



Fundusze Europejskie
dla Rozwoju Społecznego



Rzeczpospolita
Polska

Dofinansowane przez
Unię Europejską



Informatyka 1 (EMS1A1004)

Politechnika Białostocka - Wydział Elektryczny
Elektromobilność, semestr I, studia stacjonarne I stopnia

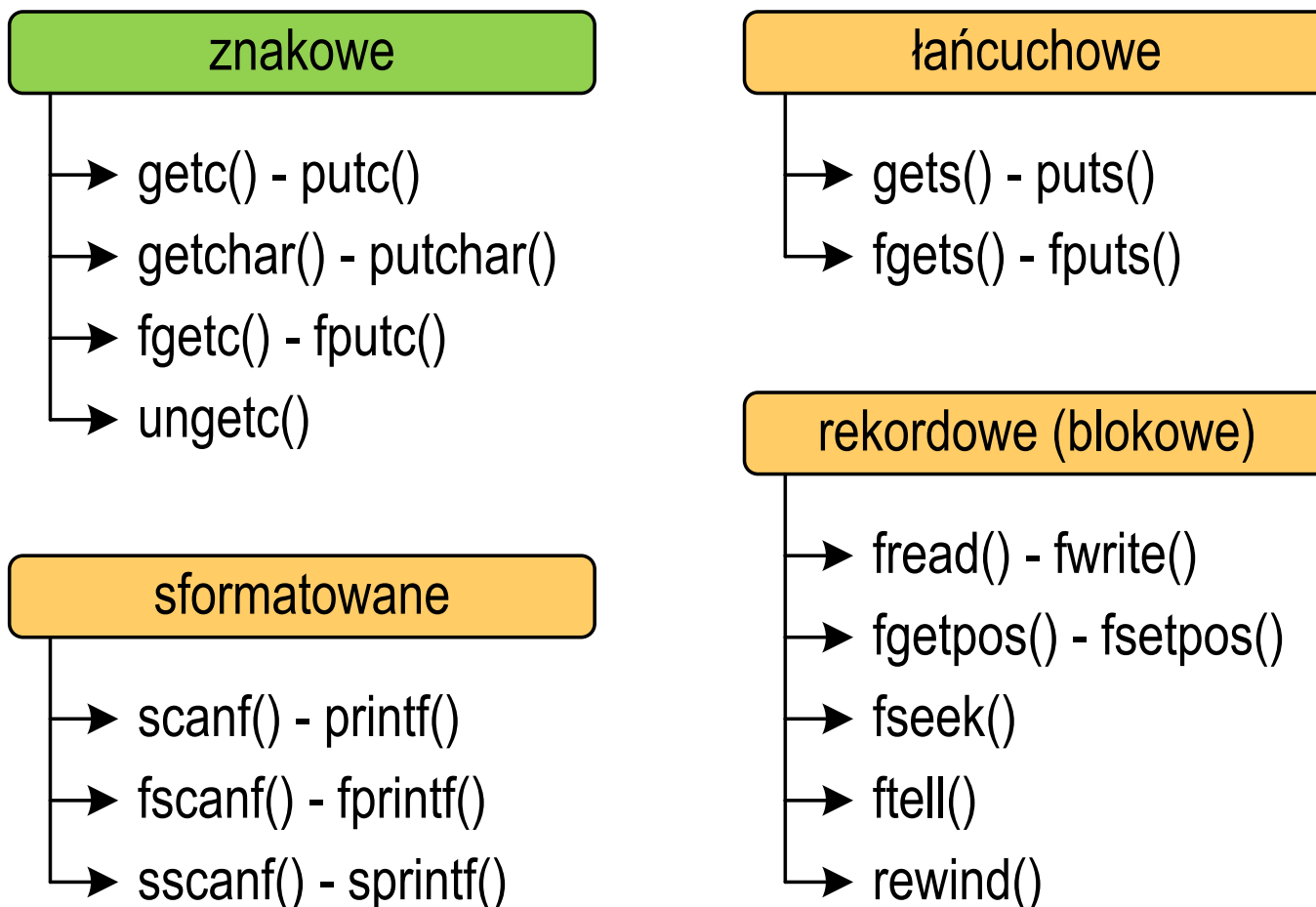
Wykład nr 14

dr inż. Jarosław Forenc

Plan wykładu nr 14

- Operacje znakowe
- Operacje łańcuchowe
- Operacje sformatowane
- Operacje rekordowe (blokowe)

Znakowe operacje wejścia-wyjścia



Znakowe operacje wejścia-wyjścia

```
int getc(FILE *fp) ;
```

```
int fgetc(FILE *fp) ;
```

- pobiera (czyta) jeden znak ze strumienia **fp** i zwraca jego kod (jako **int**) lub **EOF** (gdy napotkano koniec pliku)

```
int getchar() ;
```

- pobiera (czyta) jeden znak z klawiatury (strumienia **stdin**) i zwraca jego kod (jako **int**)

```
FILE *fp; int zn;
```

```
zn = getc(fp) ;           // z pliku
```

```
zn = fgetc(fp) ;          // z pliku
```

```
zn = getchar() ;          // z klawiatury
```

