SYSTEM BINARNY

ZAMIANA SYSTEMÓW ZAPISU LICZBY

Niech x będzie zmienną typu `unsigned int` (32-bitową). Zakładając, że dostępne są jedynie operacje na bitach, a dokładnie nie mamy możliwości skorzystania z operacji dodawania, odejmowania, mnożenia i dzielenia, zaimplementuj procedurę, która wypisze na ekranie zapis binarny (wartości) zmiennej x. •••

Niech $b = d_r d_{r-1} \dots d_1 d_0$ będzie zapisem liczby w systemie dwójkowym. Zamiana zapisu liczby b na zapis w systemie dziesiętnym odbywa się poprzez wykonanie dodawania oraz potęgowania w następującym wyrażeniu:

$$d_r \cdot 2^r + d_{r-1} \cdot 2^{r-1} \dots d_1 \cdot 2^1 + d_0 \cdot 2^0,$$

przy czym operacje dodawania i potęgowania wykonywane są w systemie o podstawie 10.[1]

Z drugiej strony, niech b będzie liczbą zapisaną w systemie dziesiętnym. Zamiana zapisu liczby b na zapis w systemie dwójkowym odbywa się poprzez rozłożenie b na sumę kolejnych potęg dwójki:

$$b = d_r \cdot 2^r + d_{r-1} \cdot 2^{r-1} \dots d_1 \cdot 2^1 + d_0 \cdot 2^0,$$

gdzie $d_i \in \{0, 1\}$. Wówczas $b = (d_r d_{r-1} \dots d_1 d_0)_2$. Co więcej, owe rozłożenie na kolejne potęgi dwójki można wykonać algorytmicznie. A dokładnie, dokonujemy kolejnych dzieleń liczby b w sposób całkowity przez 2 (w systemie dziesiętnym) i zapamiętujemy reszty z tegoż dzielenia. Reszty te, zapisane w odwrotnej kolejności, utworzą nam zapis binarny liczby b.

Zapisz liczbę $(10010)_2$ w systemie dziesiętnym. **◀ PRZYKŁAD**

$(10010)_2 = 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = (16)_{10} + (0)_{10} + (2)_{10} + (0)_{10} = (18)_{10}$

Zapisz liczbę $(81)_{10}$ w systemie dwójkowym. **◀ PRZYKŁAD**

Podejście algorytmiczne daje nam następującą tabelę ilorazów i reszt:

liczba	iloraz	reszta
81	40	1
40	20	0
20	10	0
10	5	0
5	2	1
2	1	0
1	0	1

Koleje reszty, czytane „od dołu", są następujące: $1, 0, 1, 0, 0, 0, 1$, a zatem $(81)_{10} = (1010001)_2$. Oczywiście podejście niealgorytmiczne da nam ten sam wynik:

$(81)_{10} = (64)_{10} + (16)_{10} + (1)_{10} = 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = (1010001)_2$

ZADANIE 1.1. Przedstaw liczby $(111101)_2$ oraz $(1011110)_2$ w systemie dziesiętnym.

ZADANIE 1.2. Przedstaw liczby 169 oraz 411 w systemie dwójkowym.

[1] Dla ułatwienia będziemy przyjmować, że zapis $(x_r \dots x_0)_k$ oznacza zapis w systemie o podstawie k, przy czym w dalszych rozważaniach, dla uproszczenia notacji, będziemy zakładać, że brak indeksu przy zapisie liczby oznacza zapis w systemie dziesiętnym.

ZWIĘKSZANIE LICZBY O JEDEN W SYSTEMIE DWÓJKOWYM

Niech x będzie zmienną typu `unsigned int` (32-bitową). Zakładając, że dostępne są jedynie operacje na bitach, a dokładnie nie mamy możliwości skorzystania z operacji dodawania, odejmowania, mnożenia i dzielenia, zaimplementuj procedurę, która powiększy wartość zmiennej x o jeden. •••

Algorytm zwiększania liczby o jeden w systemie dwójkowym

1. Wskaż ostatni bit rozważanej liczby.
2. Powtarzaj, co następuje:
   2.a. Jeżeli wskazywany bit to „0”, to zamień go na „1”; Koniec.
   2.b. W przeciwnym przypadku zamień go na „0” i wskaż kolejny bit na lewo; jeżeli nie ma następnego bitu w lewo, to wstaw „1”; Koniec.

