Informatyka 1

Politechnika Białostocka - Wydział Elektryczny

Elektrotechnika, semestr II, studia niestacjonarne I stopnia

Rok akademicki 2018/2019

Wykład nr 3 (22.03.2019)

dr inż. Jarosław Forenc

Plan wykładu nr 3

- Jednostki informacji cyfrowej
  - słowo, FLOPS
- Kodowanie znaków
  - ASCII, ISO/IEC 646, ISO 8859
  - EBCDIC, Windows-1250, Unicode
- Język C
  - instrukcje warunkowa if
  - operator warunkowy
  - instrukcja switch

Słowo maszynowe (słowo)

- Słowo maszynowe (słowo - ang. word) - jednostka danych używana przez określony komputer (określoną architekturę)
- Słowo składa się odgórnie określonej liczby bitów, nazywanej długością lub szerokością słowa (najczęściej jest to potęga 2, np. 8, 16, 32, 64 bity)
- Zazwyczaj wielkość słowa określa:
  - rozmiar rejestrów procesora
  - rozmiar szyny danych i szyny adresowej
- Architektury:
  - 8-bitowa: Intel 8080, Z80, Motorola 6800, Intel 8051
  - 16-bitowa: Intel 8086, Intel 80286
  - 32-bitowa: Intel od 80386 do i7, AMD od 5x86 do Athlona, ARM
  - 64-bitowa: Intel Itanium, Pentium 4/EM64T, Core 2, Core i7
    AMD Opteron, Athlon 64, Athlon II

FLOPS

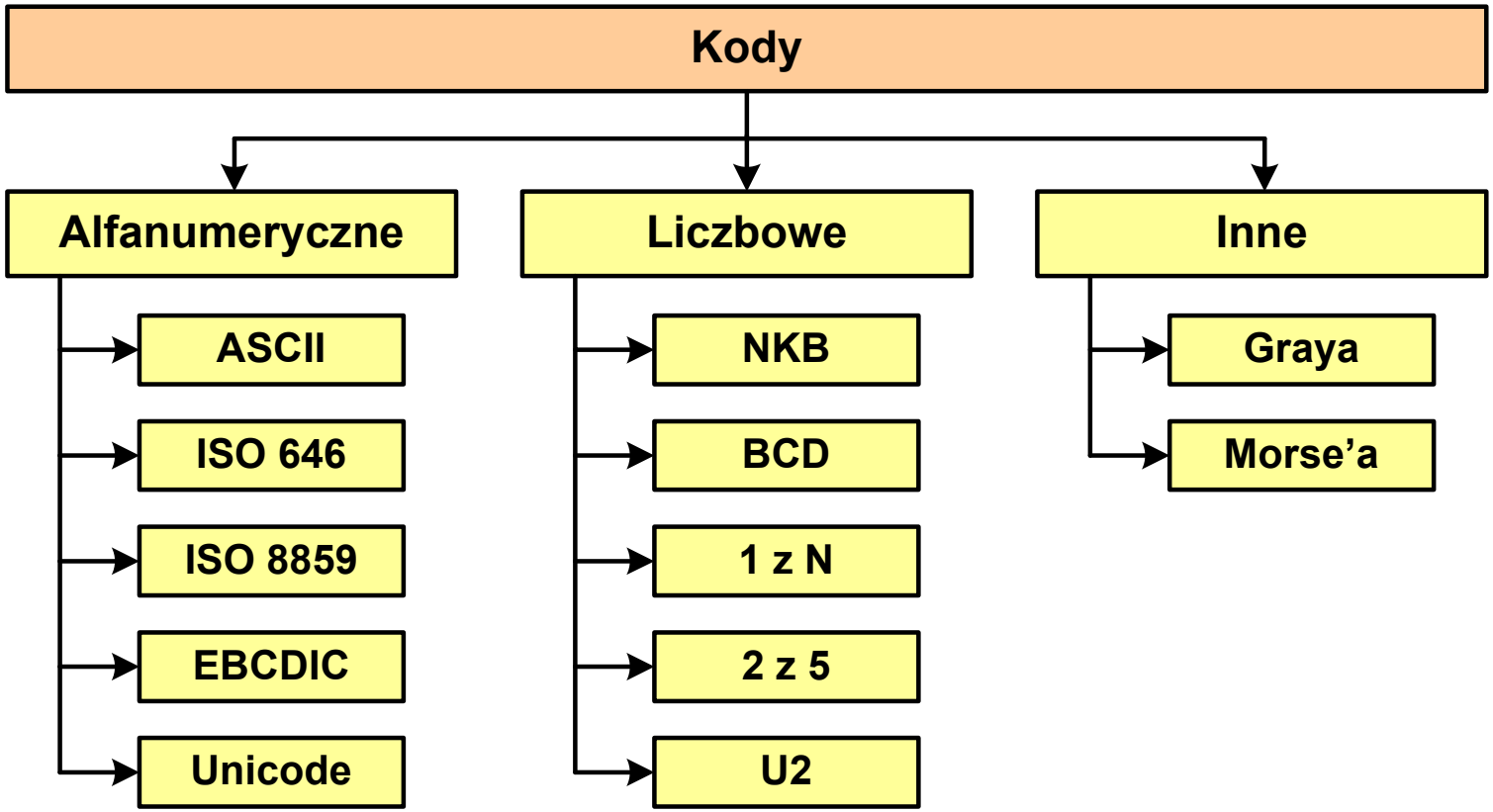
- FLOPS (FLoating point Operations Per Second)
  - liczba operacji zmiennoprzecinkowych na sekundę
  - jednostka wydajności układów zmiennoprzecinkowych
- Przykłady wydajności procesorów (teoretyczne):
  - Intel Core i7 975 3,46 GHz - 55,36 GFlops
  - Intel Core2 Quad Q9650 3,00 GHz - 48 GFlops
  - Intel Core2 Duo E8400 3,00 GHz - 24 GFlops
  - najszybszy system równoległy na świecie:

    Summit (USA) - 143.500.000 GFlops
    DOE/SC/Oak Ridge National Laboratory

    processors: IBM POWER9 (2/node)
    GPUs: 27,648 Nvidia Volta V100s (6/node)
    nodes: 4.608, cores: 2.397.824
    www.top500.org

Kodowanie

- Kodowanie - proces przekształcania jednego rodzaju postaci informacji na inną postać



Kod ASCII

- ASCII - American Standard Code for Information Interchange
  - 7-bitowy kod przypisujący liczby z zakresu 0-127:
    - literom (alfabet angielski)
    - cyfrom
    - znakom przestankowym
    - innym symbolom
    - poleceniom sterującym.

