

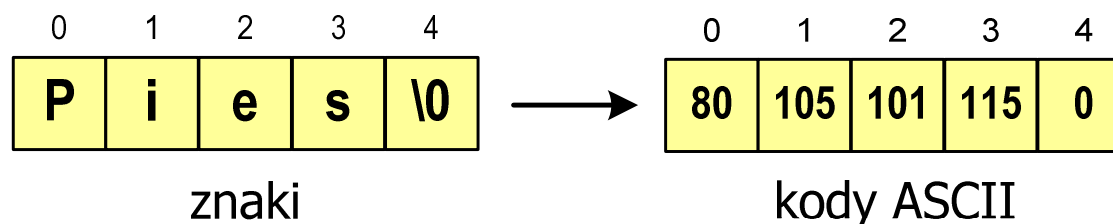
# Informatyka 2 (EZ1E3012)

---

Politechnika Białostocka - Wydział Elektryczny  
Elektrotechnika, semestr III, studia niestacjonarne I stopnia  
Rok akademicki 2022/2023

Pracownia nr 2 (15/16.10.2022)

dr inż. Jarosław Forenc



# Język C - deklaracja łańcucha znaków

- Deklaracja zmiennej przechowującej łańcuch znaków

```
char nazwa_zmiennej[rozmiar];
```

Przykład:

```
char txt[10];
```

- Tablica **txt** może przechowywać napisy o maksymalnej długości do 9 znaków

- Inicjalizacja łańcucha znaków

```
char txt1[10] = "Pies";  
char txt2[10] = {'P', 'i', 'e', 's'};  
char txt3[10] = {80, 105, 101, 115};
```