Prześledź działanie algorytmu dodawania jedynki dla liczb 10010 oraz 101011. ◀ **PRZYKŁAD**

a) $10010 + 1 = 10011$,
ponieważ $1001\underline{0} \to (= 0) \to 10011$ (Koniec).

b) $101011 + 1 = 101100$,
ponieważ $10101\underline{1} \to (= 1) \to 1010\underline{1}0 \to (= 1) \to 101\underline{0}00 \to (= 0) \to 101100$ (Koniec).

ZADANIE 1.3. Prześledź działanie algorytmu dodawania jedynki dla liczb 111110, 10011 oraz 111111.

PORÓWNYWANIE LICZB W SYSTEMIE DWÓJKOWYM

Niech x i y będą dwoma zmiennymi typu `unsigned int` (32-bitowe). Zakładając, że dostępne są jedynie operacje na bitach, a dokładnie nie mamy możliwości skorzystania z operatorów $<$ (mniejsze od) oraz $>$ (większe od), a także operacji dodawania, odejmowania, mnożenia i dzielenia, zaimplementuj procedurę, która wypisze wartość $\max(x, y)$ w postaci dziesiętnej. •••

Algorytm porównywania liczb w systemie dwójkowym

1. Jeżeli liczby są różnej długości, to większą jest liczba o dłuższym zapisie.
2. Jeżeli liczby są tej samej długości, to porównujemy bit po bicie od lewej strony do prawej:
   2.a. Jeżeli bity są takie same, to przechodzimy do następnego bitu w prawo;
   2.b. Jeżeli bity są różne, to większą jest liczba o większym bicie na rozważanej pozycji; Koniec.
3. Jeżeli wszystkie bity są takie same, to porównywane liczby są równe i Koniec.

Prześledź działanie algorytmu porównywania liczb dla podanych niżej par liczb. ◀ **PRZYKŁAD**

a) 101101 oraz 11110

Jako że pierwsza liczba jest liczbą 6-bitową, a druga — 5-bitową, otrzymujemy, że $101101 > 11110$.

b) 1011101 oraz 1011001

$(\underline{1}011101 \ ? \ \underline{1}011001) \to (=) \to (1\underline{0}11101 \ ? \ 1\underline{0}11001) \to (=) \to$
$(10\underline{1}1101 \ ? \ 10\underline{1}1001) \to (=) \to (101\underline{1}101 \ ? \ 101\underline{1}001) \to (=) \to$
$(1011\underline{1}01 \ ? \ 1011\underline{0}01) \to (>) \to$, a zatem $1011101 > 1011001$.

ZADANIE 1.4. Prześledź działanie algorytmu porównywania liczb dla następujących par liczb.

a) 1111 oraz 10001;
b) 11010 oraz 10111;
c) 1111001 oraz 1111011.

DODAWANIE LICZB W SYSTEMIE DWÓJKOWYM

Niech x i y będą dwoma zmiennymi typu `unsigned int` (32-bitowe), których wartość jest nie większa od $(2147483648)_{10}$ $(= 2^{31})$. Zakładając, że dostępne są jedynie operacje na bitach, a dokładnie nie mamy możliwości skorzystania z operacji dodawania, odejmowania, mnożenia i dzielenia, zaimplementuj procedurę, która wypisze wartość $x + y$ w postaci dziesiętnej. •••

Algorytm dodawania liczb w systemie dwójkowym

- Aby dodać do siebie dwie liczby zapisane w systemie dwójkowym, dodajemy bit po bicie od prawej do lewej, dodając jednocześnie w każdym z kroków bity przeniesienia z poprzedniej kolumny.

