Pozycyjne systemy liczbowe i reprezentacja danych liczbowych w komputerze

Informatyka Europejczyka
Rozdział 2.3

Systemy liczbowe

* Systemem liczbowym nazywamy zbiór zasad określających sposób zapisywania i nazywania liczb.

* Pozycyjny system liczbowy to system, w którym wartość cyfry zależy od miejsca, w jakim znajduje się ona w danej liczbie. Miejsce to nazywamy pozycją.

Najważniejszymi systemami pozycyjnymi stosowanymi w informatyce są:

- System dwójkowy, czyli binarny
- System ósemkowy, czyli oktalny
- System szesnastkowy, czyli heksadecymalny

Systemy liczbowe

System Binarny

Korzysta z dwóch cyfr 0 i 1

System heksadecymalny

Podstawą systemu heksadecymalnego jest szesnaście , a więc w systemie tym korzystamy z szesnastu cyfr. Cyframi tego systemu są 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F, czyli

$A_{16}=10_{10}$

$B_{16}=11_{10}$

.....

$F^{16}=15_{10}$

System oktalny

System ten ma podstawę osiem, czyli cyframi są tutaj 0,1,2,3,4,5,6,7.

System rzymski

Najbardziej znanym systemem liczbowym, który nie jest pozycyjny, jest system rzymski, zaliczany jest do systemów zwanych addytywnymi. W przypadku systemu rzymskiego dotyczy to wielokrotności liczb 5 i 10. Dostępnych jest razem siedem znaków:

* I=1
* V=5
* X=10
* L=50
* C=100
* D=500
* M=1000

Przykład
MCCXCIX=1299

Ćwiczenia

Zamień liczby podane w systemie rzymskim na system dziesiętny:

* MXLVIII
* MCMLXXXIV
* CMXLVII
* DXLIX
* MMMCDI

Ćwiczenia

Zamień liczby podane w systemie dziesiętnym na system rzymski

* 2020
* 184
* 2876
* 3012
* 488

Zadanie

Wykorzystując zdobyte umiejętności napisz program realizujący konwersję liczb z systemu dziesiętnego na rzymski.

α	β	γ	δ	ε	ς	ζ	η	θ	ι
1	2	3	4	5	6	7	8	9	10
ι	κ	λ	μ	ν	ξ	o	π	φ	
10	20	30	40	50	60	70	80	90	
ρ	σ	τ	υ	φ	χ	ψ	ω	λ	
100	200	300	400	500	600	700	800	900	

$,\alpha$	$,\beta$	...
1000	2000	...
M	M^{β}	...
10000	20000	...

Ciekawostka - Grecki system liczbowy – liczbowy system addytywny używający greckiego alfabetu do reprezentacji liczb. Obecnie w Grecji jego zastosowanie ogranicza się do reprezentacji liczebników porządkowych oraz w sytuacjach analogicznych do stosowania rzymskiego zapisu w kulturze zachodniej.

Konwersje pozycyjnych systemów liczbowych

liczba	reszta
125	1
62	0
31	1
15	1
7	1
3	1
1	1
0	

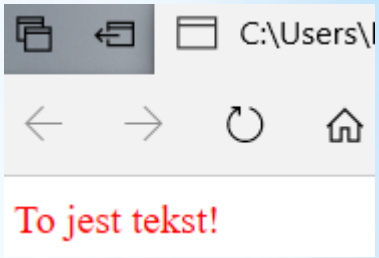
Liczba $(125)_{10} = (1111101)_2$

Liczba	reszta
459	11 = B
28	12 = C
1	1
0	

Liczba $(459)_{10} = (1CB)_{16}$

A	10
B	11
C	12
D	13
E	14
F	15

Kolory RGB



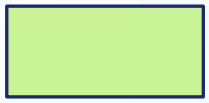
Z podstaw HTML znasz znacznik <COLOR> który zmienia kolor wybranego elementu np.:

To jest tekst!