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	NUL	32	20	Space	64	40	@	96	60	`
1	1	SOH	33	21	!	65	41	A	97	61	a
2	2	STX	34	22	"	66	42	B	98	62	b
3	3	ETX	35	23	#	67	43	C	99	63	c
4	4	EOT	36	24	$	68	44	D	100	64	d
5	5	ENQ	37	25	%	69	45	E	101	65	e
6	6	ACK	38	26	&	70	46	F	102	66	f
7	7	BEL	39	27	'	71	47	G	103	67	g
8	8	BS	40	28	(	72	48	H	104	68	h
9	9	TAB	41	29	)	73	49	I	105	69	i
10	A	LF	42	2A	*	74	4A	J	106	6A	j
11	B	VT	43	2B	+	75	4B	K	107	6B	k
12	C	FF	44	2C	,	76	4C	L	108	6C	l
13	D	CR	45	2D	-	77	4D	M	109	6D	m
14	E	SO	46	2E	.	78	4E	N	110	6E	n
15	F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	DEL

Kod ASCII - Kody sterujące

- Kody sterujące - 33 kody, o numerach: 0-31, 127

Dec	Hex	Char		Dec	Hex	Char	
0	0	NUL	(null)	16	10	DLE	(data link escape)
1	1	SOH	(start of heading)	17	11	DC1	(device control 1)
2	2	STX	(start of text)	18	12	DC2	(device control 2)
3	3	ETX	(end of text)	19	13	DC3	(device control 3)
4	4	EOT	(end of tramsmission)	20	14	DC4	(device control 4)
5	5	ENQ	(enquiry)	21	15	NAK	(negative acknowlege)
6	6	ACK	(acknowledge)	22	16	SYN	(synchronous idle)
7	7	BEL	(bell)	23	17	ETB	(end of trans. block)
8	8	BS	(backspace)	24	18	CAN	(cancel)
9	9	TAB	(horizontal tab)	25	19	EM	(end of medium)
10	A	LF	(NL line feed, new line)	26	1A	SUB	(subtitute)
11	B	VT	(vertical tab)	27	1B	ESC	(escape)
12	C	FF	(NP form feed, new page)	28	1C	FS	(file separator)
13	D	CR	(carriage return)	29	1D	GS	(group separator)
14	E	SO	(shift out)	30	1E	RS	(record separator)
15	F	SI	(shift in)	31	1F	US	(unit separator)
				127	7F	DEL	

- W języku C:

0 (NULL) - \0 7 (BEL) - \a 8 (BS) - \b

9 (TAB) - \t 10 (LF) - \n 13 (CR) - \r

Kod ASCII - Pliki tekstowe

- Elementami pliku tekstowego są wiersze, mogą one mieć różną długość
- W systemie Windows każdy wiersz pliku zakończony jest parą znaków:
  - CR, ang. carriage return - powrót karetki, kod ASCII - $13_{(10)} = 0D_{(16)}$
  - LF, ang. line feed - przesunięcie o wiersz, kod ASCII - $10_{(10)} = 0A_{(16)}$
- Załóżmy, że plik tekstowy ma postać:

```
Pierwszy wiersz pliku
Drugi wiersz pliku
Trzeci wiersz pliku
```

- Rzeczywista zawartość pliku jest następująca:

```
00000000: 50 69 65 72 77 73 7A 79|20 77 69 65 72 73 7A 20 | Pierwszy wiersz
00000010: 70 6C 69 6B 75 0D 0A 44|72 75 67 69 20 77 69 65 | pliku■■Drugi wie
00000020: 72 73 7A 20 70 6C 69 6B|75 0D 0A 54 72 7A 65 63 | rsz pliku■■Trzec
00000030: 69 20 77 69 65 72 73 7A|20 70 6C 69 6B 75 0D 0A | i wiersz pliku■■
```

- Wydruk zawiera:
  - przesunięcie od początku pliku (szesnastkowo)
  - wartości poszczególnych bajtów pliku (szesnastkowo)
  - znaki odpowiadające bajtom pliku (traktując bajty jako kody ASCII)

Kod ASCII - Pliki tekstowe

- W systemie Linux znakiem końca wiersza jest tylko LF o kodzie ASCII - $10_{(10)} = 0A_{(16)}$
- Załóżmy, że plik tekstowy ma postać:

```
Pierwszy wiersz pliku
Drugi wiersz pliku
Trzeci wiersz pliku
```

- Rzeczywista zawartość pliku jest następująca:

```
00000000: 50 69 65 72 77 73 7A 79|20 77 69 65 72 73 7A 20 | Pierwszy wiersz
00000010: 70 6C 69 6B 75 0A 44 72|75 67 69 20 77 69 65 72 | pliku■Drugi wier
00000020: 73 7A 20 70 6C 69 6B 75|0A 54 72 7A 65 63 69 20 | sz pliku■Trzeci
00000030: 77 69 65 72 73 7A 20 70|6C 69 6B 75 0A          | wiersz pliku■
```

- Podczas przesyłania pliku tekstowego (np. przez protokół ftp) z systemu Linux do systemu Windows pojedynczy znak LF zamieniany jest automatycznie na parę znaków CR i LF
- Błędne przesłanie pliku tekstowego (w trybie binarnym) powoduje nieprawidłowe jego wyświetlanie:

```
Pierwszy wiersz pliku■Drugi wiersz pliku■Trzeci wiersz pliku■
```

ISO/IEC 646

- ISO/IEC 646 - norma definiująca modyfikację 7-bitowego kodowania ASCII, stosowana w latach 70-tych i 80-tych
- W normie określono 10 pozycji na znaki w języku kraju, który przyjął tę normę oraz 2 pozycje na znaki walut

