1. Działania **modulo** i **div**

modulo - reszta z dzielenia (właściwie odejmowania)

9 **modulo** 3 -> 9 minus 3 = 6 -> 6 minus 3 = 3 -> 3 minus 3 = 0 -> wynik 0

14 **modulo** 6 -> 14 minut 6 = 8 -> 8 minut 6 = 2 -> wynik 2

UWAGA przy algorytmach: 45 modulo 100 = 45

5 modulo 2 w Python -> 4 % 2

div - dzielenie całkowite (bez reszty)

9 **div** 3 -> wynik 0 (reszta 0)

14 **div** 2 -> wynik 6 (reszta 2) (6+6+2=14)

UWAGA przy algorytmach: 45 div 100 = 0 (reszta 45), UWAGA: 45 div 45 = 0 (reszta 45)

Python -> 4 /2 (dzielenie z resztą)
Python -> 4 // 2 (dzielenie bez reszty)

2. System dziesiętny (inne nazwy: dziesiątkowy, decymalny)

Używa cyfry arabskich, właściwie europeizowanych cyfry hinduskich. Cyfry te stosowane są obecnie powszechnie na całym świecie. Są to kolejno znaczki: **0, 1, 2, 3, 4, 5, 6, 7, 8, 9.**

Jest to system pozycyjny - liczba zapisywana jest na poszczególnych pozycjach:

np. 137 (tak naprawdę $137_{(10)}$ - aby oznaczyć że chodzi o system dziesiętny)

	1 pozycja setek		3 pozycja dziesiątek		7 pozycja jedności
a więc:					
137 =	1 * 100	+	3 * 10	+	7 * 1
137 =	1 * $\mathbf{10}^2$	+	3 * $\mathbf{10}^1$	+	7 * $\mathbf{10}^0$
137 =	100	+	30	+	7
137 =	$137_{(10)}$ zwykle używając systemu dziesiętnego nie oznaczamy go				

UWAGA - Używamy liczby 10, ponieważ mamy dziesięć znaczków

2. System dwójkowy (inna nazwa: binarny)

Znaczki tylko dwa: **|** oraz **O** (pierwszy o wartości jeden, drugi o wartości zero)

||O|$_{(2)}$ =	| * $\mathbf{2}^3$	+	| * $\mathbf{2}^2$	+	O * $\mathbf{2}^1$	+	| * $\mathbf{2}^0$
||O|$_{(2)}$ =	| * 8	+	| * 4	+	O * 2	+	| * 1
||O|$_{(2)}$ =	8	+	4	+	0	+	1
||O|$_{(2)}$ =	$13_{(10)}$						

Popularnie zapisujemy, używając cyfr arabskich:

$1101_{(2)} = 13_{(10)}$ ale powinniśmy zawsze oznaczyć o jaki system nam chodzi, aby nie było pomyłek.

3. System szesnastkowy (inna nazwa: hexadecymalny)

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A (czyli 10), B (czyli 11), C (czyli 12), D (czyli 13), E (czyli 14), F (czyli 15)

Używamy aż 16 znaków. Ponieważ zabraknie cyfr arabskich, używamy też literek

$BBF1_{(16)}$ =	B * 16^3	+	B * 16^2	+	F * 16^1	+	1 * 16^0
$BBF1_{(16)}$ =	B * 4096	+	B * 256	+	F * 16	+	1 * 1
$BBF1_{(16)}$ =	11 * 4096	+	11 * 256	+	15 * 16	+	1 * 1
$BBF1_{(16)}$ =	45056	+	2816	+	240	+	1
$BBF1_{(16)}$ =	$48113_{(10)}$						

4. Własny wymyślony system pozycyjny (inna nazwa: _____ wpisz sobie ;-)

Używamy 4 znaków: & *(o wartości 0)*, # *(o wartości 1)*, $ *(o wartości 2)* oraz ! *(o wartości 3)*