Przykład: wyświetlenie pliku tekstowego

```
#include <stdio.h>

int main(void)
{
    FILE    *fp;
    int      znak;

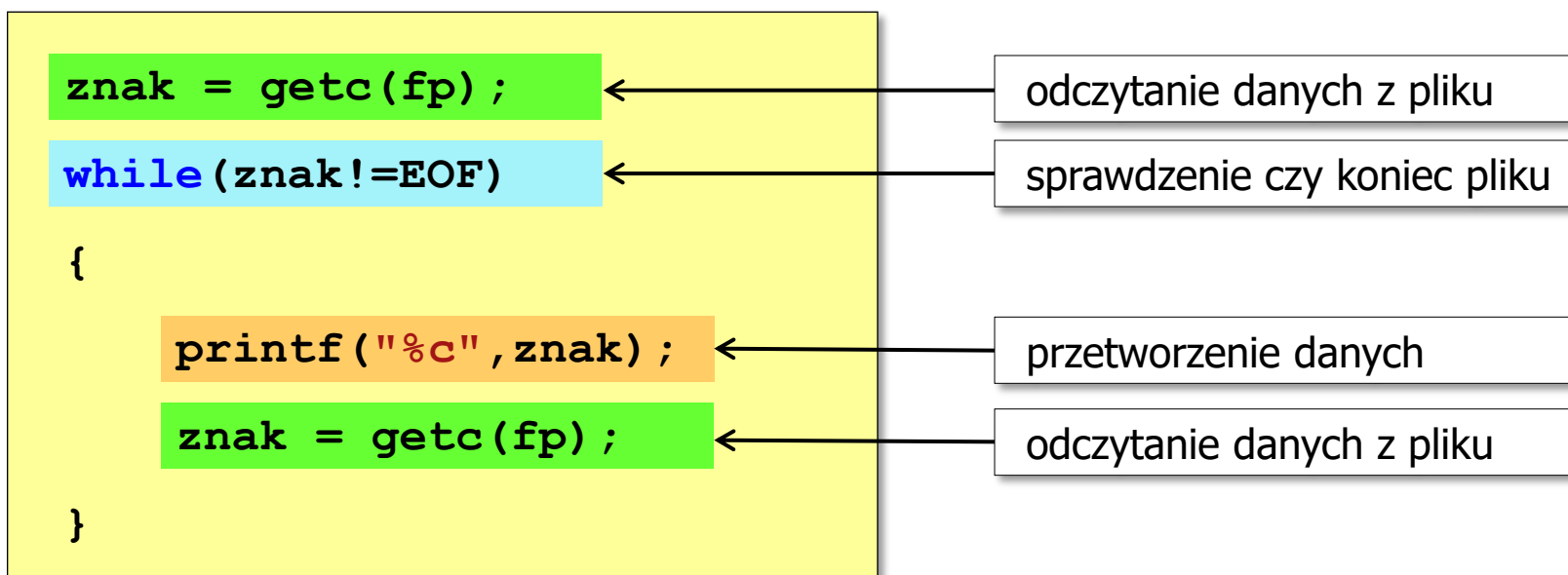
    fp = fopen("test.txt", "r");

    znak = getc(fp);
    while (znak != EOF)
    {
        printf("%c", znak);
        znak = getc(fp);
    }

    fclose(fp);
    return 0;
}
```

Schemat przetwarzania pliku

■ Typowy schemat odczytywania danych z pliku



Przykład: wyświetlenie pliku tekstowego

- Odczytanie i wyświetlenie zawartości pliku tekstowego

```
znak = getc(fp);  
while (znak != EOF)  
{  
    printf("%c", znak);  
    znak = getc(fp);  
}
```

można zapisać w krótszej postaci:

```
while ( (znak=getc(fp)) != EOF)  
    printf("%c", znak);
```

Przykład: liczba wyrazów w pliku

```
#include <stdio.h>

int main(void)
{
    FILE *fp;
    int znak, odstep = 1, ile = 0;

    fp = fopen("test.txt", "r");
    while ((znak = fgetc(fp)) != EOF)
        if (znak == ' ' || znak == '\t' || znak == '\n')
            odstep = 1;
        else
            if (odstep != 0) { odstep = 0; ile++; }
    fclose(fp);
    printf("Liczba wyrazow: %d\n", ile);

    return 0;
}
```

Ala ma laptopa i psa.