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	

- żółty - znaki narodowe
- niebieski - znaki walut

- Wszystkie pozostałe znaki są zgodne z ASCII

ISO/IEC 646 - odmiany narodowe

USA

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	

Niemcy

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	§	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	Ä	Ö	Ü	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	ä	ö	ü	ß	

Francja

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	£	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	à	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	°	ç	§	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	é	ù	é	¨	

Polska

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	#	zł	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	ę	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	ź	\	ń	ś	_
60	ą	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	ó	ł	ż	ć	

Norma: BN-74/3101-01

ISO/IEC 8859

- ISO/IEC 8859 - zestaw standardów służących do kodowania znaków za pomocą 8-bitów
- Wszystkie zestawy ISO 8859 mają znaki $0_{(10)}$-$127_{(10)}$ ($00_{(16)}$-$7F_{(16)}$) takie same jak w kodzie ASCII
- Pozycjom $128_{(10)}$-$159_{(10)}$ ($80_{(16)}$-$9F_{(16)}$) przypisane są dodatkowe kody sterujące, tzw. C1 (obecnie nie są używane)
- Od czerwcu 2004 roku ISO 8859 nie jest rozwijane.

ISO/IEC 8859

- Stosowane standardy ISO 8859:
  - ISO 8859-1 (Latin-1) - alfabet łaciński dla Europy zachodniej
  - ISO 8859-2 (Latin-2) - łaciński dla Europy środkowej i wschodniej
  - ISO 8859-3 (Latin-3) - łaciński dla Europy południowej
  - ISO 8859-4 (Latin-4) - łaciński dla Europy północnej
  - ISO 8859-5 (Cyrillic) - dla cyrylicy
  - ISO 8859-6 (Arabic) - dla alfabetu arabskiego
  - ISO 8859-7 (Greek) - dla alfabetu greckiego
  - ISO 8859-8 (Hebrew) - dla alfabetu hebrajskiego
  - ISO 8859-9 (Latin-5)
  - ISO 8859-10 (Latin-6)
  - ISO 8859-11 (Thai) - dla alfabetu tajskiego
  - ISO 8859-12 - brak
  - ISO 8859-13 (Latin-7)
  - ISO 8859-14 (Latin-8) - zawiera polskie litery
  - ISO 8859-15 (Latin-9)
  - ISO 8859-16 (Latin-10) - łaciński dla Europy środkowej, zawiera polskie litery

ISO/IEC 8859-1

- ISO/IEC 8859-1, Latin-1 („zachodnioeuropejskie”)
- kodowanie używane w Amerykach, Europie Zachodniej, Oceanii i większej części Afryki
- dostępne języki: albański, angielski, baskijski, duński, estoński, fiński, francuski, hiszpański, irlandzki, islandzki, kataloński, łaciński, niderlandzki, niemiecki, norweski, portugalski, retoromański, szkocki, szwedzki, włoski
- 191 znaków łacińskiego pisma.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Nieużywane															
90																
A0	NB SP	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	SHY	®	¯
B0	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C0	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D0	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E0	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F0	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

SP - spacja
NBSP - twarda spacja
SHY - miękki dywiz (myślnik)

ISO/IEC 8859-2

- ISO/IEC 8859-2, Latin-2 („środkowo”, „wschodnioeuropejskie”)
- dostępne języki: bośniacki, chorwacki, czeski, węgierski, polski, rumuński, serbski, serbsko-chorwacki, słowacki, słoweński, górno- i dolnołużycki
- możliwość przedstawienia znaków w języku niemieckim i angielskim
- 191 znaków łacińskiego pisma
- kodowanie zgodne z Polską Normą.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Nieużywane															
90																
A0	NB SP	Ą	˘	Ł	¤	Ľ	Ś	§	¨	Š	Ş	Ť	Ź	SHY	Ž	Ż
B0	°	ą	˛	ł	´	ľ	ś	ˇ	¸	š	ş	ť	ź	˝	ž	ż
C0	Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç	Č	É	Ę	Ë	Ě	Í	Î	Ď
D0	Đ	Ń	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ű	Ü	Ý	Ţ	ß
E0	ŕ	á	â	ă	ä	ĺ	ć	ç	č	é	ę	ë	ě	í	î	ď
F0	đ	ń	ň	ó	ô	ő	ö	÷	ř	ů	ú	ű	ü	ý	ţ	˙

SP - spacja
NBSP - twarda spacja
SHY - miękki dywiz (myślnik)

ISO/IEC 8859-2 - Litery diakrytyczne w j. polskim

- 18 liter:
  - Ą - ą
  - Ć - ć
  - Ę - ę
  - Ł - ł
  - Ń - ń
  - Ó - ó
  - Ś - ś
  - Ź - ź
  - Ż - ż

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Nieużywane															
90																
A0	NB SP	Ą	˘	Ł	¤	Ľ	Ś	§	¨	Š	Ş	Ť	Ź	SHY	Ž	Ż
B0	°	ą	˛	ł	´	ľ	ś	ˇ	¸	š	ş	ť	ź	˝	ž	ż
C0	Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç	Č	É	Ę	Ë	Ě	Í	Î	Ď
D0	Đ	Ń	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ű	Ü	Ý	Ţ	ß
E0	ŕ	á	â	ă	ä	ĺ	ć	ç	č	é	ę	ë	ě	í	î	ď
F0	đ	ń	ň	ó	ô	ő	ö	÷	ř	ů	ú	ű	ü	ý	ţ	˙

ISO/IEC 8859-1 i ISO/IEC 8859-2 - porównanie

ISO 8859-1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A0	NB SP	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	SHY	®	¯
B0	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C0	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D0	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E0	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F0	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

ISO 8859-2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A0	NB SP	Ą	˘	Ł	¤	Ľ	Ś	§	¨	Š	Ş	Ť	Ź	SHY	Ž	Ż
B0	°	ą	˛	ł	´	ľ	ś	ˇ	¸	š	ş	ť	ź	˝	ž	ż
C0	Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç	Č	É	Ę	Ë	Ě	Í	Î	Ď
D0	Đ	Ń	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ű	Ü	Ý	Ţ	ß
E0	ŕ	á	â	ă	ä	ĺ	ć	ç	č	é	ę	ë	ě	í	î	ď
F0	đ	ń	ň	ó	ô	ő	ö	÷	ř	ů	ú	ű	ü	ý	ţ	˙

EBCDIC

- EBCDIC - Extended Binary Coded Decimal Interchange Code
- 8-bitowe kodowanie znaków stworzone jako rozszerzenie kodowania BCD
  - używane głównie w systemach IBM w latach 60-tych XX wieku
  - umożliwia zapisanie do 256 różnych symboli
  - brak zachowania kolejności liter zgodnie z kolejnością kodów, np. po R nie ma S
  - kody EBCDIC nie są zgodne z ASCII.