P	i	e	s	\0	\0	\0	\0	\0	\0
---	---	---	---	----	----	----	----	----	----

## Język C - stała znakowa

- **Stałą znakową** tworzy jeden znak ujęty w apostrofy

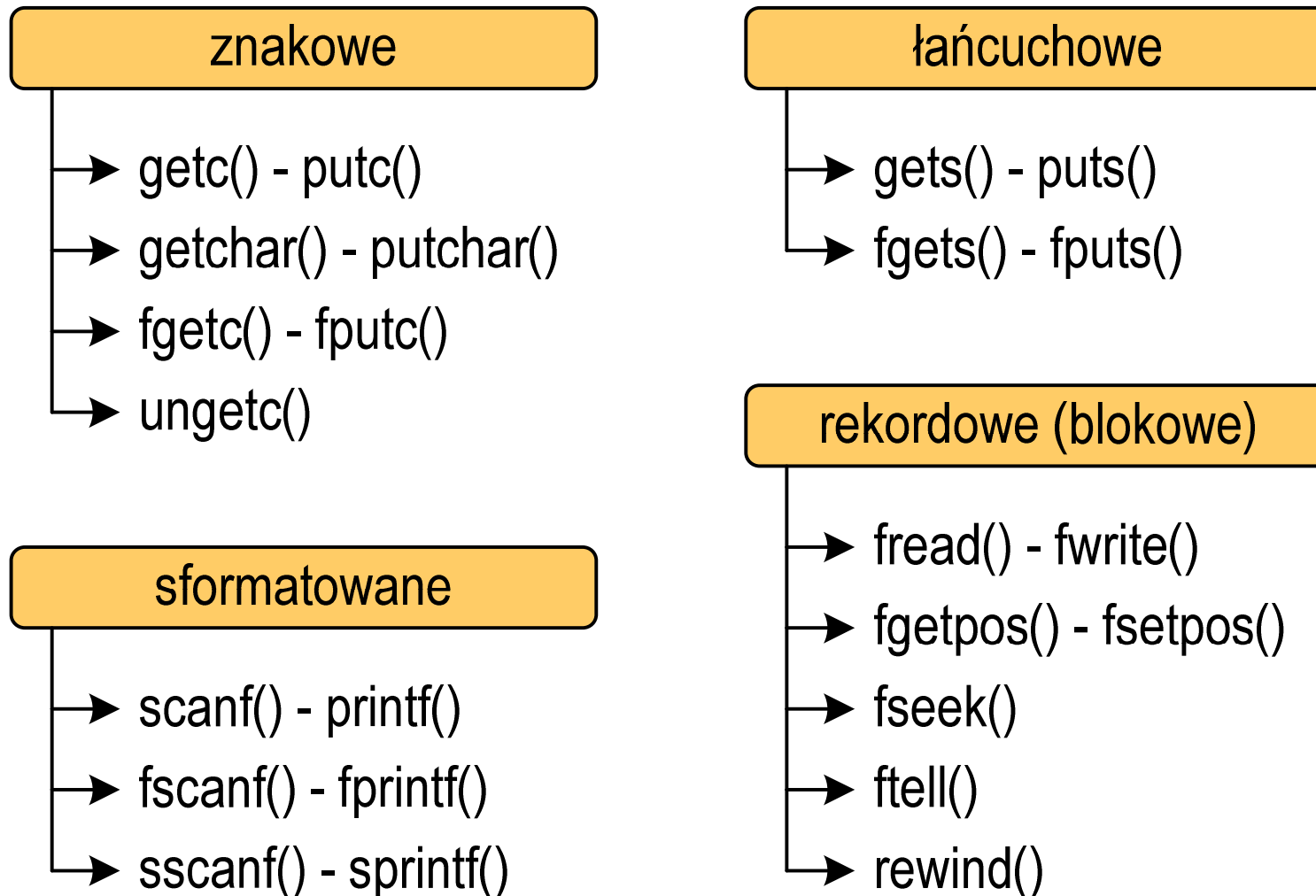
```
char znak = 'x';
```

- W rzeczywistości stała znakowa jest to liczba całkowita, której wartość odpowiada wartości kodu ASCII reprezentowanego znaku
- Zamiast powyższego kodu można napisać:

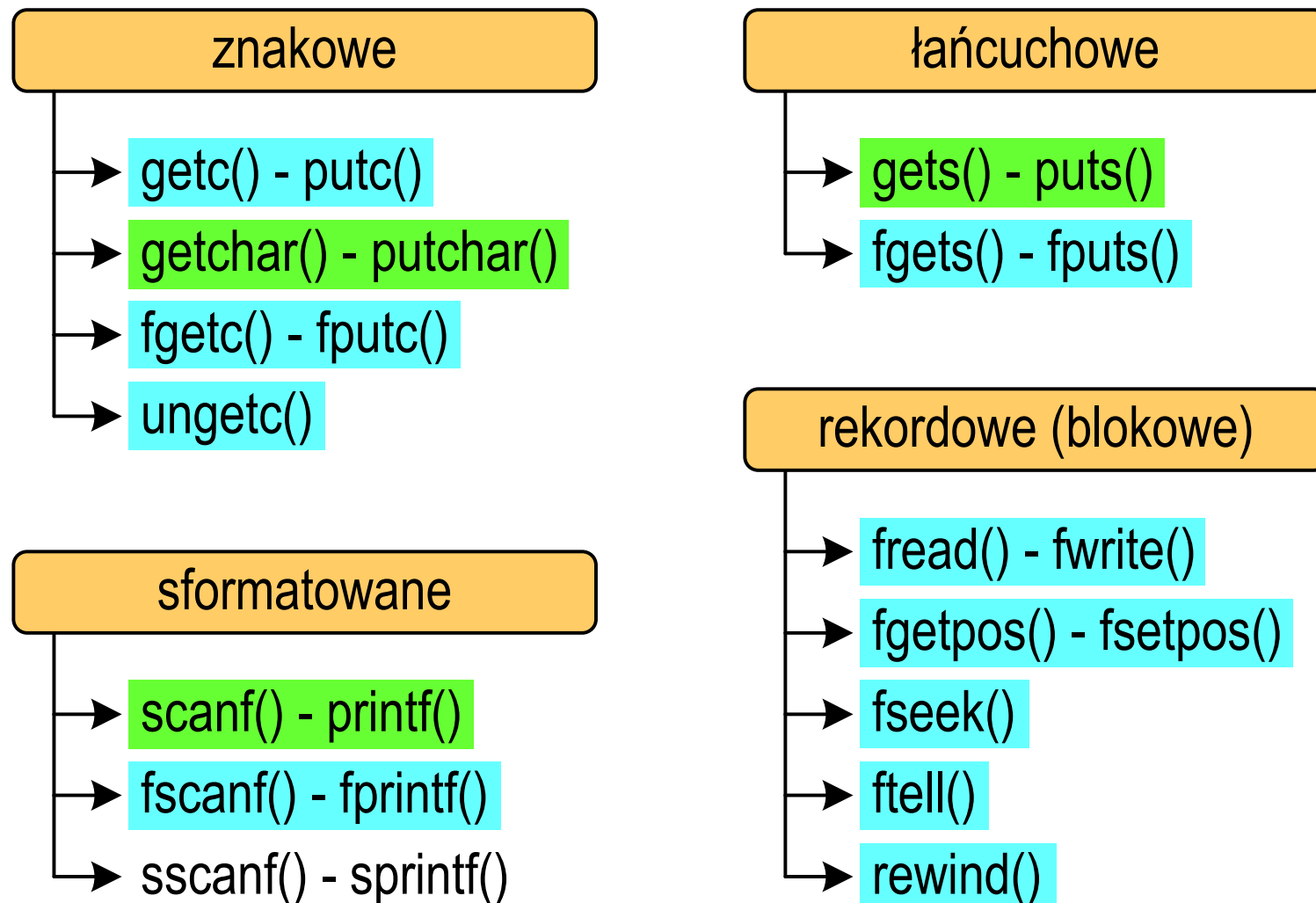
```
char znak = 120;
```

- Uwaga:
  - **'x'** - stała znakowa (jeden znak)
  - **"x"** - łańcuch znaków (dwa znaki: **'x'** oraz **'\0'**)