Wykonaj poniższe dodawania w systemie dwójkowym. **◀ PRZYKŁAD**

a) $10101 + 111$

$10101 + 111 = 11100$, ponieważ

$$\begin{array}{r} \ 111 \\ \hline 10101 \\ +\quad 111 \\ \hline 11100 \end{array}.$$

b) $111 + 111 + 111 + 111 + 111$

Rozważmy najpierw sumę $1 + 1 + 1 + 1 + 1 + 1$ ostatnich bitów. Jej wartość zapisana w systemie dziesiętnym to 6_{10}, a w binarnym – $(110)_2$. Zatem w tym przypadku bitami przeniesienia są bity 11, natomiast ostatni bit w zapisie szukanej liczby to 0.

Rozważmy teraz sumę $1 + 0 + 1 + 1 + 1 + 0$ przedostatnich bitów powiększoną jeszcze o ostatni bit przeniesienia, czyli o 1. Jej wartość zapisana w systemie dziesiętnym to 5_{10}, a w binarnym – $(101)_2$. A zatem w tym przypadku bitami przeniesienia są bity 10. Itd.

Otrzymujemy ostatecznie, że $111 + 101 + 111 + 111 + 111 + 101 = 100110$.

$$\begin{array}{r} \ 1 \\ \hline 1 \\ \hline 11 \\ \hline 10 \\ \hline 11 \\ \hline 111 \\ 101 \\ 111 \\ 111 \\ 111 \\ +\quad 101 \\ \hline 100110 \end{array}.$$

ZADANIE 1.5. Wykonaj następujące dodawania:

a) $10011 + 1100$;
b) $1111 + 1110$;
c) $110111 + 110011$;
d) $101 + 111 + 111$;
e) $1011 + 1011 + 111$.

ODEJMOWANIE LICZB W SYSTEMIE DWÓJKOWYM

Niech x i y będą dwoma zmiennymi typu `unsigned int` (32-bitowe) takimi, że wartość zmiennej x jest nie mniejsza od wartości zmiennej y. Zakładając, że dostępne są jedynie operacje na bitach, a dokładnie nie mamy możliwości skorzystania z operacji dodawania, odejmowania, mnożenia i dzielenia, zaimplementuj procedurę, która wypisze wartość $x - y$ w postaci dziesiętnej. •••

Algorytm odejmowania liczb w systemie dwójkowym

- Aby odjąć od siebie dwie liczby zapisane w systemie dwójkowym, odejmujemy bit po bicie od prawej do lewej, a w przypadku, gdy trzeba odjąć bit większy od mniejszego, „pożyczamy" jedynkę z następnej niezerowej (na lewo) pozycji (o ile jest to jeszcze możliwe).

Wykonaj poniższe odejmowania w systemie dwójkowym. **◀ PRZYKŁAD**

a) $10101 - 111$

$$\begin{array}{r} {\scriptstyle 012} \\ \hline 1002 \\ 10101 \\ -\ \ 111 \\ \hline 1110 \end{array}$$

$10101 - 111 = 1110$, ponieważ

b) $111000 - 11111$

$$\begin{array}{r} {\scriptstyle 02} \\ \hline 102 \\ 110112 \\ \hline 111000 \\ -\ \ 11111 \\ \hline 11001 \end{array}$$

oraz $111000 - 11111 = 11001$, ponieważ.

ZADANIE 1.6. Wykonaj następujące odejmowania:

a) $110111 - 110011$;
b) $10011 - 1100$;
c) $1010001 - 101110$;
d) $1011100 - 1010111$.

MNOŻENIE I DZIELENIE LICZB W SYSTEMIE DWÓJKOWYM

Niech x i y będą dwoma zmiennymi typu `unsigned int` (32-bitowe) takimi, że wartość zmiennej x jest podzielna przez wartość zmiennej y. Zakładając, że dostępne są jedynie operacje na bitach, a dokładnie nie mamy możliwości skorzystania z operacji dodawania, odejmowania, mnożenia i dzielenia, zaimplementuj procedurę, która wypisze wartość ilorazu x/y w postaci dziesiętnej. •••

Algorytm mnożenia liczb w systemie dwójkowym

- Aby pomnożyć dwie liczby (zapisane dwójkowo), mnożymy pierwszą liczbę przez poszczególne bity drugiej, a otrzymane wyniki, każdy kolejno przesunięty o jedną kolumnę w lewo, na koniec sumujemy.

