Файлов ВХОД/ИЗХОД

Файл наричаме всеки конкретен набор от информация (данни), реализиран на конкретен хардуерен носител (хард диск, магнитна лента, CD, DVD и пр.), с възможност да бъде обработван (изменян, изтриван, допълван) чрез специализирано за това хардуерно устройство.

Съвкупността от действия за работа с файлове, свързани най-вече с четенето от фалове и писането във файлове се нарича Файлов ВХОД/ИЗХОД (**В/И**).

В езика С не съществуват оператори за реализиране на **В/И**. Въвеждането и извеждането на данни се осъществява чрез входно/изходни функции, дефинирани в хедърния файл **STDIO.H**

Самите функции биват два вида:

- функции за работа с файлове на високо ниво (буфериран вход/изход);
- функции за работа с файлове на ниско ниво (небуфериран вход/изход).

Файлов ВХОД/ИЗХОД

Етапи при работа с файлове

а) ***Отваряне на файла*** – осъществява се връзка между абстрактния файл и физическото устройство.

б) ***Обработка*** – обмен на данни (запис, четене, актуализация).

в) ***Затваряне на файла*** – преустановяване на връзката между файла и физическото устройство.

Видове файлове

а) **Текстов** файл: ***Текстовият файл*** представлява последователност от символи, като всеки символ се съхранява в отделен байт. Текстовият файл може да бъде разделен на редове, като всеки ред е с произволна дължина и завършва със символа за нов ред **'\n'**.

б) **Двоичен** (бинарен) файл: ***Двоичният файл*** съдържа двоичното представяне на данните. Той е последователност от байтове, които точно съответстват на байтовете във външното устройство.

В текстовите файлове съответствието не е точно: например символът „нов ред" може да се конвертира в последователност от два символа „връщане в началото на текущия ред" и „преминаване на нов ред". Най-често двоичният файл се използва за съхранение на структури.

Файлов ВХОД/ИЗХОД

В хедърния файл **STDIO.H**, освен входно/изходните функции, са дефинирани още и няколко макроса и няколко типа данни, използвани от файловата система. Най-важният тип е **FILE**. Той се ползва за дефиниране на указател към файл.

Други два типа са:

size_t = дефинира обект, който съдържа размера на най-големия файл, разрешен от операционната система;

fpos_t = дефинира обект, който съдържа информация, определяща позицията във файла.

Типът **FILE** се представлява от структура, която на практика идентифицира файла. Тази структура се състои от група променливи, достъпни чрез едно общо име. Всяка от променливите отговаря за съдържанието на различна информация за файла, като размера на файла, текущото му местонахождение, режима на достъп до него и др.

Файлов ВХОД/ИЗХОД

В/И система на С се основава на идеята за изграждане на абстрактно ниво на връзката между програмиста и обслужващия хардуер. В основата на тази връзка стои понятието «**поток**».

Под **интерфейс** ще разбираме всяка съвкупност от необходими средства, които осигуряват ефективен обмен на информация между различни по тип устройства.

ПОТОК наричаме абстрактния интерфейс, който логически опосредства връзката към конкретен файл в една С-програма.

Логистиката на потока има за цел да «**скрива**» хардуера от погледа на програмиста и като символ на какъв да е хардуера да му предоставя едно единствено и много просто и лесно за управление нещо: ПОТОК. Всеки поток, свързан с файл, се ползва от специално дефинирания (в **stdio.h**) за целта тип структура за управление на файла, наречен **FILE**.

Деклариране на поток (файлов указател):

FILE *****stream**;**

stream = *име на* файловия указател, инициращ поток.

Файлов ВХОД/ИЗХОД

Отварянето (или създаването) на файл **filename** за работа с него и асоциирането му с поток се осъществява посредством **функция за отваряне fopen()**:

Асоцииране на файл с поток:

FILE *fopen(char *filename, char *filemode);

filename = *име на* файла.

filemode = режим на достъп до файла.

Функцията **fopen()** връща указател към структурата, асоциирана с файла **filename** при процеса на отварянето му (или **NULL** при грешка). При това файлът се позиционира в началото си (индикаторът на позицията се инициализира). След това всяка операция за четене или запис води до нарастване на индикатора за позицията. **EOF** се използва за маркиране край на файл и има стойност **-1**. Параметърът **filemode** определя режима на работа с файла:

"r" = отваря текстов файл за четене,

"r+" = отваря текстов файл за четене /запис,

"rb" = отваря двоичен файл **за** четене,

"rb+" = отваря двоичен файл **за** четене/запис,

Файлов ВХОД/ИЗХОД

"w" = създава текстов файл за запис,
"w+" = създава текстов файл за четене /запис,
"wb" = създава двоичен файл за запис,
"wb+" = създава двоичен файл за четене/ запис,
"a" = добавя към текстов файл,
"a+" = добавя към или създава текстов файл за четене /запис,
"ab" = добавя към двоичен файл,
"ab+" = добавя към **или** създава двоичен файл за четене /запис.

