Универзитет „Св. Кирил и Методиј“ во Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Датотеки

Структурно програмирање

ФИНКИ 2014

Датотеки

- Сместувањето на податоци во променливите (кои се чуваат во работната меморија) е само привремено.
- За перманентното чување на големи количества податоци се користат датотеки.
  - Датотеките се основниот ентитет за чување на податоци на перманентните медиуми.
  - Датотеките претставуваат едноставна секвенца од бајти која завршува со ознака за крај на датотеката (end-of-file marker).

Датотеки

- Перманентните медиуми вообичаено се третираат како надворешни влезно-излезни уреди.
  - За работата со ваквите уреди да биде транспарентна за програмерот и што понезависна од природата на уредот, се користи еден вид на апстракција помеѓу самиот уред и програмата.
  - Оваа апстракцијата се вика тек (stream).
- Во програмскиот јазик Ц преку апстракцијата тек, програмите на ист начин се обраќаат и кон датотеките на диск, тастатурата, екранот, влезно излезни порти, мрежни ресурси, печатачи, ...
- Датотеката е физички ентитет кој прима и/или праќа податоци.

Датотеки

- Со отворањето на датотека се креира тек кој се асоцира кон неа и целата комуникација со датотеката понатаму се одвива преку овој тек. Во секоја Ц програма автоматски се креираат 3 тека:
  - `stdin` – стандардниот влез асоциран кон тастатурата;
  - `stdout` - стандардниот излез асоциран кон екранот; и
  - `stderr` – стандардниот излез за грешки, повторно асоциран кон екранот.

FILE структура

- Во библиотеката на функции stdio.h постои дефинирана структура за опис на датотеки која се вика FILE.
  - Во неа се чуваат клучните податоци потребни за работа со датотеката како нејзиното име, почетната локација, бројот на знаци во баферот, начинот на работа и др.
  - Пред да се отпочне каква било работа со датотеката таа мора да се *отвори*.

Отворање датотека

- Отворање на датотека се извршува со помош на функцијата fopen( ) која ги извршува следните активности:
  - на структурата FILE и ги доделува потребните информации за комуникација на програмите со датотеката;
  - враќа покажувач кон локацијата на која се наоѓаат податоците за структурата.
- Сите понатамошни операции со датотеката се одвиваат преку овој покажувач, односно тој е задолжителен аргумент на сите функции за работа со датотеки.
- Истиот обезбедува дистинкција со која од отворените датотеки се работи.

Отворање датотека

За отворање на датотека потребни се информации за:

- името на датотеката и
- видот на датотеката (текстуална/бинарна)
- начинот на кој ќе и се пристапува (пишување/читање/додавање...)

Декларирањето покажувачи за датотеките се врши со:

```
FILE *pokf1, *pokf2, ... , *pokfn;
```

а нејзино отворање со функцијата:

```
FILE *fopen(char *ime, char *nacin_na_rabota);
```

- Функцијата враќа покажувач кон структура од видот FILE во која ќе се чуваат податоците за отворената датотека.
- Прима два аргумента: име на датотеката која треба да се отвори и начин на (режим во) кој треба да се отвори.
- Ако отворањето на датотеката не е успешно функцијата враќа NULL покажувач.

Пр:
```
pokf1 = fopen("myFile.dat", "rb");
```

Начини на работа со датотека

- Начинот на работа во најгенералниот случај може да биде:
  - r - читање од датотека
  - w - пишување во датотека
  - a - додавање на крајот на датотеката
- како и комбинации од некој од овие знаци и знаците b, + и t, при што:
  - b- значи работа со бинарна датотека;
  - +- значи отворање на датотеката и за читање и за пишување.

Начини на работа со датотека

Начините на отворање за датотека според ANSI стандардот за Ц се следниве:

- "r“ отворање текстуална датотека само за читање
- "w“ креирање текстуална датотека за запишување
- "a“ додавање текст на крајот на датотеката
- "rb“ отворање бинарна датотека само за читање
- "wb“ креирање бинарна датотека за запишување
- "ab“ додавање податоци на бинарна датотека
- "r+“ отворање текстуална датотека за читање и запишување
- "w+“креирање текстуална датотека за читање и запишување
- "a+“ отворање текстуална датотека за читање и запишување
- "rb+“ отворање бинарна датотека за читање и запишување
- "wb+“ креирање бинарна датотека за читање и запишување
- "ab+" отворање бинарна датотека за читање и запишување

Вообичаено користење на функцијата за отворање

Вообичаен е следниов начин на користење на функцијата:

```
FILE *pd;
if((pd = fopen("test","w")) == NULL)
{
    puts("Ne moze da se otvori datotekata");
       exit(-1);
}
...
```

со кој пред да се обидеме да пишуваме во, или читаме од датотеката, се проверува дали истата е успешно отворена.