Liczba slow: 5

Znakowe operacje wejścia-wyjścia

```
int putc(int znak, FILE *fp);
```

```
int fputc(int znak, FILE *fp);
```

- wpisuje **znak** do otwartego strumienia **fp**

```
int putchar(int znak);
```

- wyświetla **znak** na ekranie (wpisuje do strumienia **stdout**)

```
FILE *fp; int zn = 'a';  
putc(zn, fp);           // do pliku  
fputc(zn, fp);          // do pliku  
putchar(zn);            // na ekran
```

Przykład: zapisanie alfabetu do pliku tekstowego

```
#include <stdio.h>

int main(void)
{
    FILE *fp = fopen("alfabet.txt", "w");

    for (int i='A'; i<='Z'; i++)
        putc(i, fp);

    fclose(fp);

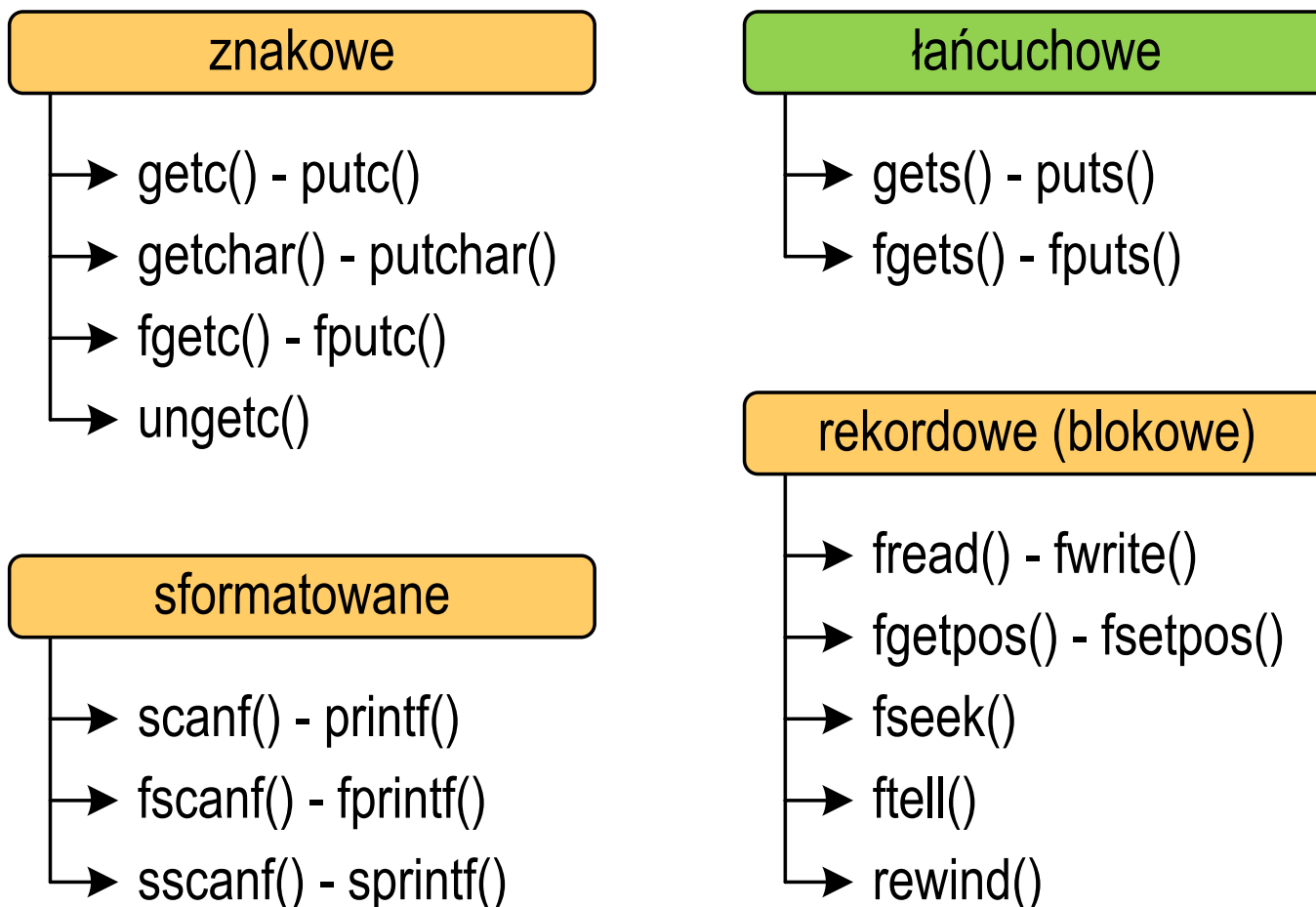
    return 0;
}
```

ABCDEFGHIJKLMNOPQRSTUVWXYZ

- Stosując strumień **stdout** można wyświetlić alfabet na ekranie

```
for (int i='A'; i<='Z'; i++)
    putc(i, stdout);
```

Łańcuchowe operacje wejścia-wyjścia



Łańcuchowe operacje wejścia-wyjścia

```
char* gets(char *buf);
```

- czyta linię znaków z klawiatury (strumienia `stdin`) i zapisuje w tablicy `buf`; wczytywanie jest kończone po napotkaniu `'\n'`, który zastępowany jest znakiem `'\0'`

```
char* fgets(char *buf, int max, FILE *fp);
```