EBCDIC

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	*Znaki kontrolne*															
10																
20																
30																
40	SP	NB SP	â	ä	à	á	ã	å	ç	ñ	[	.	<	(	+	!
50	&	é	ê	ë	è	í	î	ï	ì	ß	]	$	*	)	;	^
60	-	/	Â	Ä	À	Á	Ã	Å	Ç	Ñ	¦	,	%	_	>	?
70	ø	É	Ê	Ë	È	Í	Î	Ï	Ì	`	:	#	@	'	=	"
80	Ø	a	b	c	d	e	f	g	h	i	«	»	ð	ý	Þ	±
90	°	j	k	l	m	n	o	p	q	r	ª	º	æ	¸	Æ	¤
A0	µ	~	s	t	u	v	w	x	y	z	¡	¿	Ð	Ý	þ	®
B0	¢	£	¥	·	©	§	¶	¼	½	¾	¬	|	¯	¨	´	×
C0	{	A	B	C	D	E	F	G	H	I	SHY	ô	Ö	Ò	Ó	õ
D0	}	J	K	L	M	N	O	P	Q	R	¹	û	ü	ù	ú	ÿ
E0	\	÷	S	T	U	V	W	X	Y	Z	²	Ô	Ö	Ò	Ó	Õ
F0	0	1	2	3	4	5	6	7	8	9	³	Û	Ü	Ù	Ú	

EBCDIC i ISO 8859-1 - porównanie

ISO 8859-1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	*Znaki kontrolne*															
10																
20	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	*Nieużywane*															
90																
A0	NB SP	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	SHY	®	¯
B0	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C0	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D0	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E0	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F0	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

EBCDIC

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	*Znaki kontrolne*															
10																
20																
30																
40	SP	NB SP	â	ä	à	á	ã	å	ç	ñ	[	.	<	(	+	!
50	&	é	ê	ë	è	í	î	ï	ì	ß	]	$	*	)	;	^
60	-	/	Â	Ä	À	Á	Ã	Å	Ç	Ñ	¦	,	%	_	>	?
70	ø	É	Ê	Ë	È	Í	Î	Ï	Ì	`	:	#	@	'	=	"
80	Ø	a	b	c	d	e	f	g	h	i	«	»	ð	ý	Þ	±
90	°	j	k	l	m	n	o	p	q	r	ª	º	æ	¸	Æ	¤
A0	µ	~	s	t	u	v	w	x	y	z	¡	¿	Ð	Ý	þ	®
B0	¢	£	¥	·	©	§	¶	¼	½	¾	¬	|	¯	¨	´	×
C0	{	A	B	C	D	E	F	G	H	I	SHY	ô	Ö	Ò	Ó	õ
D0	}	J	K	L	M	N	O	P	Q	R	¹	û	ü	ù	ú	ÿ
E0	\	÷	S	T	U	V	W	X	Y	Z	²	Ô	Ö	Ò	Ó	Õ
F0	0	1	2	3	4	5	6	7	8	9	³	Û	Ü	Ù	Ú	

Windows-1250

- Windows-1250 (CP-1250) - strona kodowa używana przez system Microsoft Windows do reprezentacji tekstów w językach środkowoeuropejskich używających alfabetu łacińskiego
- Obsługiwane języki: albański, chorwacki, czeski, polski, rumuński, słowacki, słoweński, węgierski (ale także niemiecki)
- Windows-1250 jest podobny do ISO 8859-2 - posiada wszystkie jego drukowalne znaki (a także kilka dodatkowych), lecz kilka z nich zajmuje inne miejsca.

ISO 8859-2 i Windows-1250 - porównanie

ISO 8859-2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	Nieużywane															
90																
A0	NB SP	Ą	˘	Ł	¤	Ľ	Ś	§	¨	Š	Ş	Ť	Ź	SHY	Ž	Ż
B0	°	ą	˛	ł	´	ľ	ś	ˇ	¸	š	ş	ť	ź	˝	ž	ż
C0	Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç	Č	É	Ę	Ë	Ě	Í	Î	Ď
D0	Đ	Ń	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ű	Ü	Ý	Ţ	ß
E0	ŕ	á	â	ă	ä	ĺ	ć	ç	č	é	ę	ë	ě	í	î	ď
F0	đ	ń	ň	ó	ô	ő	ö	÷	ř	ů	ú	ű	ü	ý	ţ	˙

Windows-1250

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Znaki kontrolne															
10																
20	SP	!	"	#	$	%	&	‘	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80	€		‚		„	…	†	‡		‰	Š	‹	Ś	Ť	Ž	Ź
90		‘	’	“	”	•	–	—		™	š	›	ś	ť	ž	ź
A0	NB SP	ˇ	˘	Ł	¤	Ą	¦	§	¨	©	Ş	«	¬	SH Y	Ž	Ż
B0	°	±	˛	ł	´	µ	¶	·	¸	ą	ş	»	Ľ	˝	ľ	ż
C0	Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç	Č	É	Ę	Ë	Ě	Í	Î	Ď
D0	Đ	Ń	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ű	Ü	Ý	Ţ	ß
E0	ŕ	á	â	ă	ä	ĺ	ć	ç	č	é	ę	ë	ě	í	î	ď
F0	đ	ń	ň	ó	ô	ő	ö	÷	ř	ů	ú	ű	ü	ý	ţ	˙