◀ PRZYKŁAD

$10101 \cdot 101 = 1101001$, ponieważ

$$\begin{array}{r} 10101 \\ \cdot\quad 101 \\ \hline 10101 \\ 00000 \\ 10101 \\ \hline 1101001 \end{array}.$$

Uwaga. Aby ułatwić sobie mnożenie liczb, mając na uwadze przemienność mnożenia, wygodniej jest mnożyć liczbę o większej liczbie jedynek przez liczbę o mniejszej liczbie jedynek, czyli np. lepiej rozpatrywać iloczyn $1011 \cdot 10001$ niż iloczyn $10001 \cdot 1011$.

ZADANIE 1.7. Wykonaj następujące mnożenia:

a) $10011 \cdot 1100$;
b) $101 \cdot 111$;
c) $1111 \cdot 111$;
d) $111000 \cdot 111$.

◀ PRZYKŁAD

$1101001 : 101 = 10101$, ponieważ

$$\begin{array}{r|l} & 10101 \\ \hline 1101001 & : 101 \\ 101 & \\ \hline ==110 & \\ 101 & \\ \hline ==101 & \\ 101 & \\ \hline ==0 & \end{array}.$$

Uwaga. Liczba jest podzielna przez 2^i, jeśli w jej zapisie binarnym występuje na końcu i bitów równych 0.

ZADANIE 1.8. Wykonaj następujące dzielenia:

a) $11000 : 1000$;
b) $100011 : 101$;
c) $1010001 : 1001$;
d) $110001 : 111$;
e) $1001101 : 111$.

SYSTEMY O PODSTAWIE BĘDĄCEJ POTĘGĄ DWÓJKI

W systemie szesnastkowym używa się następujących „cyfr”: $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, \mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}, \mathrm{E}, \mathrm{F}$. Przyjmijmy notację, że liczbę zapisaną w systemie szesnastkowym będzie poprzedzać znak dolara \$.

Zamień zapis liczby \$A1 z szesnastkowego na dziesiętny. **◀ PRZYKŁAD**

$\$A1 = 10 \cdot 16^1 + 1 \cdot 16^0 = 160 + 1 = 161$.

Zamień zapis liczby 320 z dziesiętnego na szesnastkowy. **◀ PRZYKŁAD**

320 = \$140, ponieważ

liczba	iloraz	reszta
320	20	0
20	1	4
1	0	1

.

ZADANIE 1.9. Zamień zapis z szesnastkowego na dziesiętny następujących liczb:

a) \$A91;
b) \$C2;
c) \$FCA.

ZADANIE 1.10. Zamień zapis z dziesiętnego na szesnastkowy następujących liczb:

a) 199;
b) 541;
c) 855.

Zamień zapis 10111100 z binarnego na ósemkowy i szesnastkowy. **◀ PRZYKŁAD**

$10111100 = (274)_8$, ponieważ

010	111	100
2	7	4

.

Analogicznie, 10111100 = \$BC, ponieważ

1011	1100
B	C

.

ZADANIE 1.11. Zamień zapis z binarnego na ósemkowy oraz szesnastkowy następujących liczb:

a) 100010;
b) 1011101;
c) 111110110.

Zamień zapis \$A1 z szesnastkowego na binarny oraz ósemkowy. **◀ PRZYKŁAD**

\$A1 = 10100001, ponieważ

A	1
1010	0001

.

Natomiast aby otrzymać zapis ósemkowy, przekształcamy otrzymany wyżej zapis binarny 10100001 w sposób opisany w poprzednim przykładzie, otrzymując $\$A1 = 10\ 100\ 001 = (241)_8$.

ZADANIE 1.12. Zamień zapis z szesnastkowego na binarny oraz ósemkowy następujących liczb:

a) \$C2;
b) \$A91;
c) \$FCA.