Ако датотеката се отвора за пишување, и истата не постои се креира, а ако постои се пребришува.

Функции за читање и пишување

- Во Ц се дефинирани само еден вид на датотеки – секвенцијални, односно сите запишувања и читања од датотеките се одвиваат во секвенцијален редослед.
- За читање и пишување на единечни знаци во датотеката се користат специјални влезно-излезни функции fgetc( ) и fputc( )
- (стандардот ANSI ги прифаќа и алтернативните имиња getc() и putc()).

```
int fgetc(FILE  * fptr);
int fputc(int promenliva, FILE * fptr);
```

Функции за читање и пишување

- fgetc( ) го чита следниот бајт од датотеката и го враќа како резултат во понискиот бајт од целобројната променлива.
- fputc( ) го запишува понискиот бајт од целобројната променлива во датотеката и истата ја враќа како резултат или враќа EOF ако дошло до грешка.
- По завршената работа датотеката се затвора со функцијата fclose( ):

```
int fclose(FILE *pokazuvac_na_datoteka);
```

- fclose( ) враќа 0 ако затворањето е успешно и -1 ако е неуспешно.

```c
#include <stdio.h>
#include <stdlib.h>
void main() {
char niza[80] = "Test za rabota so datoteki";
FILE *fp; char *p=niza; int i;
/* otvaranje na datoteka za zapisuvanje */
if((fp = fopen("datoteka", "w"))==NULL) {
   printf("Ne moze da se otvori datotekata\n");
exit(1);
}
/* zapisuvanje niza na disk */
while(*p) {
   if(fputc(*p, fp)==EOF)
   {
      printf("Greska pri zapisuvanje\n");
      exit(1);
   }

   p++;
}
fclose(fp);
```

Програма која отвора датотека за запишување, запишува во неа, ја затвора, ја отвора за читање, чита од неа и испишува на екран.

```
/* otvaranje na datoteka za citanje */
if((fp = fopen("datoteka", "r"))==NULL)
{
   printf("Ne moze da se otvori datotekata\n");
   exit(1);
}
/* citanje od datotekata */
for(;;)
{
   if((i = fgetc(fp)) == EOF)
      break;
      putchar(i);
}
fclose(fp);
}
```

Програма која отвора датотека за запишување, запишува во неа, ја затвора, ја отвора за читање, чита од неа и испишува на екран.

Други функции за работа со датотеки

```
fgets(pok_na_niza, max_dolz, pok_na_dat);
fputs(pok_na_niza, pok_na_dat);
fprintf(pok_na_dat,"kontrolna_niza", lista na promenlivi);
fscanf(pok_na_dat,"kontrolna_niza", lista na pokazuvaci na promenlivi);
```

fgets() чита знаци од датотеката и ги сместува во меморијата кон која покажува покажувачот.

- Знаците се читаат од датотеката сè додека не се наиде на '\n' или EOF (не заврши датотеката) или додека не се прочитаат максимално n-1 знаци.
- Ако знакот '\n' е прочитан влегува во низата и зад него се додава NULL терминатор.
- Како резултат функцијата враќа покажувач кон прочитаната низа, а ако датотеката заврши или дојде до грешка при читањето, враќа NULL.
- За утврдување дали неуспешното читање е резултат на грешка или EOF код, се користат функциите feof( ) и ferror( ).

`int feof(FILE *file)` - одредува дали настапил крај на датотеката

`int ferror(FILE *file)` - одредува дали дошло до грешка

Функциите getw и putw

- Функциите се слични на getc и putc и се употребуваат за читање и снимање цели броеви во датотека.
- Корисни се кога се работи со податоци кои се состојат само од цели броеви
- Генералната форма на getw и putw е:

```
putw(integer,fp);
getw(fp);
```

Пример снимање податоци за колоните X и У во датотека

```
#include <stdio.h>
void vpisi(FILE *pdat)
{
    float x, y;
    char xnaslov[31], ynaslov[31];
    int i=0;
    printf("Vnesi naslov za X kolona: ");
    scanf("%30s",xnaslov);
    printf("Vnesi naslov za Y kolona: ");
    scanf("%30s",ynaslov);
    fprintf(pdat, "%30s%30s\n", xnaslov, ynaslov);
    while(1) {
        printf("Vnesi x%d, y%d : ", i, i);
        if(scanf("%f%f", &x, &y)!=2) break;
        fprintf(pdat, "%30.8f%30.8f\n",x,y);
        i++;
    }
}
```

```
int main()
{
FILE *pd;
    if((pd = fopen(“Koloni.txt”, "w")) == NULL) {
     printf("Ne se otvori datotekata %s", “Koloni.txt”);
     exit(1);
    }
    else {
     vpisi(pd);
     fclose(pd);
    }
    return(0);
}
```