# Język C - standardowe funkcje wejścia-wyjścia



# Język C - standardowe funkcje wejścia-wyjścia



## Język C - wyświetlenie tekstu

- Wyświetlenie tekstu funkcją `printf()` wymaga specyfikatora `%s`

```
char napis[15] = "Jan Kowalski";  
printf("Osoba: [%s]\n", napis);
```

```
Osoba: [Jan Kowalski]
```

- W specyfikatorze `%s`: szerokość określa szerokość pola, zaś precyzja - liczbę pierwszych znaków z łańcucha

```
char napis[15] = "Jan Kowalski";  
printf("[%10.6s]\n", napis);
```

```
[    Jan Ko]
```

## Język C - wyświetlenie tekstu

- Do wyświetlenia tekstu można zastosować funkcję `puts()`

`puts()`

`int puts(const char *s);`

- Funkcja `puts()` wypisuje na `stdout` (ekran) zawartość łańcucha znakowego (ciąg znaków zakończony znakiem `'\0'`), zastępując znak `'\0'` znakiem `'\n'`

```
char napis[15] = "Jan Kowalski";  
puts(napis);
```

Jan Kowalski

## Język C - wyświetlenie tekstu

- Wyświetlenie znaku funkcją `printf()` wymaga specyfikatora `%c`

```
char znak = 'x';  
printf("Znak to: [%c]\n", znak);
```

```
Znak to: [x]
```

- Do wyświetlenia znaku można zastosować także funkcję `putchar()`

<code>putchar()</code>	<code>int putchar(int znak);</code>
------------------------	-------------------------------------

```
putchar('K'); putchar(111); putchar(0x74);
```

```
Kot
```

## Język C - wyświetlenie tekstu

- Łańcuch znaków jest zwykłą tablicą - można więc odwoływać się do jej pojedynczych elementów

```
char txt[15] = "Ola ma laptopa";  
  
printf("Znaki: ");  
for (int i=0; i<15; i++) printf("%c ",txt[i]);
```

```
Znaki: O l a   m a   l a p t o p a
```

```
printf("Kody: ");  
for (int i=0; i<15; i++) printf("%d ",txt[i]);
```

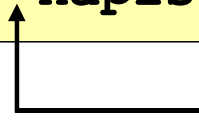
```
Kody:  79 108 97 32 109 97 32 108 97 112 116 111 112 97 0
```

## Język C - wczytanie tekstu

- Do wczytania tekstu funkcją `scanf()` stosowany jest specyfikator `%s`

```
char napis[15];  
scanf("%s", napis);
```

brak znaku `&`



- W specyfikatorze formatu `%s` można podać szerokość

```
char napis[15];  
scanf("%10s", napis);
```

- W powyższym przykładzie `scanf()` zakończy wczytywanie tekstu po pierwszym białym znaku (spacja, tabulacja, enter) lub w momencie pobrania 10 znaków

## Język C - wczytanie tekstu

- W przypadku wprowadzenia tekstu "To jest napis", funkcja `scanf()` zapamięta tylko wyraz "To"
- Zapamiętanie całego wiersza tekstu (do naciśnięcia klawisza `Enter`) wymaga użycia funkcji `gets()`

`gets()`

`char *gets(char *s);`

- Funkcja `gets()` wprowadza wiersz (ciąg znaków zakończony `'\n'`) ze strumienia `stdin` (klawiatura) i umieszcza w obszarze pamięci wskazywanym przez wskaźnik `s` zastępując `'\n'` znakiem `'\0'`

```
char napis[15];  
gets(napis);
```

## Język C - wczytanie znaku

- Wczytanie jednego znaku funkcją `scanf()` wymaga specyfikatora formatu `%c` (przed zmienną `znak` musi wystąpić operator `&`)

```
int znak;  
scanf ("%c", &znak);
```

- Do wczytania znaku można zastosować także funkcję `getchar()`

<code>getchar()</code>	<code>int getchar(void);</code>
------------------------	---------------------------------

```
int znak;  
znak = getchar();
```

## Język C - plik nagłówkowy string.h

<code>strcpy()</code>	<code>char *strcpy(char *s1, const char *s2);</code>
-----------------------	--

- Kopiuje łańcuch `s2` do łańcucha `s1`

```
char txt[10];  
txt = "Pies";           /* BŁĄD!!! */  
strcpy(txt, "Pies");
```

<code>strlen()</code>	<code>size_t strlen(const char *s);</code>
-----------------------	--

- Zwraca długość łańcucha znaków, nie uwzględnia znaku `'\0'`

<code>strcmp()</code>	<code>int strcmp(const char *s1, const char *s2);</code>
-----------------------	--

- Porównuje łańcuchy `s1` i `s2` z rozróżnianiem wielkości liter