Problem kodowania polskich liter diakrytycznych

- Problem z wyświetlaniem polskich liter diakrytycznych
  - Tekst zapisany w standardzie ISO-8859-2:

```
Ą Ć Ę Ł Ń Ó Ś Ź Ż
ą ć ę ł ń ó ś ź ż
```

  - Tekst wyświetlony w Notatniku systemu Windows (Windows-1250):

```
ˇ Ć Ę Ł Ń Ó ¦ ¬ Ż
± ć ę ł ń ó ¶ Ľ ż
```

Unicode (Unikod)

- Komputerowy zestaw znaków mający obejmować wszystkie pisma i inne znaki (symbole techniczne, wymowy) używane na świecie
- Unicode przypisuje unikalny numer każdemu znakowi, niezależny od używanej platformy, programu czy języka
- Rozwijany przez konsorcjum utworzone przez firmy komputerowe, producentów oprogramowania oraz grupy użytkowników
  - http://www.unicode.org
- Pierwsza wersja: Unicode 1.0 (10.1991)
- Ostatnia wersja: Unicode 12.0 (05.03.2019)
  - The Unicode Consortium. The Unicode Standard, Version 12.0.0, (Mountain View, CA: The Unicode Consortium, 2019)
  - http://www.unicode.org/versions/Unicode12.0.0/
  - koduje 137.928 znaków

Unicode - Zakresy

Zakres:	Znaczenie:
U+0000 - U+007F	Basic Latin (to samo co w ASCII)
U+0080 - U+00FF	Latin-1 Supplement (to samo co w ISO/IEC 8859-1)
U+0100 - U+017F	Latin Extended-A
U+0180 - U+024F	Latin Extended-B
U+0250 - U+02AF	IPA Extensions
U+02B0 - U+02FF	Spacing Modifiers Letters
...	
U+0370 - U+03FF	Greek
U+0400 - U+04FF	Cyrillic
...	
U+1D00 - U+1D7F	Phonetic Extensions
U+1D80 - U+1DBF	Phonetic Extensions Supplement
U+1E00 - U+1EFF	Latin Extended Additional
U+1F00 - U+1FFF	Greek Extended
...	

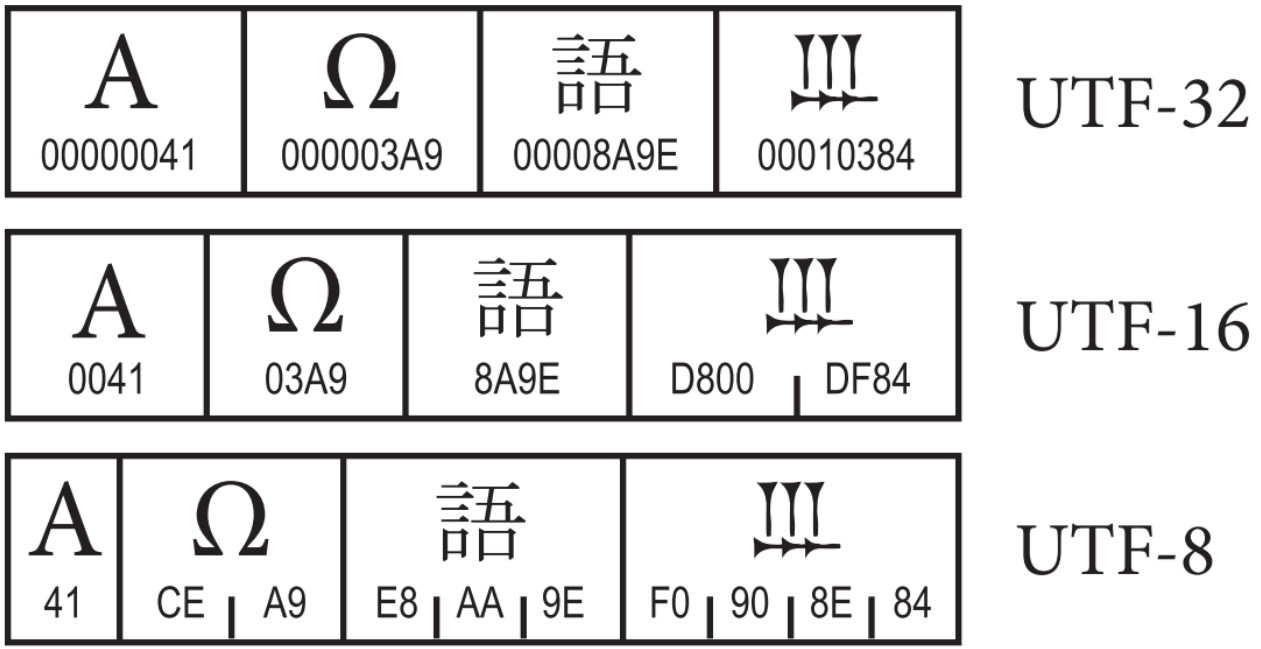
Unicode

- Standard Unicode definiuje nie tylko kody numeryczne przypisane poszczególnym znakom, ale także określa sposób bajtowego kodowania znaków
- Kodowanie określa sposób w jaki znaki ze zbioru mają być zapisane w postaci binarnej
- Istnieją trzy podstawowe metody kodowania:
  - 32-bitowe: UTF-32
  - 16-bitowe: UTF-16
  - 8-bitowe: UTF-8

  gdzie: UTF - UCS Transformation Format
  UCS - Universal Character Set
- Wszystkie metody obejmują wszystkie kodowane znaki w Unicode.

Unicode

- Metody kodowania różnią się liczbą bajtów przeznaczonych do opisania kodu znaku



źródło: The Unicode Consortium. The Unicode Standard, Version 8.0

Unicode - kodowanie UTF-32

- UTF-32 - sposób kodowania standardu Unicode wymagający użycia 32-bitowych słów

- Kod znaku ma zawsze stałą długość 4 bajtów i przedstawia numer znaku w tabeli Unikodu
- Kody obejmują zakres od 0 do 0x10FFFF (od 0 do 1 114 111)
- Kodowanie to jest jednak bardzo nieefektywne - zakodowane ciągi znaków są 2-4 razy dłuższe niż ciągi tych samych znaków zapisanych w innych kodowaniach.

