

Strumienie i pliki.

- **Plik** - ciąg bajtów o skończonej długości
- Nawa pliku nie stanowi jego części
- Położenie pliku określone przez ścieżkę dostępu
- Pliki są opatrzone **atrybutami**: uprawnienia, własności pliku, np. plik ukryty, itp.
- Plik tekstowy a plik binarny
- W Unix/Linux wszystko jest plikiem

<http://pl.wikipedia.org/wiki/Plik>

Format pliku: określa rodzaj i sposób zapisu danych w pliku.

- Rozszerzenia plików (DOS, Winows)
txt html c csv bmp mp3 jpg
- Metadane zawarte w pliku, sygnatura formatu zakodowana najczęściej na początku pliku: nagłówek pliku, liczba magiczna
FF D8 FF dla formatu jpg
D0 CF 11 E0 dokumenty MS Office
- Metadane zewnętrzne, np. umieszczone w systemie plików.
Typy **MIME**, Multipurpose Internet Mail Extensions
Content-Type: text/plain
Content-Type: audio/mpeg:

http://en.wikipedia.org/wiki/List_of_file_signatures

http://en.wikipedia.org/wiki/File_format

System plików: sposób przechowywania plików na nośniku.

- fizyczny zapis danych, blokowa struktura danych (sektory, klastry)
- logiczna struktura widoczna dla użytkownika
- DOS/Windows: FAT, FAT32, NTFS
- Linux: ext2, ext3, ext4
- inne: HFS (Mac OS), NFS (sieć), ISO9660 (CD-ROM)
- Hierarchia systemów plików: katalogi, podkatalogi i pliki.

Dostęp do pliku: ścieżka + nazwa pliku

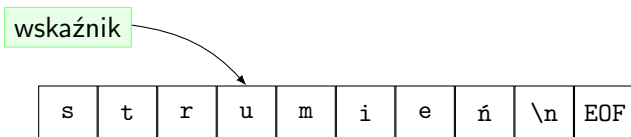
C:\Documents and Settings\user\moje dokumenty\plik.txt
/home/user/doc/plik.txt

http://en.wikipedia.org/wiki/List_of_file_systems

- nazwa pliku
- rozmiar
- data utworzenia, data modyfikacji, dostępu
- uprawnienia: właściciel, grupy i ich prawa (odczyt/zapis)
- DOS/Windows: ukryty, tylko do odczytu, archiwalny, systemowy, zaszyfrowany (NTFS), skompresowany (NTFS)
- rodzaje plików: plik zwykły, katalog, dowiązanie, FIFO, blokowe, ...
- położenie pliku, wskaźnik do miejsca na nośniku (często tablica wskaźników FAT), Linux i-węzły

- W C nie ma typu plikowego
- Pascal: pliki tekstowe, amorficzne (niezredagowane), pliki zdefiniowane (jednorodne)
- **Deskryptor pliku:** liczba całkowita, niskopoziomowa reprezentacja
Potrzebne funkcje systemowe, np. biblioteka `unistd.h`
- **Strumień:** ogólny sposób komunikacji między plikami, urządzeniami i procesami
- Wykorzystanie strumieni: dostęp do plików, komunikacja między procesami, komunikacja sieciowa, komunikacja z urządzeniami, łańcuchy jako strumień.
- Plik nagłówkowy `stdio.h`, obsługa wejścia i wyjścia (***Standard input output***)

- Strumień to zmienna typu FILE*
wskaźnik do pliku, „uchwyt” do pliku
- Struktura zawierająca informacje o pliku: nazwa, deskryptor pliku, rodzaj dostępu, znacznik pozycji, adres bufora, ...
- Strumień udostępnia dane jako sekwencję bajtów zakończoną wartością końca pliku EOF
- Sekwencyjny odczyt i zapis, automatyczne buforowanie
- Wskaźnik bieżącej pozycji - miejsce odczytu/zapisu
- Obsługa błędów i końca pliku



1. Deklaracja zmiennej typu FILE*

```
FILE* plik;
```

2. Uzyskanie dostępu do strumienia fopen()

```
plik = fopen("plik.txt", "w");
```

3. Odczyt, zapis lub zmiana pozycji w strumieniu, np.:

```
fprintf(plik, "Hello world!\n");  
fscanf(plik2, "%d", &x);
```

4. Zamknięcie strumienia fclose()

```
fclose(plik);
```

Nie zapomnij o obsłudze błędów.
Każda operacja na pliku może zakończyć się niepowodzeniem.

```
FILE *fopen(char *ścieżka, char *tryb);
```

- ścieżka do pliku:
 - względem bieżącego katalogu "dane.xml", "../plik.txt"
 - bezwzględna: "C:\\plik.txt", "/home/user/plik". Nie najlepszy pomysł.
- tryb otwarcia:
 - "w" zapis (*write*), plik powstaje od początku
 - "r" odczyt (*read*) pliku istniejącego
 - "a" dopisanie (*append*) na końcu pliku istniejącego
- tryb binarny: "rb", "wb", "ab". W systemie Linux nie ma różnicy pomiędzy trybem tekstowym i binarnym.

```
FILE *odczyt, *zapis;  
odczyt = fopen("plik1.txt", "r");  
zapis = fopen("plik2.txt", "w");
```

BŁĘDY PRZY OTWIERANIU STRUMIENIA

- W przypadku niepowodzenia `fopen()` zwraca wartość `NULL`
- Wówczas nie ma możliwości wykonania jakiegokolwiek operacji na strumieniu
- Zawsze sprawdzaj czy udało się uzyskać dostęp do strumienia

```
FILE *plik;  
plik=fopen("plik.txt","r");  
if ( plik != NULL ) { /* ... */ }
```

- Możliwe przyczyny: zła nazwa pliku, zła ścieżka, brak nośnika, uszkodzenie nośnika, brak uprawnień do odczytu lub zapisu, za duża liczba otworzonych strumieni, itp...

FORMATOWANY ZAPIS I ODCZYT

```
int fprintf(FILE *plik, char *format, ...);  
int fscanf(FILE *plik, char *format, ...);
```

ZAPIS I ODCZYT POJEDYNCZEGO BAJTU (ZNAKU)

```
int fgetc(FILE *plik);  
int fputc(int c, FILE *plik);
```

ODCZYT ŁAŃCUCHA

```
char *fgets(char *tablica, int rozmiar, FILE *strumien);
```

- Wartość zwracana może informować o błędach lub końcu pliku
- Odczyt i zapis strumieni binarnych: `fread`, `fwrite`
- Więcej informacji w dokumentacji biblioteki `stdio.h`

TYPOWY SCHEMAT: ZAPIS DO PLIKU

```
1  #include<stdio.h>
2
3  int main()
4  {
5      FILE *plik = NULL;
6      float pi = 3.1415;
7
8      plik = fopen( "plik.txt", "w" );
9      if(plik != NULL )
10     {
11         /* Tutaj operacje na pliku */
12         fprintf(plik,"Witaj swiecie!\nPI=%f\n", pi);
13         fclose( plik );
14     }
15     else
16     {
17         /* Obsluga bledu otwarcia pliku */
18         printf("Blad otwarcia pliku %s\n", "plik.txt" );
19     }
20     return 0;
21 }
```

KONIEC STRUMIENIA

```
int feof(FILE* plik);
```

Wartość niezerowa gdy wystąpił koniec pliku.

Funkcje odczytu: `fscanf()`, `fgetc()`, `fgets()` zwracają EOF

INNE BŁĘDY OPERACJI WEJŚCIA-WYJŚCIA

```
int ferror(FILE *stream);
```

Wartość niezerowa gdy wystąpił błąd.

TYPOWY SCHEMAT: ODCZYT ZNAKÓW Z PLIKU

```
1  #include<stdio.h>
2
3  int main()
4  {
5      FILE *plik = NULL;
6      int znak;
7
8      plik = fopen( "plik.txt", "r" );
9      if( plik == NULL )
10     {
11         perror("Wystapil blad");
12         return 1;
13     }
14
15     while( feof(plik) == 0 )
16     {
17         znak = fgetc(plik);
18         if (znak != EOF) printf("%c\n", znak);
19     }
20     fclose( plik );
21
22     return 0;
23 }
```

STANDARDOWE WEJŚCIE I WYJŚCIE PROGRAMU

```
cat < plik1.txt > plik2.txt
```

< przekierowanie strumienia wejściowego

> przekierowanie strumienia wyjściowego

POTOKI

```
ls | wc
```

- Strumień to podstawowy sposób komunikacji między procesami.
- Unix/Linux: zbiór małych narzędzi o wielkich możliwościach

Strumienie FILE* dostępne w `stdio.h`

0 `stdin`

standardowe wejście, domyślnie klawiatura
`getchar()`, `scanf()`, ...

1 `stdout`

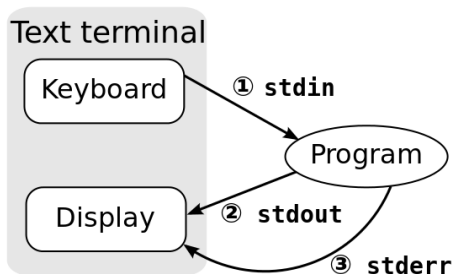
standardowe wyjście, domyślnie ekran
`putchar()`, `printf()`, ...

2 `stderr`

standardowe wejście diagnostyczne, domyślnie ekran
`perror()`, ...

```
z=getchar();  
putchar(z);  
printf("x=%d\n",x);  
scanf("%f",&y);  
gets(tab);
```

```
z=fgetc(stdin);  
fputc(z, stdout);  
fprintf(stdout, "x=%d\n",x);  
fscanf(stdin, "%f", &y);  
fgets(tab, n, stdin);
```



Źródło: <http://upload.wikimedia.org/>

- DOS/Windows: konwersja plików tekstowych
Nowy wiersz: `\r\n` $\iff$ `\n`
- Niepoprawny format danych dla `scanf()`, `fscanf()`
Wartość zwracana to ilość wczytanych elementów.

```
if( scanf("%d", &x) > 0 ) { /* OK */ }
```

- Ostrożnie z mieszaniem odczytu formatowanego z nieformatowanym

```
scanf("%d", &x);  
getchar();      /* wczyta co zostawił scanf() */
```

- Jednoczesny zapis i odczyt może wymagać dodatkowych zabiegów (np. czyszczenie bufora)
- W Unix/Linux wielkość liter w ścieżkach ma znaczenie
`plik.txt` `PLIK.TXT` `Plik.txt` `Plik.TXT`
- W razie niepewności zajrzyj do dokumentacji

- 🌐 Wikipedia: [👉 Plik](#), [👉 List of file signatures](#), [👉 File format](#).
[👉 System plików](#), [👉 List of file systems](#), [👉 Comparison of file systems](#), [👉 Standardowe strumienie](#)
- 🌐 The GNU C Library: [👉 Input/Output Overview](#),
[👉 Input/Output on Streams](#)