- czyta znaki z otwartego strumienia `fp` i zapisuje je w tablicy `buf`; przerywa pobieranie znaków po odczytaniu `'\n'` lub `max-1` znaków; zwraca `NULL` po napotkaniu końca pliku

```
FILE *fp; char txt[20];  
gets(txt);           // z klawiatury  
fgets(txt, 20, fp);  // z pliku
```

Łańcuchowe operacje wejścia-wyjścia

```
int puts(const char *buf);
```

- wyświetla łańcuch znaków `buf` na ekranie (wpisuje do strumienia `stdout`);
zastępuje znak `'\0'` znakiem `'\n'`

```
int fputs(const char *buf, FILE *fp);
```

- wpisuje znaki z tablicy `buf` do otwartego strumienia `fp`;
nie dołącza znaku końca wiersza `'\n'`

```
FILE *fp; char txt[20] = "Witaj swiecie";  
puts(txt);           // na ekran  
fputs(txt, fp);      // do pliku
```

Przykład: wyświetlenie pliku tekstowego

```
#include <stdio.h>

int main(void)
{
    FILE    *fp;
    char    buf[15];

    fp = fopen("test.txt", "r");

    while (fgets(buf, 15, fp) != NULL)
        fputs(buf, stdout);

    fclose(fp);

    return 0;
}
```

Przykład: wyświetlenie pliku tekstowego

- Zawartość pliku **test.txt**

```
Poprzednikiem języka C CR LF  
był język B, CR LF  
który CR LF  
Ritchie rozwinał w język C. CR LF
```

- Kolejne wywołania funkcji **fgets(buf,15,fp);**

```
Poprzednikiem języka C CR LF  
był język B, CR LF  
który CR LF  
Ritchie rozwinał w język C. CR LF
```

Przykład: wyświetlenie pliku tekstowego

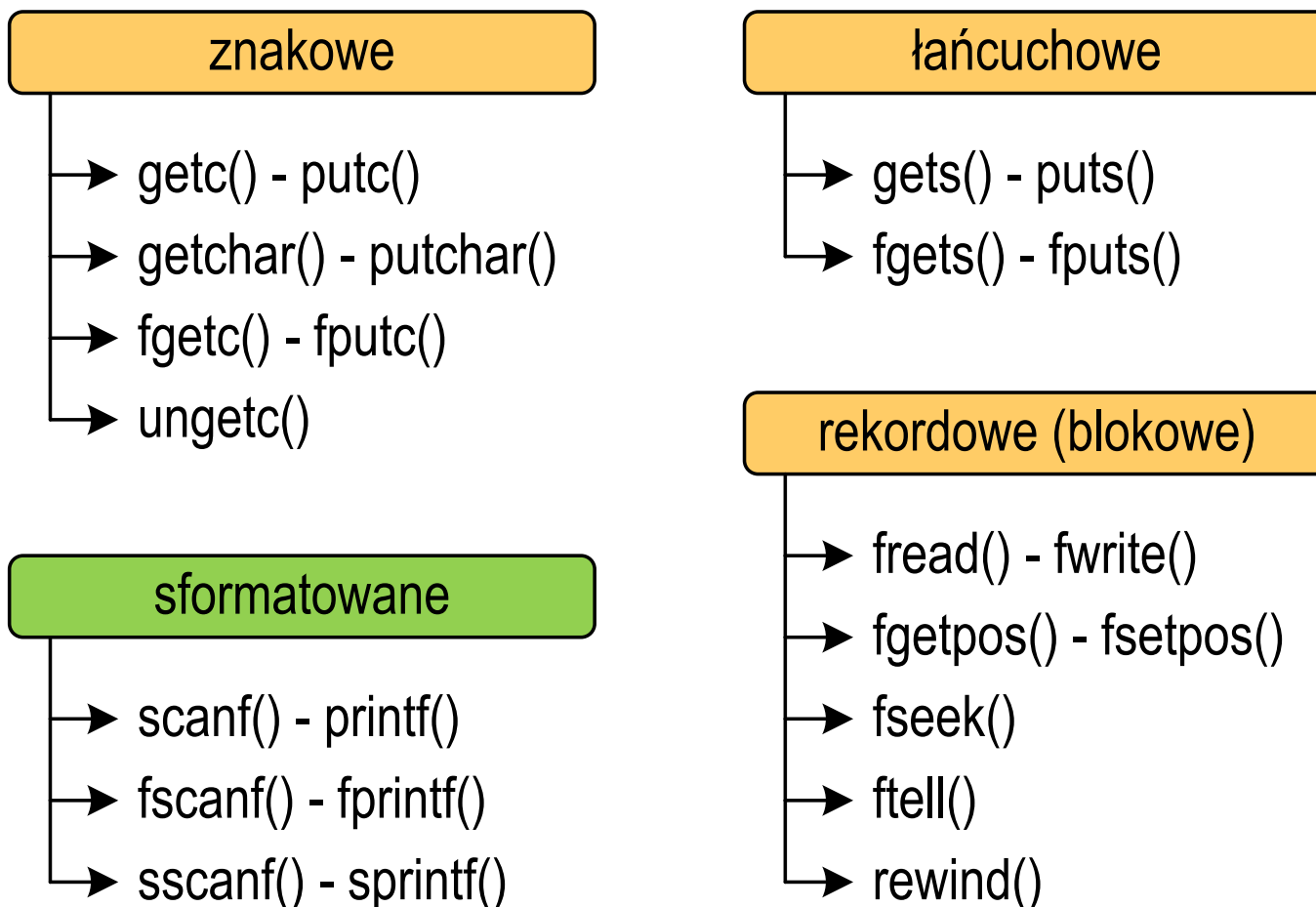
- Kolejne wywołania funkcji `fgets(buf,15,fp);` i zawartość tablicy `buf`

```
Poprzednikiem języka C CR LF  
był język B, CR LF  
ktory CR LF  
Ritchie rozwinał w język C. CR LF
```