Liczbę $(175)_8$ przedstaw w postaci dwójkowej i dziesiętnej. **◀ PRZYKŁAD**

$(175)_8 = 1111101$, ponieważ

1	7	5
1	111	101

.

Następnie $(175)_8 = 1 \cdot 8^2 + 7 \cdot 8^1 + 5 \cdot 8^0 = (64)_{10} + (56)_{10} + (5)_{10} = (125)_{10}$.

ZADANIE 1.13. Zamień zapis z ósemkowego na dwójkowy i dziesiętny następujących liczb:

a) $(713)_8$;
b) $(1027)_8$;
c) $(37700)_8$.

ZADANIE 1.14. Zamień zapis $(90)_{10}$ oraz $(160)_{10}$ z dziesiętnego na ósemkowy.

◀ PRZYKŁAD

Pewna liczba x zapisana w zapisie ósemkowym ma 5 cyfr.
Ile będzie miała ona cyfr w zapisie szesnastkowym?

Liczba x, która w zapisie ósemkowym ma 5 cyfr, należy do zbioru

$$\{(10000)_8, (10001)_8, \ldots, (77776)_8, (77777)_8\}.$$

Jako że 8 i 16 są potęgami dwójki, w łatwy sposób możemy zamienić zapisy $(10000)_8$ i $(77777)_8$ w zapis dwójkowy, z którego w równie łatwy sposób otrzymamy zapis szesnastkowy.

$$(10000)_8 = 1\ 000\ 000\ 000\ 000 = 1\ 0000\ 0000\ 0000 = \$1000,$$

$$(77777)_8 = 111\ 111\ 111\ 111\ 111 = 111\ 1111\ 1111\ 1111 = \$7FFF.$$

A zatem liczba x w zapisie szesnastkowym będzie miała 4 cyfry.

ZADANIE 1.15. Pewna liczba x zapisana w zapisie czwórkowym ma 7 cyfr.

a) Ile będzie miała ona cyfr w zapisie ósemkowym?
b) Ile będzie miała ona cyfr w zapisie dwójkowym?

ZADANIE 1.16. Pewna liczba x zapisana w zapisie ósemkowym ma 5 cyfr. Ile będzie miała cyfr ona w zapisie czwórkowym?

UŁAMKI W SYSTEMIE DWÓJKOWYM

Zapis $(0.d_1d_2\dots d_r)_2$ w systemie o podstawie k oznacza liczbę $d_1 \cdot k^{-1} + d_2 \cdot k^{-2} \dots d_r \cdot k^{-r}$.

◀ PRZYKŁAD

$(0.11)_2 = 1 \cdot 2^{-1} + 1 \cdot 2^{-2} = (0.75)_{10}$

ZADANIE 1.17. Przedstaw ułamki $(0.1101)_2, (0.0011)_2$ oraz $(0.01101)_2$ w postaci dziesiętnej.

Aby zamienić zapis ułamka (<1) z systemu dziesiętnego na system o podstawie k, należy rozważaną część ułamkową kolejno mnożyć (w systemie dziesiętnym) przez k, wypisując kolejno otrzymywane części całkowite do momentu, aż część ułamkowa będzie równa 0.

◀ PRZYKŁAD

Zamień zapis $(0.8125)_{10}$ z dziesiętnego na binarny.

	·2
część całkowita	część ułamkowa
0.	0.8125
1	0.625
1	0.25
0	0.5
1	0.0

Otrzymujemy ostatecznie, że $0.8125 = (0.1101)_2$.

ZADANIE 1.18. Zamień zapis z dziesiętnego na binarny następujących liczb:

a) 0.5625;
b) 0.78125;
c) 0.15625;
d) 0.328125;
e) 7.5625;
f) 11.15625;
g) 13.328125.