Lub

To jest tekst!

kolor	Hex	Dec
R	ff	255
G	00	0
B	00	0



Opisz kolory w kodzie RGB
Edytor PAINT jest do tego idealny ;)

Konwersje między systemami niedziesiętnymi

* Najczęściej stosowanymi systemami liczbowymi w informatyce są: binarny, oktalny i heksadecymalny. Binarny wynika ze sposobu zapisu liczb w pamięci komputera (bit- 0,1)
* System ósemkowy i szesnastkowy to systemy, których podstawy są potęgami liczby 2. Istnieje więc możliwość bezpośredniej konwersji między tymi systemami.

Przykład (bin na okt.)

* Liczba binarna zapisana na trzech miejscach ma wartość w przedziale [0,111] co w systemie dziesiętnym wynosi [0,7]
* Wykonamy konwersję liczby 1011101111_2 na system oktany. Najpierw musimy pogrupować liczbę po trzy cyfry zaczynając od prawej

1 011 101 111

binarna	oktalna
1_2	$1*2^0=1$
011_2	$1*2^1+1*2^0=3$
101_2	$1*2^2+0*2^1+1*2^0=5$
111_2	$1*2^2+1*2^1+1*2^0=7$

Wynik: 1011101111_2 to 1357_8

Przykład (bin na hex.)

Grupujemy liczbę na cztery cyfry [0,1111] czyli dziesiętnie [0,15(F)]

Liczba: 110011011_2

1 1001 1011

binarna	hex
1	$1*2^0=1$
1001	$1*2^3+0*2^2+0*2^1+1*2^0=9$
1011	$1*2^3+0*2^2+1*2^1+1*2^0=B(11)$

Po konwersji $110011011_2 = 19B_{16}$

Ćwiczenia

Zamień podane liczby całkowite z systemu dziesiętnego na ósemkowy i szesnastkowy z wykorzystaniem systemu dwójkowego.

* 523
* 458
* 399
* 878
* 1001
* 1112
* 2056

Operacje arytmetyczne - sumowanie

1	1	0	0
1	0	1	0
0 dalej 1	1	1	0

Sumujemy $11011_2 + 111110_2$

1	*1*	*1*	*1*	*1*		
		1	1	0	1	1
+	1	1	1	1	1	0
1	0	1	1	0	0	1

Suma czterech liczb binarnych

1	2	2	2	3	2	1	
		1	1	1	1	1	1
				1	1	1	0
			1	0	1	1	1
+		1	1	0	1	1	1
1	**0**	**0**	**1**	**1**	**0**	**1**	**1**

$$111111_2+1110_2+10111_2+110111_2=10011011_2$$

Odejmowanie w systemie binarnym

Problem pojawia się w przypadku , gdy chcemy wykonać operację odejmowania , a liczba od której odejmujemy, jest zbyt mała. Należy wówczas pobrać wartość z lewej strony. Każda jedynka pobrana z lewej strony zamieniana jest na podstawę systemu, czyli dwa i wykonujemy odejmowanie

	0	2			
1	~~1~~	0	1	1	0
-		1	0	1	0
1	**0**	**1**	**1**	**0**	**0**

$$110110_2 - 1010_2 = 101100_2$$

Trochę trudniejsze odejmowanie

	1	1	1	1	
0	~~2~~	~~2~~	~~2~~	~~2~~	~~2~~
~~1~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~
-			1	1	1
	1	**1**	**0**	**0**	**1**

$100000_2-111_2=11001_2$

Mnożenie liczb binarnych

Mnożenie liczb binarnych jest działaniem, które łączy w sobie operacje mnożenia i dodawania.