P	o	p	r	z	e	d	n	i	k	i	e	m		\0
j	e	z	y	k	a		C	LF	\0					
b	y	l		j	e	z	y	k		B	,	LF	\0	
k	t	o	r	y	LF	\0								
R	i	t	c	h	i	e		r	o	z	w	i	n	\0
a	l		w		j	e	z	y	k		C	.	LF	\0

LF = \n

Sformatowane operacje wejścia-wyjścia



Sformatowane operacje wejścia-wyjścia

```
int printf(const char *format, ...);
```

- wyświetla dane na ekranie (wyprowadza do strumienia **stdout**)

```
int fprintf(FILE *fp, const char *format, ...);
```

- wyprowadza dane do otwartego strumienia **fp** (najczęściej pliku)

```
int sprintf(char *buf, const char *format, ...);
```

- zapisuje dane do tablicy znaków **buf**

```
FILE *fp; char txt[30];  
printf("Witaj swiecie");           // na ekran  
fprintf(fp, "Witaj swiecie");      // do pliku  
sprintf(txt, "Witaj swiecie");     // do tablicy znaków
```

Przykład: zapisanie liczb do pliku tekstowego

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(void)
{
    FILE *fp; float x; int i;

    srand((unsigned int)time(NULL));
    fp = fopen("liczby.txt", "w");
    for (i=0; i<10; i++)
    {
        x = (float)rand()/RAND_MAX*100;
        fprintf(fp, "%f\n", x);
    }
    fclose(fp);

    return 0;
}
```

```
3.830073
70.848717
99.322487
19.812616
7.132175
49.134800
10.238960
18.668173
8.914456
69.258705
```

Przykład: zapisanie danych do pliku tekstowego

```
#include <stdio.h>

int main(void)
{
    FILE *fp;
    int    wiek = 21;
    float  wzrost = 1.78f;
    char   imie[10] = "Jan", nazw[10] = "Kowalski";

    fp = fopen("dane.txt", "w");
    fprintf(fp, "Imie:      %s\n", imie);
    fprintf(fp, "Nazwisko:  %s\n", nazw);
    fprintf(fp, "Wiek:      %d [lat]\n", wiek);
    fprintf(fp, "Wzrost:    %.2f [m]\n", wzrost);
    fclose(fp);

    return 0;
}
```

Imie:	Jan
Nazwisko:	Kowalski
Wiek:	21 [lat]
Wzrost:	1.78 [m]

Sformatowane operacje wejścia-wyjścia

```
int scanf(const char *format, ...);
```

- czyta dane z klawiatury (strumienia `stdin`)

```
int fscanf(FILE *fp, const char *format, ...);
```

- czyta dane z otwartego strumienia `fp` (najczęściej pliku)

```
int sscanf(const char *buf, const char *format, ...);
```

- czyta dane z tablicy znaków `buf`

```
FILE *fp; char txt[30] = "15 3.14"; int x; float y;  
scanf("%d %f", &x, &y);           // z klawiatury  
fscanf(fp, "%d %f", &x, &y);       // z pliku  
sscanf(txt, "%d %f", &x, &y);      // z tablicy znaków
```

Przykład: odczytanie danych z pliku tekstowego

- Odczytanie danych różnych typów z pliku tekstowego

```
Nowak Grzegorz 15-12-2000  
Kowalski Wojciech 03-05-1997  
Jankowska Anna 23-05-1995  
Mazur Krzysztof 14-01-1990  
Krawczyk Monika 03-11-1995  
Piotrowska Maja 12-06-1998  
Dudek Piotr 31-12-1996  
Pawlak Julia 01-01-1997
```

Grzegorz	Nowak	wiek: 25
Wojciech	Kowalski	wiek: 28
Anna	Jankowska	wiek: 30
Krzysztof	Mazur	wiek: 35
Monika	Krawczyk	wiek: 30
Maja	Piotrowska	wiek: 27
Piotr	Dudek	wiek: 29
Julia	Pawlak	wiek: 28

Wykrycie końca pliku

Funkcje	Metoda
getc(), fgetc()	zwracana wartość: EOF
fgets()	zwracana wartość: NULL
fscanf()	wywołanie funkcji: feof()

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

- zwraca wartość różną od zera, jeśli podczas ostatniej operacji odczytu pliku wskazywanego przez **fp** został wykryty jego koniec; w przeciwnym razie zwraca wartość **0** (zero)

Przykład: odczytanie danych z pliku tekstowego