Unicode - kodowanie UTF-16

- UTF-16 - sposób kodowania standardu Unicode wymagający użycia 16-bitowych słów

- Dla znaków z przedziału od U+0000 do U+FFFF używane jest jedno słowo, którego wartość jest jednocześnie kodem znaku w Unicode
- Dla znaków z wyższych pozycji używa się dwóch słów:
  - pierwsze słowo należy do przedziału: U+D800 - U+DBFF
  - drugie słowo należy do przedziału: U+DC00 - U+DFFF.

Unicode - kodowanie UTF-8

- UTF-8 - kodowanie ze zmienną długością reprezentacji znaku wymagające użycia 8-bitowych słów

- Znaki Unikodu są mapowane na ciągi bajtów
  - 0x00 do 0x7F - bity 0xxxxxxx
  - 0x80 do 0x7FF - bity 110xxxxx 10xxxxxx
  - 0x800 do 0xFFFF - bity 1110xxxx 10xxxxxx 10xxxxxx
  - 0x10000 do 0x1FFFFF - bity 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
  - 0x200000 do 0x3FFFFFF - bity 111110xx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx
  - 0x4000000 do 0x7FFFFFFF - bity 1111110x 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx

Unicode

	010	011	012	013	014	015	016	017
0	Ā 0100	Đ 0110	Ġ 0120	İ 0130	ŀ 0140	Ő 0150	Š 0160	Ű 0170
1	ā 0101	đ 0111	ġ 0121	ı 0131	Ł 0141	ő 0151	š 0161	ű 0171
2	Ă 0102	Ē 0112	Ģ 0122	IJ 0132	ł 0142	Œ 0152	Ţ 0162	Ų 0172
3	ă 0103	ē 0113	ģ 0123	ij 0133	Ń 0143	œ 0153	ţ 0163	ų 0173
4	Ą 0104	Ĕ 0114	Ĥ 0124	Ĵ 0134	ń 0144	Ŕ 0154	Ť 0164	Ŵ 0174
5	ą 0105	ĕ 0115	ĥ 0125	ĵ 0135	Ņ 0145	ŕ 0155	ť 0165	ŵ 0175
6	Ć 0106	Ė 0116	Ħ 0126	Ķ 0136	ņ 0146	Ŗ 0156	Ŧ 0166	Ŷ 0176
7	ć 0107	ė 0117	ħ 0127	ķ 0137	Ň 0147	ŗ 0157	ŧ 0167	ŷ 0177

European Latin

0100 Ā LATIN CAPITAL LETTER A WITH MACRON
≡ 0041 A 0304 ◌̄

0101 ā LATIN SMALL LETTER A WITH MACRON
• Latvian, Latin, ...
≡ 0061 a 0304 ◌̄

0102 Ă LATIN CAPITAL LETTER A WITH BREVE
≡ 0041 A 0306 ◌̆

0103 ă LATIN SMALL LETTER A WITH BREVE
• Romanian, Vietnamese, Latin, ...
≡ 0061 a 0306 ◌̆

0104 Ą LATIN CAPITAL LETTER A WITH OGONEK
≡ 0041 A 0328 ◌̨

0105 ą LATIN SMALL LETTER A WITH OGONEK
• Polish, Lithuanian, ...
≡ 0061 a 0328 ◌̨

0106 Ć LATIN CAPITAL LETTER C WITH ACUTE
≡ 0043 C 0301 ◌́

0107 ć LATIN SMALL LETTER C WITH ACUTE
• Polish, Croatian, ...
→ 045B ћ cyrillic small letter tshe
≡ 0063 c 0301 ◌́

Unicode

27308 **CJK Unified Ideographs Extension B** **27342**

27308 虫 142.8	蟂 UCS2003	蟂 GKX-1086.03	蟂 T4-4721
27309 虫 142.8	蟄 UCS2003	蟄 GKX-1086.05	蟄 T5-4955
2730A 虫 142.8	蟾 UCS2003	蟾 GKX-1086.08	蟾 T4-467D
2730B 虫 142.8	蟿 UCS2003	蟿 GKX-1086.10	蟿 T6-6223
2730C 虫 142.8	蠁 UCS2003	蠁 GKX-1086.12	蠁 T5-495F
2730D 虫 142.8	蠑 UCS2003	蠑 GKX-1086.22	蠑 T4-4677
2731B 虫 142.8	蠹 UCS2003	蠹 GKX-1088.15	蠹 T6-617B
2731C 虫 142.8	蠧 UCS2003	蠧 GKX-1088.16	蠧 T6-6221
2731D 虫 142.8	蠶 UCS2003	蠶 GKX-1088.17	蠶 T5-4960
2731E 虫 142.7	蠸 UCS2003	蠸 GKX-1088.18	
2731F 虫 142.8	蠻 UCS2003	蠻 GKX-1088.19	蠻 T6-6174
27320 虫 142.8	蠿 UCS2003	蠿 GKX-1088.20	蠿 T6-617D
2732F 虫 142.8	衂 UCS2003	衂 GHC	
27330 虫 142.9	衟 UCS2003	衟 GHC	
27331 虫 142.8	衠 UCS2003	衠 G4K	
27332 虫 142.8	衣 UCS2003	衣 GHC	
27333 虫 142.8	衲 UCS2003	衲 GHC	
27334 虫 142.8	衳 UCS2003	衳 T5-4953	

Język C - Pierwiastek kwadratowy

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

int main(void)
{
    float x, y;

    printf("Podaj liczbe: ");
    scanf("%f",&x);

    y = sqrt(x);

    printf("Pierwiastek liczby: %f\n",y);

    return 0;
}
```

```
Podaj liczbe: 15
Pierwiastek liczby: 3.872983
```

```
Podaj liczbe: -15
Pierwiastek liczby: -1.#IND00
```

Język C - Pierwiastek kwadratowy

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

int main(void)
{
    float x, y;

    printf("Podaj liczbe: ");
    scanf("%f",&x);

    if (x>=0)
    {
        y = sqrt(x);
        printf("Pierwiastek liczby: %f\n",y);
    }
    else
        printf("Blad! Liczba ujemna\n");

    return 0;
}
```

```
Podaj liczbe: 15
Pierwiastek liczby: 3.872983
```

```
Podaj liczbe: -15
Blad! Liczba ujemna
```