$\$\$!\#_{(własny)}$ =	$ * 4^3	+	$ * 4^2	+	! * 4^1	+	# * 4^0
$\$\$!\#_{(własny)}$ =	$ * 4^3	+	$ * 4^2	+	! * 4^1	+	# * 4^0
$\$\$!\#_{(własny)}$ =	$ * 64	+	$ * 16	+	! * 4	+	# * 1
$\$\$!\#_{(własny)}$ =	2 * 64	+	2 * 16	+	3 * 4	+	1 * 1
$\$\$!\#_{(własny)}$ =	128	+	32	+	12	+	1
$\$\$!\#_{(własny)}$ =	$173_{(10)}$						

Istnieje wiele innych systemów. W informatyce korzystamy też często z systemu ósemkowego (0,1,2,3,4,5,6,7) lub czwórkowego (0,1,2,3).

5. Szybka zamiana małych z systemu dwójkowego na dziesiętny

a) Wypisujemy kolejne potęgi dwójki

1	2	4	8	16	32	64	128	256	512

b) Zamieniamy liczbę 137, a więc znajdujemy największą liczbę w tabeli mniejszą lub równą 137 i wpisujemy pod nią jedynkę.

1	2	4	8	16	32	64	128	256	512
							1		

c) Następnie do 128 dodajemy 64, jeśli jest więcej niż 137 to wpisujemy zero

1	2	4	8	16	32	64	128	256	512
						0	1		

d) Wpisujemy kolejno do lewej, ale jeśli suma będzie mniejsza wpisujemy jeden.

1	2	4	8	16	32	64	128	256	512
			1	0	0	0	1		

e) Następnie do 128 dodajemy 8 i kolejną liczbę, jeśli jest więcej niż 137 to wpisujemy zero, jesli mniej lub równo wpisujemy 1

1	2	4	8	16	32	64	128	256	512
1	0	0	1	0	0	0	1		

f) Spisujemy naszą liczbę w drugim wierszu od prawej: 10001001

1	2	4	8	16	32	64	128	256	512
1	0	0	1	0	0	0	1		

g) $10001001_{(2)}=137_{(10)}$

Uwaga - liczby nieparzyste w systemie dwójkowym zawsze kończą się na 1

6. Algorytm zamiany liczby na liczbę binarną wykorzystujące działanie DIV (dzielenie całkowite)

a) Dzielimy liczbę przez 2 (bo binarny) i zapisujemy wynik po liczbą, a po prawej resztę

137	|	1
68	|	

b) Powtarzamy dzielenie, aż w lewej kolumnie pojawi się zero

137	|	1
68	|	0
34	|	0
17	|	1
8	|	0
4	|	0
2	|	0
1	|	1
0		

<— tutaj uwaga na pomyłkę
podzielnik większy równy niż dzielona liczba

c) spisujemy liczbę od dołu: $10001001_{(2)}=137_{(10)}$

7. Ten sam algorytm zamiany liczby na liczbę szesnastkową

a) Dzielimy liczbę przez 16 (bo szesnastkowy) i zapisujemy wynik po liczbą, a po prawej resztę

$3007_{(10)} = BBF_{(16)}$

3007	|	15	->	F
187	|	11	->	B
11	|	11	->	B
0				

<— tutaj uwaga na pomyłkę

7. Ten sam algorytm zamiany liczby na liczbę wg wymyślonego systemu liczbowego z pkt.4

a) Dzielimy liczbę przez 4 (bo cztery znaki) i zapisujemy wynik po liczbą, a po prawej resztę

& *(o wartości 0)*, # *(o wartości 1)*, $ *(o wartości 2)* oraz ! *(o wartości 3)*

$173_{(10)} = \$\$!\#_{(własny)}$

173	|	1	->	#
43	|	3	->	!
10	|	2	->	$
2	|	2	->	$
0				

<— tutaj uwaga na pomyłkę