```
#include <stdio.h>

int main()
{
    FILE *fp;
    char naz[20], im[20];
    int d, m, r;

    fp = fopen("osoby.txt", "r");
    fscanf(fp, "%s %s %d-%d-%d", naz, im, &d, &m, &r);
    while (!feof(fp))
    {
        printf("%-12s %-12s wiek: %d\n", im, naz, 2025-r);
        fscanf(fp, "%s %s %d-%d-%d", naz, im, &d, &m, &r);
    }
    fclose(fp);

    return 0;
}
```


Przykład: odczytanie danych z pliku tekstowego

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    FILE *fp;
```

```
    char naz[20], im[20];
```

```
    int d, m, r;
```

```
    fp = fopen("osoby.txt", "r");
```

```
    fscanf(fp, "%s %s %d-%d-%d", naz, im, &d, &m, &r);
```

```
    while (!feof(fp))
```

```
    {
```

```
        printf("%-12s %-12s wiek: %d\n", im, naz, 2025-r);
```

```
        fscanf(fp, "%s %s %d-%d-%d", naz, im, &d, &m, &r);
```

```
    }
```

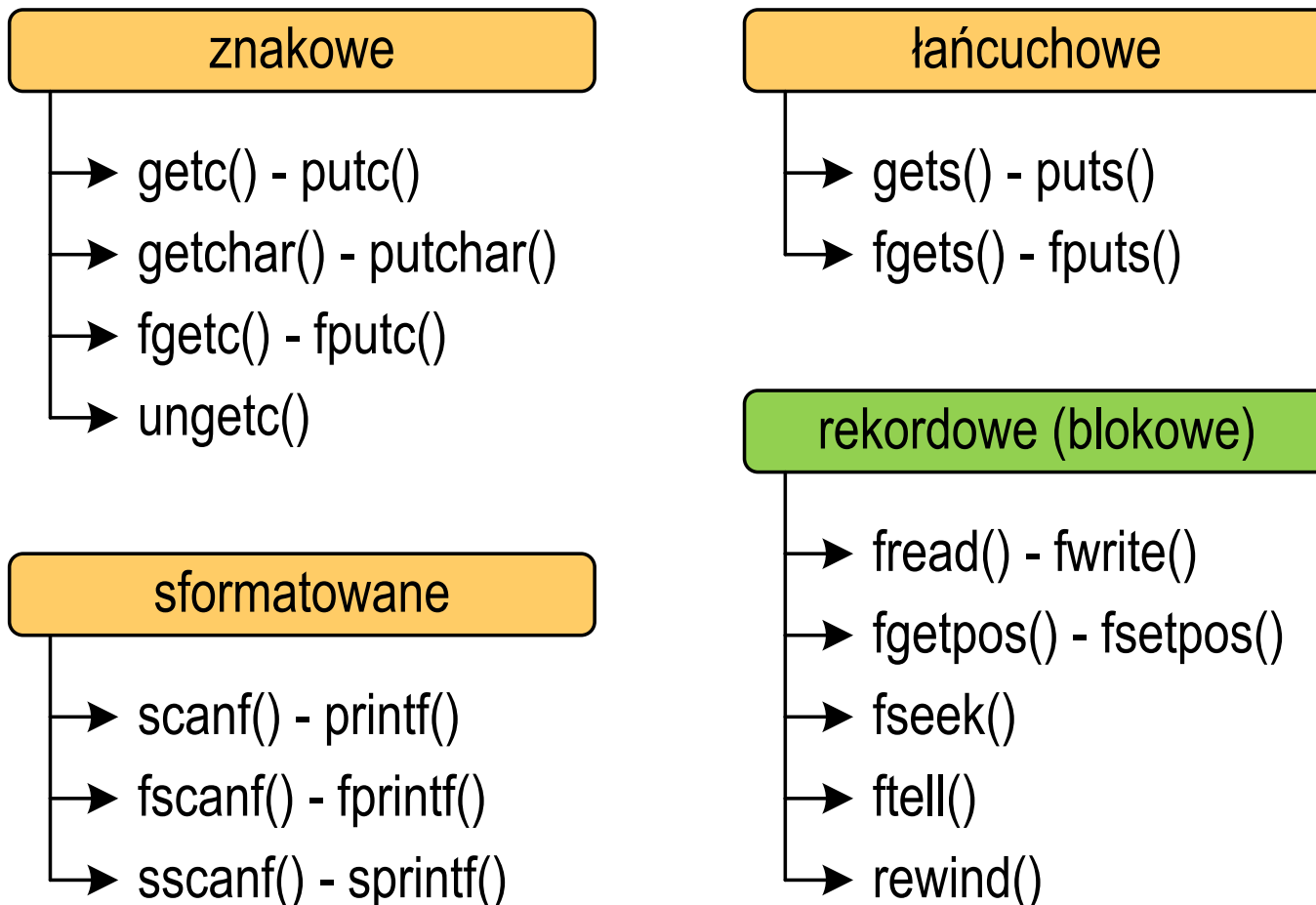
```
    fclose(fp);
```

```
    return 0;
```

```
}
```