◀ **PRZYKŁAD**

Wykonaj następujące działania:

a) $(56)_7 + (43)_7 = (132)_7$, ponieważ

$$\begin{array}{r} {\scriptstyle 11} \\ \hline 56 \\ + \;\; 43 \\ \hline 132 \end{array}$$

b) $(41)_5 - (24)_5 = (12)_5$, ponieważ

$$\begin{array}{r} {\scriptstyle 36} \\ \hline 41 \\ - \;\; 24 \\ \hline 12 \end{array}$$

c) $(13)_6 \cdot (4)_6 = (100)_6$, ponieważ

$$\begin{array}{r} {\scriptstyle 2} \\ \hline 13 \\ \cdot \;\; 4 \\ \hline 100 \end{array}$$

ZADANIE 1.19. Wykonaj następujące działania:

a) $(13)_4 + (33)_4$;
b) $(122)_3 + (122)_3 + (122)_3$;
c) $(456)_7 + (223)_7$;
d) $(302)_4 - (13)_4$;
e) $(4236)_7 - (2543)_7$;
f) $(13)_4 \cdot (3)_4$;
g) $(135)_7 \cdot (642)_7$.

ZADANIE 1.20. Liczby $(201)_3$ i $(241)_7$ przedstaw w postaci dziesiętnej.

ZADANIE 1.21. Liczby $(80)_{10}$ i $(120)_{10}$ przedstaw w postaci trójkowej i siódemkowej.

REPREZENTACJA LICZB W KOMPUTERZE

Zmienne typu `short int` przechowywane są zwykle w dwóch bajtach, czyli 16 bitach.[2] Pierwszy bit określa znak liczby – jeżeli wynosi on 0, to liczba jest dodatnia, w przeciwnym razie liczba jest ujemna.

- Jeżeli liczba jest dodatnia, to pozostałe piętnaście bitów stanowi zapis binarny tej liczby.
- Liczby ujemne przechowywane są w tak zwanym systemie uzupełnieniowym U2, tzn. liczba ujemna o wartości bezwzględnej x przedstawiana jest jako liczba $2^{16} - x$ w postaci binarnej.

◀ PRZYKŁAD

Rozważmy liczbę 82. Jest ona liczbą dodatnią.
Jej zapis w postaci binarnej to 1010010. Zatem jest ona przechowywana w postaci:

znak	15 bitów
0	000 0000 0101 0010

, czyli ostatecznie $82 = (0000\ 0000\ 0101\ 0010)_{\texttt{int}}$.

Rozważmy teraz liczbę -82. Jest ona liczbą ujemną. Zapis jej wartości bezwzględnej, czyli 82, w postaci binarnej to 1010010. Zatem jest ona przechowywana w postaci:

$$\begin{array}{rr} & 1\ \underbrace{0000\ 0000\ 0000\ 0000}_{\text{16 zer}} \\ - & \qquad\qquad 1010010 \\ \hline & 1111\ 1111\ 1010\ 1110 \end{array}$$

, czyli ostatecznie $-82 = (1111\ 1111\ 1010\ 1110)_{\texttt{int}}$.

Zapis liczby -82 w postaci `int` można również uzyskać następująco:

- Zapis jej wartości bezwzględnej na 16 bitach „zaprzeczamy” i dodajemy „1”;

$$\begin{array}{rr} & 0000\ 0000\ 0101\ 0010 \\ \hline & 1111\ 1111\ 1010\ 1101 \\ + & \qquad\qquad\qquad 1 \\ \hline & 1111\ 1111\ 1010\ 1110 \end{array}$$

- Bądź też odejmujemy „1” od jej wartości bezwzględnej na 16 bitach i „zaprzeczamy”.

$$\begin{array}{rr} & 0000\ 0000\ 0101\ 0010 \\ - & \qquad\qquad\qquad 1 \\ \hline & 0000\ 0000\ 0101\ 0001 \\ \hline & 1111\ 1111\ 1010\ 1110 \end{array}$$

◀ PRZYKŁAD

Rozważmy liczbę $(0000000001010010)_{\texttt{int}}$. Jej „rozkodowywanie” przebiega analogicznie. Jako że pierwszy bit jest równy `0`, zatem jest to liczba dodatnia. A zatem, jako że jej zapis binarny to 1010010, zakodowaną liczbą jest 82.

Rozważmy teraz liczbę $(1111111110101110))_{\texttt{int}}$. Jako że jej pierwszy bit jest równy `1`, zatem jest to liczba ujemna. