) v datoteki

DEMO

Preostale funkcije za delo z datotekami

ferror (fp) Vrne TRUE , če je indikator napake za dani (datotečni) tok (stream) setiran (zaradi neuspele predhodne vh/izh operacije)

feof (fp) Vrne TRUE, če želimo brati po koncu datoteke

clearerr (fp) Resetira indikatorja vh/izh napake in konca datoteke

fflush (fp) Sistem zapisuje podatke v datoteko preko medpomnilnika. Ta funkcija forsira prepis podatkov iz medpomnilnika v datoteko.

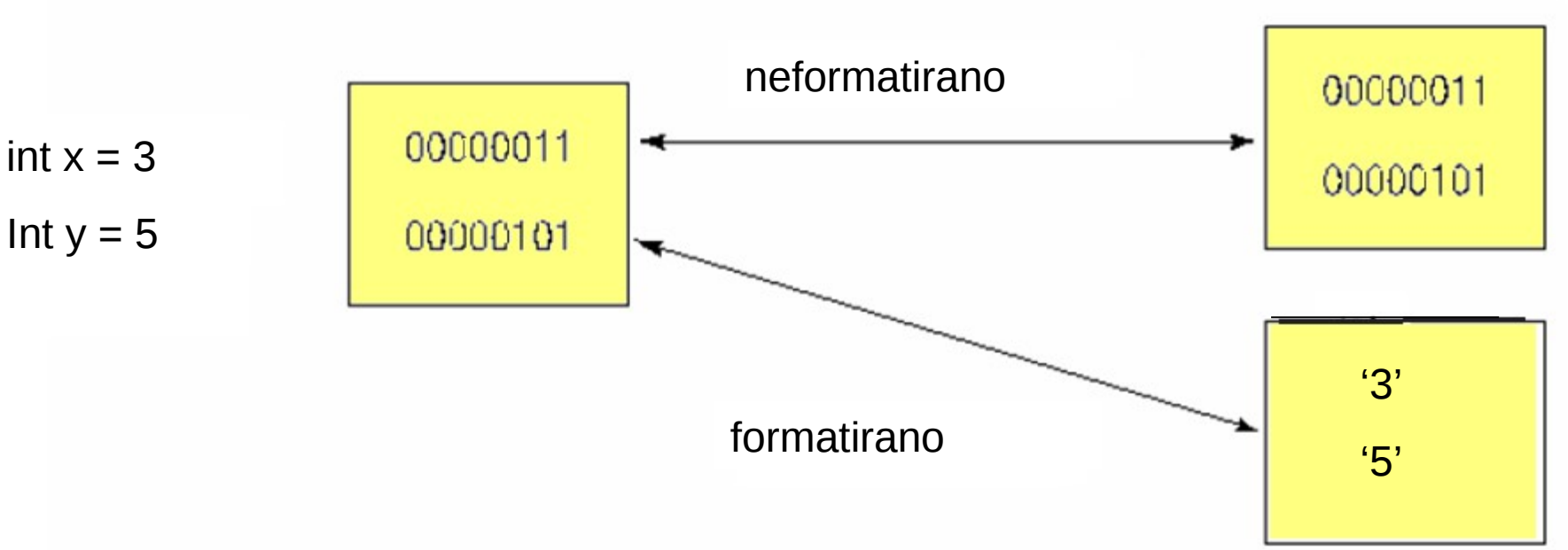
char buffer [BUFSIZ] setbuf (fp, bufer) Sami zagotovimo polje za medpomnilnik.BUFSIZ je definiran v stdio.h .

Če namesto naslova polja buffer navedemo (char *) NULL, bo do operacije brez medpomnilnika (unbuffered)

Formatiran, neformatiran vhod-izhod

Formatiran izhod
- •Podatki pretvorjeni v ASCII neodvisno od vhodnega formata

Neformatiran izhod
- •Zapis na disk brez konverzije (koda takšna, kot je)



ASCII koda:

‘3’ = 00110011 = 033 (hex) = 51(dec)
‘5’ = 00110101 = 035 (hex) = 53(dec)

Demo1

Primer binarnega pisanja in branja

Preprost primer binarnega pisanja in branja

Bolj resen primer binarnega pisanja in branja

Binarno pisanje DEMO

Binarno Branje DEMO

```
Delavec:JanezPeteK    leta:10
 1000.0  9001.0  1100.0   0.0   0.0  2000.0  5000.0  3000.0  1000.1  3500.0

Delavec:AnaPag    leta:11
 400.0  500.0  1300.1  350.0  745.0  3000.0  200.0  100.0  100.0  50.0

Delavec:Natasa    leta:10
 950.0  1050.0  1350.0  410.0  797.0  200.4  2600.0  2000.0  1500.0  2000.0

Delavec:Peter    leta:11
 1000.0  9000.0  1100.0   0.0   0.0  2000.4  5000.0  3000.0  1000.0  3500.0
```

Neformatirano pisanje

int fwrite(void *ptr, int size, int num_items, FILE *fp);

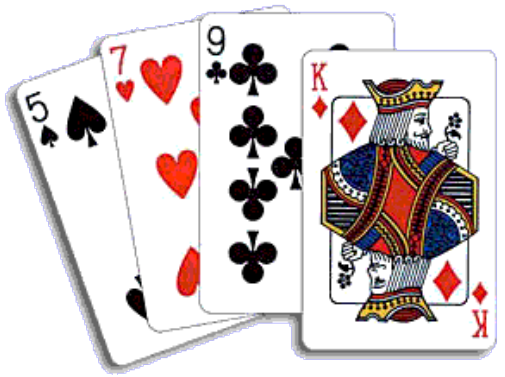
```
#include <stdio.h>
struct karta {
    int vrednost;
    char barva;
};

int main(void) {
    struct karta paket[ ] = { {2,'s'}, {7,‘k'}, {8,'c'}, {2,'s'}, {2,'h'} };

    FILE *fp= fopen("unformated", "wb");
    printf("%d", sizeof(struct karta) );
    fwrite(paket, sizeof(struct karta), 5, fp);     /* ta dva stavka naredita
isto */
    fwrite(paket, 1, sizeof(struct karta)*5, fp); /* ta dva stavka naredita
isto */
```

Neformatirano branje

int fread (void *ptr, int size, int num_items, FILE *fp);



```
#include <stdio.h>
struct karta {
    int vrednost;
    char barva;
};

Int main(void){
    struct karta paket[10];
    FILE *fp= fopen("unformated", "rb");
    fread(paket, sizeof(struct karta), 10, fp);   /* beremo 2 * 5 kart  */
    fclose(fp);
    printf("vrednost= %d, barva = %c\n", paket[6].vrednost, paket[6].barva );
}
```

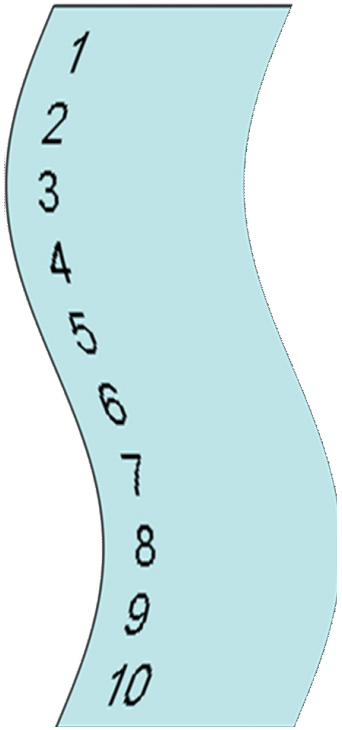
Formatirano pisanje datoteke

Zapis 10 števil v datoteko

```
#include <stdio.h>
#define MAX 10

int main() {
    FILE *f;
    int x;
    f=fopen(“stevila.txt","w");
    if (!f) return 1;
    for (x=1; x<=MAX; x++) fprintf (f,"%d\n",x);
    fclose(f);
    return 0;
}
```

stevila.txt



Formatirano branje datoteke

Branje vrstic v datoteki in izpis na zaslon

```
#include <stdio.h>

int main() {
    FILE *f;
    char s[1000];
    f=fopen(“pesmica.txt","r");
    if (!f) return 1;
    while (fgets(s,1000,f)!=NULL) printf("%s",s);
    fclose(f);
    return 0;
}
```

pesmica.txt

Demo