Grzegorz	Nowak	wiek: 25
Wojciech	Kowalski	wiek: 28
Anna	Jankowska	wiek: 30
Krzysztof	Mazur	wiek: 35
Monika	Krawczyk	wiek: 30
Maja	Piotrowska	wiek: 27
Piotr	Dudek	wiek: 29
Julia	Pawlak	wiek: 28

Rekordowe (blokowe) operacje wejścia-wyjścia



Rekordowe (blokowe) operacje wejścia-wyjścia

```
size_t fwrite(const void *p, size_t s, size_t n, FILE *fp);
```

- zapisuje **n** elementów o rozmiarze **s** bajtów każdy, do pliku określanego przez **fp**, biorąc dane z obszaru pamięci wskazywanego przez **p**
- zwraca liczbę faktycznie zapisanych elementów

```
size_t fread(void *p, size_t s, size_t n, FILE *fp);
```

- pobiera **n** elementów o rozmiarze **s** bajtów każdy, z pliku określanego przez **fp** i umieszcza odczytane dane w obszarze pamięci wskazywanym przez **p**
- zwraca liczbę faktycznie odczytanych elementów

Przykład: zapisanie danych do pliku binarnego

```
#include <stdio.h>

int main(void)
{
    FILE *fp;
    int    x = 10, tab[5] = {1,2,3,4,5};
    float  y = 1.2345f;

    fp = fopen("dane.dat", "wb");
    fwrite(&x, sizeof(int), 1, fp);
    fwrite(tab, sizeof(int), 5, fp);
    fwrite(tab, sizeof(tab), 1, fp);
    fwrite(&y, sizeof(float), 1, fp);
    fclose(fp);

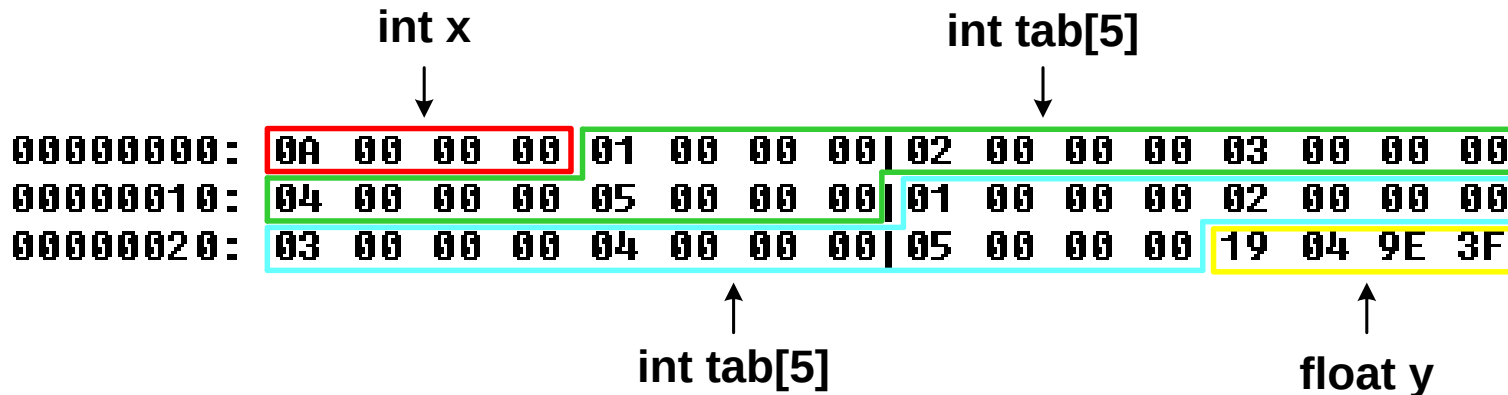
    return 0;
}
```

Przykład: zapisanie danych do pliku binarnego

- Czterokrotne wywołanie funkcji `fwrite()`

```
fwrite(&x,sizeof(int),1,fp); // int x = 10;
fwrite(tab,sizeof(int),5,fp); // int tab[5] = {1,2,3,4,5};
fwrite(tab,sizeof(tab),1,fp); // int tab[5] = {1,2,3,4,5};
fwrite(&y,sizeof(float),1,fp); // float y = 1.2345;
```

spowoduje zapisanie do pliku 48 bajtów:



Przykład: odczytanie liczb z pliku binarnego

```
#include <stdio.h>

int main(void)
{
    FILE *fp;
    int x, ile = 0;

    fp = fopen("liczby.dat", "rb");
    fread(&x, sizeof(int), 1, fp);
    while (!feof(fp))
    {
        ile++; printf("%d\n", x);
        fread(&x, sizeof(int), 1, fp);
    }
    fclose(fp);
    printf("Odczytano: %d liczb\n", ile);
    return 0;
}
```

```
37
31
83
27
6
62
31
50
Odczytano: 8 liczb
```