				1	1	0	1	1	1
			x			1	0	1	1
				1	1	0	1	1	1
			1	1	0	1	1	1	
+	1	1	0	1	1	1			
1	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**0**	**1**

$$110111_2 * 1011_2 = 1001011101_2$$

Ćwiczenia

Wykonaj w systemie binarnym następujące działania

$10110_2 + 1101_2 * 111_2 =$

$111000_2 + 11_2 + 101100_2 * 110_2 =$

Liczby ujemne w systemie binarnym

Wartości ujemne zapisuje się, używając kodu uzupełnieniowego do dwóch, zwanego **kodem U2**. Ogólny zapis liczby w kodzie U2:

$$y = \begin{cases} x & \text{dla } x >= 0 \\ 2^n + x & \text{dla } x < 0 \end{cases}$$

x – liczba, którą chcemy zapisać w kodzie U2.

n – liczba bitów przeznaczonych do zapisania kodowanej liczby.

y – liczba x zapisana za pomocą kodu U2.

System U2 przeznaczony do przechowywania liczb całkowitych dodatnich i ujemnych. W języku C++ zmienne typu int, long long, czy char używają tego systemu do przechowywania wartości. Żeby można było wykonywać operacje w tym systemie, należy określić na ilu bitach będziemy operować.

Przykład

Przekonwertujemy liczbę -56 czyli x=-56

Potrzebujemy 8 bitów czyli n=8

Najstarszy bit w zapisie oznacza wartość znaku (1-liczba ujemna, 0-liczba dodatnia)

czyli wartość kodowanej liczby x zawiera się w przedziale $[-2^{n-1},2^{n-1}]$->dziesiętnie [-128,128]

Dysponujemy 1 bajtem – czyli 8 bitów przeznaczonych na zapis liczby.

n=8

x= -56

Korzystając ze wzoru, wyznaczymy wartość liczby x=-56 w kodzie U2. Obliczymy wg. wzoru:

$y=2^8+(-56)=256-56=200_{10}=11001000_{u2}$

Znaczący bit (1- liczba ujemna, 0- liczba dodatnia)

Inaczej U2 liczby dodatnie

Zamiana liczb dodatnich

* Dla przykładu przedstawimy liczbę 50
* na ośmiu bitach w **U2.**
* Najpierw zamieniamy ją na system dwójkowy:
* $50 = (110010)_2$
* Następnie z lewej strony dopełniamy zerami tak, aby w sumie otrzymać osiem bitów i w rezultacie otrzymujemy:
* $50 = (00110010)_{U2}$

Inaczej U2 liczby ujemne

* Teraz zamieńmy liczbę −50 na system **U2**. Tu algorytm jest bardziej skomplikowany.
* W pierwszym kroku wyznaczamy wartość bezwzględną z tej liczby:
* |−50|=50
* W drugim kroku otrzymaną liczbę zamieniamy na postać binarną:
* $50 = (110010)_2$
* W trzecim kroku przedstawiamy ją na ośmiu bitach:
* 50 = 00110010
* W czwartym kroku negujemy wszystkie bity (każdy bit zamieniamy na przeciwny: zero na jedynkę, jedynkę na zero):
* (00110010) = ~11001101
* Na końcu zwiększamy otrzymaną postać o **1**:
* 11001101 + 1 = 11001110

* W ten sposób otrzymaliśmy liczbę −50
* zapisaną na **ośmiu bitach** w systemie **U2:**

* $-50 = (11001110)_{U2}$

Przelicz wcześniej podaną liczbę -56 tym sposobem na U2.

Zapisz podane liczby ujemne w kodzie U2

* -108_{10} dla n=8 bitów
* -241_{10} dla n=16 bitów

Zamiana z U2 na system dziesiętny

* Najbardziej znaczący bit (ten który stoi po lewej stronie) określa znak liczby. Jeśli jest to **jedynka,** to liczba jest **ujemna**, w przeciwnym razie jest ona **dodatnia.** Dla przykładu posłużmy się otrzymaną wyżej postacią:
* $(11001110)_{U2}$
* Teraz odliczamy kolejne potęgi dwójki począwszy od strony lewej. Przy pierwszym bicie stoi 2^0