Język C - instrukcja warunkowa if

```
if (wyrażenie)
    instrukcja1
```

- jeśli wyrażenie jest prawdziwe, to wykonywana jest instrukcja1
- gdy wyrażenie jest fałszywe, to instrukcja1 nie jest wykonywana

```
if (wyrażenie)
    instrukcja1
else
    instrukcja2
```

- jeśli wyrażenie jest prawdziwe, to wykonywana jest instrukcja1, zaś instrukcja2 nie jest wykonywana
- gdy wyrażenie jest fałszywe, to wykonywana jest instrukcja2, zaś instrukcja1 nie jest wykonywana

- Wyrażenie w nawiasach:
  - prawdziwe - gdy jego wartość jest różna od zera
  - fałszywe - gdy jego wartość jest równa zero

Język C - instrukcja warunkowa if

```
if (wyrażenie)
    instrukcja
```

- Instrukcja:
  - prosta - jedna instrukcja zakończona średnikiem
  - złożona - jedna lub kilka instrukcji objętych nawiasami klamrowymi

```
if (x>0)
    printf("inst1");
```

```
if (x>0)
{
    printf("inst1");
    printf("inst2");
    ...
}
```

Język C - instrukcja warunkowa if

```
if (wyr)
    instr;
```

```
if (wyr)
{
    instr;
    instr;
}
```

```
if (wyr)
    instr;
else
    instr;
```

```
if (wyr)
{
    instr;
    instr;
}
else
{
    instr;
    instr;
}
```

```
if (wyr)
{
    instr;
    instr;
}
else
    instr;
```

```
if (wyr)
    instr;
else
{
    instr;
    instr;
}
```

```
if (wyr)
{
    instr;
}
else
{
    instr;
}
```

Język C - Operatory relacyjne (porównania)

Operator	Przykład	Znaczenie
`>`	`a > b`	a większe od b
`<`	`a < b`	a mniejsze od b
`>=`	`a >= b`	a większe lub równe b
`<=`	`a <= b`	a mniejsze lub równe b
`==`	`a == b`	a równe b
`!=`	`a != b`	a nierówne b (a różne od b)

- Wynik porównania jest wartością typu int i jest równy:
  - 1 - gdy warunek jest prawdziwy
  - 0 - gdy warunek jest fałszywy (nie jest prawdziwy)

Język C - Operatory logiczne

Operator	Znaczenie	Opis
`!`	`NOT, nie`	jednoargumentowy operator negacji logicznej - zmienia argument różny od zera na wartość 0, a argument równy zero na wartość 1
`&&`	`AND, i`	dwuargumentowy operator koniunkcji, iloczyn logiczny
`||`	`OR, lub`	dwuargumentowy operator alternatywy, suma logiczna

- Wynikiem zastosowania operatorów logicznych && i || jest wartość typu int równa 1 (prawda) lub 0 (fałsz)

```
if (x>5 && x<8)
```

```
if (x<=5 || x>8)
```

Język C - Wyrażenia logiczne

- Wyrażenia logiczne mogą zawierać:
  - operatory relacyjne
  - operatory logiczne
  - operatory arytmetyczne
  - operatory przypisania
  - zmienne
  - stałe
  - wywołania funkcji
  - ...
- Kolejność operacji wynika z priorytetu operatorów

Operator	Typ operatora
`!`	logiczny
`* / %`	arytmetyczne
`+ -`	arytmetyczne
`> < >= <=`	relacyjne
`== !=`	relacyjne
`&&`	logiczny
`||`	logiczny
`=`	przypisania

Język C - Wyrażenia logiczne

```
int  x = 0, y = 1, z = 2;
```

```
if (x == 0)
```
wynik: 1 (prawda)

```
if (x = 0)
```
wynik: 0 (fałsz) (!!!)

```
if (x != 0)
```
wynik: 0 (fałsz)

```
if (x =! 0)
```
wynik: 1 (prawda) (!!!)

```
if ( z > x + y )
```

```
if ( z > (x + y) )
```
wynik: 1 (prawda)

Język C - Wyrażenia logiczne

```
int  x = 0, y = 1, z = 2;
```

```
if ( x>2 && x<5 )
```

```
if ( (x>2) && (x<5) )
```

wynik: 0 (fałsz)

- Wyrażenia logiczne obliczane są od strony lewej do prawej
- Proces obliczeń kończy się, gdy wiadomo, jaki będzie wynik całego wyrażenia

```
if ( 2 < x < 5 )
```

wynik: 1 (prawda) (!!!)

Język C - Wyrażenia logiczne

- W przypadku sprawdzania czy wartość wyrażenia jest równa lub różna od zera można zastosować skrócony zapis
- Zamiast:

```
if (x==0)
```

```
if (x!=0)
```

można napisać:

```
if (!x)
```

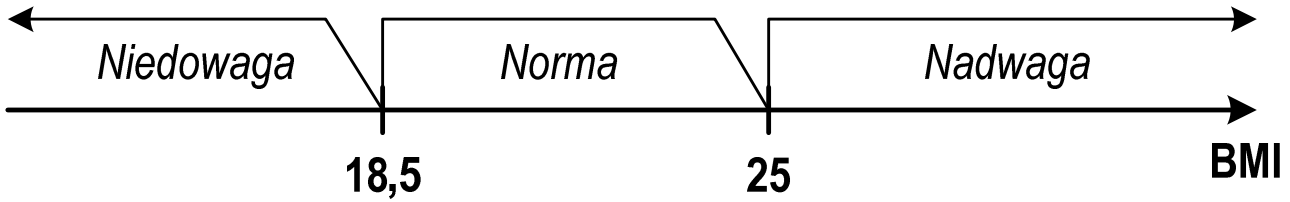
```
if (x)
```

Język C - BMI

- BMI - współczynnik powstały przez podzielenie masy ciała podanej w kilogramach przez kwadrat wzrostu podanego w metrach

$$BMI = \frac{masa}{wzrost^2}$$

- Dla osób dorosłych:
  - BMI < 18,5 - wskazuje na niedowagę
  - BMI ≥ 18,5 i BMI < 25 - wskazuje na prawidłową masę ciała
  - BMI ≥ 25 - wskazuje na nadwagę