Przykład: odczytanie liczb z pliku binarnego

- Po otwarciu pliku wskaźnik pozycji pliku pokazuje na jego początek

↓

```
25 00 00 00 1F 00 00 00|53 00 00 00 1B 00 00 00 | %■■■■■■■■S■■■■■■■■  
06 00 00 00 3E 00 00 00|1F 00 00 00 32 00 00 00 | ■■■■>■■■■■■■■2■■■
```

- Po odczytaniu jednej liczby: `fread(&x,sizeof(int),1,plik);`
wskaźnik jest automatycznie przesuwany o `sizeof(int)` bajtów

↓

```
25 00 00 00 1F 00 00 00|53 00 00 00 1B 00 00 00 | %■■■■■■■■S■■■■■■■■  
06 00 00 00 3E 00 00 00|1F 00 00 00 32 00 00 00 | ■■■■>■■■■■■■■2■■■
```

- Po odczytaniu kolejnej liczby: `fread(&x,sizeof(int),1,plik);`
wskaźnik jest ponownie przesuwany o `sizeof(int)` bajtów

↓

```
25 00 00 00 1F 00 00 00|53 00 00 00 1B 00 00 00 | %■■■■■■■■S■■■■■■■■  
06 00 00 00 3E 00 00 00|1F 00 00 00 32 00 00 00 | ■■■■>■■■■■■■■2■■■
```

- Plik binarny zawiera liczby: 37 31 83 27 6 62 31 50

Rekordowe (blokowe) operacje wejścia-wyjścia

```
void rewind(FILE *fp);
```

- ustawia wskaźnik pozycji w pliku wskazywanym przez **fp** na początek pliku

```
long int ftell(FILE *fp);
```

- zwraca bieżące położenie w pliku (liczbę bajtów od początku pliku)

```
int fseek(FILE *fp, long int offset, int mode);
```

- przechodzi bezpośrednio do dowolnego bajtu w pliku wskazywanym przez **fp**
- **offset** - wielkość przejścia w bajtach, **mode** - punkt początkowy (**SEEK_SET** - początek pliku, **SEEK_CUR** - bieżąca pozycja, **SEEK_END** - koniec pliku)
- gdy wywołanie jest poprawne, to funkcja zwraca wartość **0**; gdy wystąpił błąd (np. próba przekroczenia granic pliku), to funkcja zwraca wartość **-1**

Przykład: odczytanie liczby o podanym numerze

```
#include <stdio.h>

int main(void)
{
    FILE *fp;
    int x, nr;

    fp = fopen("dane.dat", "rb");
    printf("Nr: "); scanf("%d", &nr);
    while (fseek(fp, (nr-1)*sizeof(int), SEEK_SET) == 0)
    {
        fread(&x, sizeof(int), 1, fp);
        printf("Liczba: %d\n", x);
        printf("Nr: "); scanf("%d", &nr);
    }
    printf("Koniec!\n");
    fclose(fp);
    return 0;
}
```

7	3	3	0	3	9	6	4	1	8
6	0	4	5	4	9	4	5	4	5
9	9	8	0	0	5	3	5	1	0

```
Nr: 6
Liczba: 9
Nr: 14
Liczba: 5
Nr: 29
Liczba: 1
Nr: -1
Koniec!
```

Rekordowe (blokowe) operacje wejścia-wyjścia

```
int fgetpos(FILE *fp, fpos_t *pos);
```

- zapamiętuję pod zmienną **pos** bieżące położenie w pliku wskazywanym przez **fp**
- zwraca **0**, gdy wywołanie jest poprawne i wartość niezerową, gdy wystąpił błąd

```
int fsetpos(FILE *fp, const fpos_t *pos);
```

- przechodzi do położenia **pos** w pliku wskazywanym przez **fp**
- zwraca **0**, gdy wywołania jest poprawne i wartość niezerową, gdy wystąpił błąd

Koniec wykładu nr 14

Dziękuję za uwagę!



Fundusze Europejskie
dla Rozwoju Społecznego



Rzeczpospolita
Polska

Dofinansowane przez
Unię Europejską



Materiały przygotowane w ramach projektu „PB 5.0 - dostosowanie oferty dydaktycznej Politechniki Białostockiej do potrzeb nowoczesnej gospodarki oraz zielonej i cyfrowej transformacji” (nr umowy FERS.01.05-IP.08-0327/23-00) w ramach programu Fundusze Europejskie dla Rozwoju Społecznego 2021-2027 współfinansowanego ze środków Europejskiego Funduszu Społecznego Plus.

Publikacja jest udostępniona na licencji Creative Commons - Uznanie autorstwa 4.0 (CC BY 4.0). Pełna treść licencji dostępna na stronie creativecommons.org.