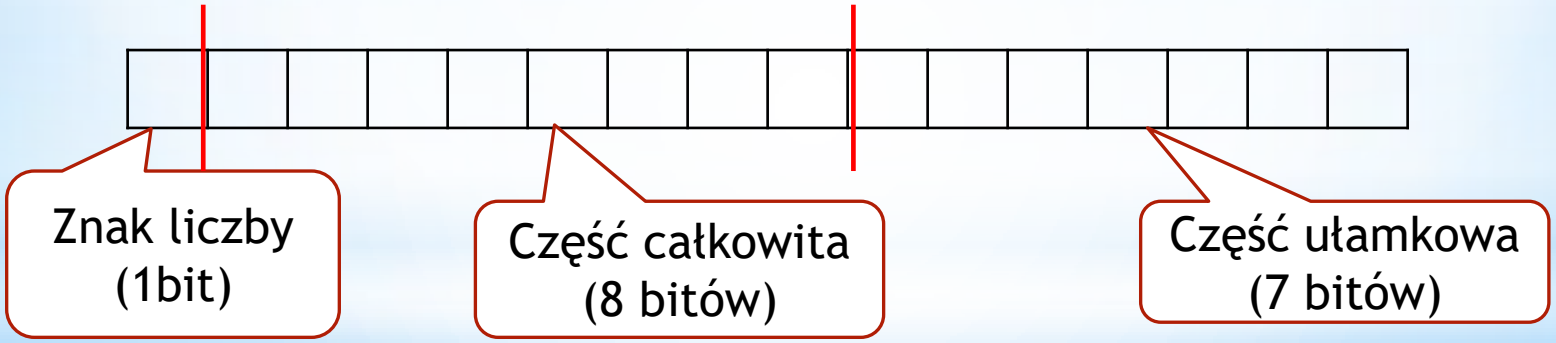
, przy następnym 2^1, ..., natomiast przy ostatnim -2^7.

Teraz dodajemy tylko te potęgi liczby dwa, które stoją nad cyfrą **1**:

* $-2^7+2^6+2^3+2^2+2^1=-128+64+8+4+2=-50$

Stałopozycyjna reprezentacja liczb

Stałopozycyjna reprezentacja liczb charakteryzuje się stałym położeniem przecinka, który oddziela część całkowitą od części ułamkowej zapisywanej liczby. Liczba zapisywana będzie dokładna jeżeli nie wykroczy poza zakres przeznaczony do zapisu.



Konwertując liczby z systemu binarnego na decymalny, mnożymy kolejne cyfry tej liczby przez potęgi dwójki. W części ułamkowej mnożnikiem są ujemne potęgi liczby 2.

Przykład

Zapiszemy liczbę rzeczywistą $101111{,}01101_2$ w systemie dziesiętnym.

1	0	1	1	1	1	,	0	1	1	0	1
2^5	2^4	2^3	2^2	2^1	2^0	,	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}

$$1*2^5+0*2^4+1*2^3+1*2^2+1*2^1+1*2^0+0*2^{-1}+1*2^{-2}+1*2^{-3}+0*2^{-4}+1*2^{-5} = 47\frac{13}{32}_{10}$$

Zamiana liczby ułamkowej z dziesiętnego na dwójkowy

Wykonywany jest przez mnożenie części ułamkowej przez 2 tak długo, aż w części ułamkowej uzyskamy 0 lub zauważymy, że wynikiem jest ułamek nieskończony.

Przekonwertujemy liczbę ułamkową $0,1875_{10}$ na system binarny

0	1875
0	365
0	75
1	5
1	0

Mnożymy część ułamkową przez 2 aż uzyskamy 0.

$0,1875_{10}=0,0011_2$

Ułamek nieskończony

Przekonwertujemy liczbę $0,2_{10}$ na system binarny

0	2
0	4
0	8
1	6
1	2
0	4
0	8
1	6
1	2
0	4
.....	

Sekwencja „0011” powtarza się, więc wynikiem jest ułamek okresowy $0,(0011)_2$.

Ćwiczenia

* Przekonwertuj podane liczby rzeczywiste na system dziesiętny
  - $10100,11101_2$
  - $0,0111011_2$
* Przekonwertuj podane liczby rzeczywiste na system binarny
  - $852,6875_{10}$
  - $620,09375_{10}$