Język C - BMI

```c
#include <stdio.h>

int main(void)
{
    double masa, wzrost, bmi;

    printf("Podaj mase [kg]:  "); scanf("%lf",&masa);
    printf("Podaj wzrost [m]: "); scanf("%lf",&wzrost);
    bmi = masa / (wzrost*wzrost);
    printf("bmi: %.2f\n",bmi);

    if (bmi<18.5)
        printf("Niedowaga\n");
    if (bmi>=18.5 && bmi<25)
        printf("Norma\n");
    if (bmi>=25)
        printf("Nadwaga\n");

    return 0;
}
```

```
Podaj mase [kg]:  84
Podaj wzrost [m]: 1.85
bmi: 24.54
Norma
```

Język C - BMI

- Zamiast trzech instrukcji if:

```
if (bmi<18.5)
    printf("Niedowaga\n");
if (bmi>=18.5 && bmi<25)
    printf("Norma\n");
if (bmi>=25)
    printf("Nadwaga\n");
```

można zastosować tylko dwie:

```
if (bmi<18.5)
    printf("Niedowaga\n");
else
    if (bmi<25)
        printf("Norma\n");
    else
        printf("Nadwaga\n");
```

Język C - Operator warunkowy

- Operator warunkowy skład się z dwóch symboli i trzech operandów

```
wyrażenie1 ? wyrażenie2 : wyrażenie3
```

- Najczęściej zastępuje proste instrukcje if-else

```
float akcyza, cena, pojemnosc;
```

```
if (pojemnosc <= 2000)
    akcyza = cena*0.031;    /*  3.1% */
else
    akcyza = cena*0.186;    /* 18.6% */
```

```
akcyza = pojemnosc <= 2000 ? cena*0.031 : cena*0.186;
```

Język C - Operator warunkowy

```
if (x < 0)
     y = -x;
 else
     y = x;
```

```
y = x < 0 ? -x : x;
```

- obliczenie modułu liczby x

```
if (a > b)
     max = a;
 else
     max = b;
```

```
max = a > b ? a : b;
```

- wyznaczenie max z dwóch liczb

- Operator warunkowy ma bardzo niski priorytet
- Niższy priorytet mają tylko operatory przypisania (=, +=, -=,…) i operator przecinkowy (,)

Język C - Operator warunkowy

- x studentów chce dojechać z akademika do biblioteki - ile taksówek powinni zamówić? (jedna taksówka może przewieźć 4 osoby)

```
#include <stdio.h>

int main(void)
{
    int x, taxi;

    printf("Podaj liczbe studentow: ");
    scanf("%d",&x);

    taxi = x / 4 + (x % 4 ? 1 : 0);

    printf("Liczba taxi: %d\n",taxi);

    return 0;
}
```

```
Podaj liczbe studentow: 23
Liczba taxi: 6
```

Język C - Instrukcja switch

- Instrukcja wyboru wielowariantowego switch

```
switch (wyrażenie)
{
    case wyrażenie_stałe: instrukcje;
    case wyrażenie_stałe: instrukcje;
    case wyrażenie_stałe: instrukcje;
    ...
    default: instrukcje;
}
```

- wyrażenie_stałe - wartość typu całkowitego, znana podczas kompilacji
  - stała liczbowa, np. 3, 5, 9
  - znak w apostrofach, np. 'a', 'z', '+'
  - stała zdefiniowana przez const lub #define

Język C - Instrukcja switch

- Program wyświetlający słownie liczbę z zakresu 1..5 wprowadzoną z klawiatury

```
#include <stdio.h>

int main(void)
{
      int liczba;

      printf("Podaj liczbe (1..5): ");
      scanf("%d",&liczba);
```

Język C - Instrukcja switch

```
switch (liczba)
{
   case 1: printf("Liczba: jeden\n");
           break;
   case 2: printf("Liczba: dwa\n");
           break;
   case 3: printf("Liczba: trzy\n");
           break;
   case 4: printf("Liczba: cztery\n");
           break;
   case 5: printf("Liczba: piec\n");
           break;
   default: printf("Inna liczba\n");
}
```

```
Podaj liczbe: 2
Liczba: dwa
```

```
Podaj liczbe: 0
Inna liczba
```

Język C - Instrukcja switch

```
switch (liczba)
{
   case 1:
   case 3:
   case 5: printf("Liczba nieparzysta\n");
           break;
   case 2:
   case 4: printf("Liczba parzysta\n");
           break;
   default: printf("Inna liczba\n");
}
```

```
Podaj liczbe: 2
Liczba parzysta
```

- Te same instrukcje mogą być wykonane dla kilku etykiet case

Język C - Instrukcja switch

```
switch (liczba)
{
   case 1: case 3: case 5:
           printf("Liczba nieparzysta\n");
           break;
   case 2: case 4:
           printf("Liczba parzysta\n");
           break;
   default: printf("Inna liczba\n");
}
```

```
Podaj liczbe: 2
Liczba parzysta
```

- Etykiety case mogą być pisane w jednym wierszu

Język C - Instrukcja switch

```
switch (liczba%2)
{
    case 1: case -1:
            printf("Liczba nieparzysta\n");
            break;
    case 0:
            printf("Liczba parzysta\n");
}
```

```
Podaj liczbe: 2
Liczba parzysta
```

- Część domyślna (default) może być pominięta

Język C - Instrukcja switch (bez break)

```
switch (liczba)
{
   case 1: printf("Liczba: jeden\n");
   case 2: printf("Liczba: dwa\n");
   case 3: printf("Liczba: trzy\n");
   case 4: printf("Liczba: cztery\n");
   case 5: printf("Liczba: piec\n");
   default: printf("Inna liczba\n");
}
```

```
Podaj liczbe: 2
Liczba: dwa
Liczba: trzy
Liczba: cztery
Liczba: piec
Inna liczba
```

- Pominięcie instrukcji break spowoduje wykonanie wszystkich instrukcji występujących po danym case (do końca switch)

Koniec wykładu nr 3

Dziękuję za uwagę!

(następny wykład: 29.03.2019)