Читање/снимање бинарни податоци

- Податоците што се запишуваат во датотека со fprintf се запишуваат во текстуален формат.
- Снимањето на податоци во бинарна форма заштедува простор на дискот и го забрзува преносот на податоците.
- Функции што овозможуваат читање и запишување на податоците во бинарна форма се fread( ) и fwrite( ).

```
int fwrite(int *bafer, int golemina, int n,
      FILE *pokazuvac_na_datoteka)
int fread(int *bafer, int golemina, int n,
      FILE *pokazuvac_na_datoteka)
```

каде

- *bafer* е покажувач на почетокот на меморискиот блок што ќе се сними во датотеката, односно во кој ќе се сместат прочитаните податоци;
- *n* е бројот на елементите во баферот;
- *golemina* е должината на секој од елементите изразена во бајти.

И двете функции како вредност го враќаат бројот на елементи што се запишани или прочитани.

Овој број се разликува од вредноста на n само ако настанала грешка или е достигнат крајот на датотеката.

Директен пристап до податоци во датотека

Функции за поместување низ датотеката:

`rewind(FILE *f)` - поместување на почетокот на датотеката

`int fseek(FILE *f, long offset, int pocetok)` релативно поместување за offset бајти во зависност од вредноста pocetok. Ако *pocetok* е:

SEEK_SET се врши позиционирање во однос на почетокот на датотеката

SEEK_CUR се врши позиционирање во однос на моменталната позиција во датотеката

SEEK_END се врши позиционирање во однос на крајот на датотеката

Следната наредба за читање/пишување ќе се изврши на новопоставената позиција.

fseek враќа 0 ако позицијата била успешно поставена.

ftell

Со функцијата

long ftell(FILE *pokazuvac_na_datoteka)

може да се испита тековната позиција во датотеката која е претставена како број на бајти од нејзиниот почеток.

Моменталната позиција во датотеката може да се промени во почетната и со функцијата `rewind(FILE *fp)` која е идентична на `fseek(fp, 0L, SEEK_SET)`

Да се напише програма која за дадена текстуална датотека ке изброи колку зборови подолги од 3 букви почнуваат и завршуваат на иста буква. Да не се прави разлика мегу голема и мала буква. Зборовите се составени од произволен број на букви, а мегусебно се одделени со најмалку еден специјален знак, цифра или белина. Името на влезната датотека се задава од командна линија, а ако не е зададено се чита од стандардниот влез.

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
main(int argc, char *argv[])
{
    char prva=0,posledna=0,c;
    FILE *fin;
    int b=0;  /* brojac na zborovi */
    int iw=0; /* oznaka deka se naogame vo zbor */
    int len=0;/* dolzina na zborot */
    if(argc>2) /* ako ime poveke od eden parametar vo komandnata linija */
    {                  /* ispecati kako se upotrebuva programata */
        printf("Upotreba: %s ime_na_datoteka\n",argv[0]);
        exit(-1);
    }
    if(argc==1)  /* ako ne e zadaden parametar vo komandnata linija */
        fin=stdin;             /* citaj od standardniot vlez */
    else if((fin=fopen(argv[1],"r"))==NULL)
    {          /* inaku od datotekata zadadena vo komandnata linija */
     fprintf(stderr,"Ne mozam da ja pronajdam datotekata %s\n",argv[1]);
     exit(0);
    }
```

```
    while((c=fgetc(fin))!=EOF)    /* citaj znak po znak se do krajot */
    {
        if(isalpha(c))      /* ako e procitana bukva */
        {
            if(!iw)         /* ako ne se naogas vo zbor */
            {
                iw=1;       /* obelezi deka si vlegol vo zbor */
                prva=c;     /* ova e voedno i prvata bukva od zborot */
            }
            len++;          /* izbroj ja stotuku procitanata bukva */
            posledna=c;     /* zapomni ja poslednata procitana bukva */
        }
        else                /* ako e procitanoto ne e bukva */
        {
            if(iw)          /* ako si se naogal vo zbor */
            {
             iw=0;          /* obelezi deka si izlegol od zbor */
             if(len>3 && toupper(prva)==toupper(posledna))
              b++; /*dokolku zborot gi zadovoluva uslovite izbroj go*/
              len=0;/*resetiraj ja dolzinata za merenje na sl. zbor*/
            }
        }
    }
    printf("%d zborovi\n",b);
}
```

Прашања?