Освобождаване на файл от поток:
int fclose(FILE *stream);
stream = *име на* файловия указател, инициращ поток.

Освобождаването на потока от връзката му с файла се осъществява посредством **функция за затваряне fclose().** Тази функция затваря отворения файл, асоцииран с потока **stream** и изчиства буфера му. Връща **0** при коректно затваряне или **EOF** при грешка.

Примери

```
FILE *fp; //декларира файлов указател fp (поток)
char myfile[80]; //декларира символен низ myfile[80]
fp=fopen(myfile,"r"); //отваря myfile за четене и го асоциира с fp
if(fp==NULL) //проверка да не е върнат нулев указател
{cout<<"Error !\n";exit(1);}
fclose(fp);
```

Стандартни функции за ВХОД/ИЗХОД

int fgetc(FILE *stream);

Чете следващия символ (байт) от текущата позиция на потока **stream** и увеличава позиционния файлов индикатор. Връща прочетения символ (като го преобразува в цяло число) или **EOF** при достигане край на файл или грешка.

int fputc(char ch, FILE *stream);

Записва символа ch в текущата позиция на потока stream и увеличава позиционния файлов индикатор. Връща записания символ ch или EOF при грешка.

char *fgets(char *s, int n, FILE *stream);

Чете символи от входния файл по потока **stream** в низа **s**, докато достигне край на ред, край на файл или след като прочете **n-1** символа. Връща низа **s** или **NULL** при достигане край на файл или грешка.

int fputs(char *s, FILE *stream);

Записва всички символи от низа **s** във файла **stream** без нулевия символ. Връща последния записан символ или **EOF** при грешка.

Стандартни функции за ВХОД/ИЗХОД

int fscanf (FILE *stream, "*параметри за формат*", *списък аргументи*);
Чете символи от входния файл stream, преобразува ги според *параметри за формат* и запазва стойностите на адресите на *аргументите*. Връща броя на прочетените данни или **EOF** при грешка или край на файл.

int fprintf(FILE *stream, "*параметри за формат*", *списък аргументи*);
Записва *аргументите* в изходния файл stream според *параметри за формат* . Връща броя на изпратените символи към изходния файл или **EOF** при грешка.

int fwrite(void *ptr, unsigned size, int count, FILE *stream);
Записва **count** на брой данни от адрес **ptr** в изходния поток **stream**, като всяка данна е с размер **size** байта. Връща броя на записаните данни, който трябва да бъде равен на **count**.

int fread(void *ptr, unsigned size, int count, FILE *stream);
Чете **count** на брой данни от входния поток **stream** и ги запазва на адрес **ptr**, като всяка данна е с размер **size** байта. Връща броя на прочетените данни, който може да бъде по-малък от **count**, ако е достигнат край на файл или **0** при грешка.

Стандартни функции за ВХОД/ИЗХОД

int fseek(FILE *stream, long int offset, int wherefrom);

Извършва директен достъп в отворения поток stream, като позиционира файловия указател в зависимост от стойността на **offset**, която определя адресното отместване в байтове (за двоични потоци), спрямо точката на търсене wherefrom. Връща 0 или ненулева стойност при грешка

Точката на търсене wherefrom може да бъде:
SEEK_SET или **0** = **начало**
SEEK_CUR или **1** = **текуща позиция**
SEEK_END или **2** = **край**

long int ftell(FILE *stream);

Връща текущата позиция на отворения поток stream или **-1L** при грешка.

void rewind(FILE *stream);

Позиционира потока stream в началото му.

Стандартни функции за ВХОД/ИЗХОД

void remove(FILE *stream);
Изтрива потока **stream**.

int fflush(FILE *stream);
Изпразва всички буфери, свързани с изхода или обновява аргументите, потокът остава отворен. Връща **0** или **EOF** при грешка.

int feof(FILE *stream);
Връща 0, ако не е достигнат край на файл или ненулева стойност при достигане край на файл.

```
#include <stdio.h>
#include <stdlib.h>
int main ()
{  FILE *fp;
  fp=fopen("test.txt", "r");/* Отваря текстов файл за четене*/
  if(fp==NULL)    /* Проверка за съществуване на файла      */
  { printf("Файлът не може да бъде отворен.\n");
    exit(1);          /* Прекъсва изпълнението на програмата*/
   }
/* Обработка на компонентите на файла          */
  fclose(fp);    /*  Затваря на файла       */
  return 0;
}
```

Стандартни функции за ВХОД/ИЗХОД - примери

```
#include <stdio.h>
#include <stdlib.h>
int main ()
{
  int ch;    FILE *fp;
  fp=fopen("test.txt", "r");
  if(fp==NULL)
  {printf("Файлът не може да бъде отворен.\n");
    exit(1);}
  while((ch=fgetc(fp))!=EOF) putchar(ch);
  fclose(fp);
  return 0;
}
```

Отваря текстов файл за четене

Четене на символи от отворения файл

```
#include <stdio.h>
#include <stdlib.h>
int main ()
{
  int ch;    FILE *fp;
  fp=fopen("test.txt", "w");
  if(fp==NULL)
  {printf("Файлът не може да бъде отворен.\n");
    exit(1);}
  while((ch=getchar())!=EOF) fputc(ch,fp);
  fclose(fp);
  return 0;
}
```

Отваря текстов файл за запис

Запис на символи в отворения файл

Стандартни функции за ВХОД/ИЗХОД - примери

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main ()
{  int ch;   FILE  *fp;   char s[81];
   fp=fopen("test.txt", "r");
   if(fp==NULL)
   {printf("Файлът не може да бъде отворен.\n");
    exit(1);}
   while(fgets(s, 80, fp) != NULL)  puts(s);
   fclose(fp);
   return 0;
}
```

Отваря текстов файл за четене

Четене на символен низ от отворения файл

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main ()
{ int ch;  FILE  *fp;   char s[81];
   fp=fopen("test.txt", "w");
   if(fp==NULL)
   {printf("Файлът не може да бъде отворен.\n");
    exit(1);}
while(gets(s)!=NULL) {strcat(s,"\n"); fputs(s,fp);}
   fclose(fp);
   return 0;
}
```

Отваря текстов файл за запис

Запис на символи в отворения файл

Стандартни функции за ВХОД/ИЗХОД - примери

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main ()
{  int ch, x;   FILE  *fp;
   fp=fopen("test.txt", "r");
   if(fp==NULL)
   {printf("Файлът не може да бъде отворен.\n");
     exit(1);}
   while(fscanf(fp,"%d",&x)==1) printf("%d\n",x);
   fclose(fp);
   return 0;
}
```

Отваря файл за четене

Четене на числови данни от отворения файл

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main ()
{ int ch, x;  FILE  *fp;
   fp=fopen("test.txt", "w");
   if(fp==NULL)
   {printf("Файлът не може да бъде отворен.\n");
     exit(1);}
while(scanf("%d",&x)==1) fprintf(fp,"%d\n",x);
   fclose(fp);
   return 0;
}
```

Отваря файл за запис

Запис на числови данни в отворения файл

Стандартни функции за ВХОД/ИЗХОД - примери

```
#include <stdio.h>
#include <stdlib.h>
int sum(FILE *fp, char *filename);

int main ()
{   FILE *text;
printf("Sum=%d\n", sum(text,"test.txt"));
return(0);
}

int sum(FILE *fp, char *filename)
{  int x, s=0;
 fp=fopen(filename,"r");
 if(fp==NULL)
{printf("Файлът не може да бъде отворен.\n"); exit(1);}
while(fscanf(fp,"%d",&x)==1)    s+=x;
fclose(fp);
return(s);
}
```

Горната програма извиква функията **sum (text,"test.txt")** за да реализира събирането на всички числа от файла "**test.txt**". Тази извикана функция първо инициализира променлива за сумата (**s=0;**), след това отваря файла за четене и проверява дали при отварянето няма някаква грешка. Ако всичко е ок, докато не се стигне край на файла, се четат компонентите му (числата, разположени по редове) и се натрупват, събирайки се върху s. Накрая файлът се затваря и с инструкцията **return (s);** резултата се предава на извикващата функция **main()** за отпечатване и приключване на програмата.

Стандартни функции за ВХОД/ИЗХОД - примери

```
#include <stdio.h>
#include <stdlib.h>
int main ()
{
FILE *f1, *f2;  char ch;
f1=fopen("newtest.txt","w");
f2=fopen("test.txt","r");
if(f1==NULL || f2==NULL)
{ printf("Грешка!\n");  exit(1);  }
while((ch=fgetc(f2)) != EOF)  fputc(ch, f1);
fclose(f1);  fclose(f2);
}
```

```
#include <stdio.h>
#include <stdlib.h>
int main ()
{
FILE *fp;  long int pos;
fp=fopen("test.txt","r+b");
if(fp==NULL)
{printf("Файлът не може да бъде отворен.\n");
exit(1);}
fseek(fp,0,SEEK_END);
pos=ftell(fp);
fclose(fp);
printf("Дължина на файла =" "%ld [байта].\n", pos);
}
```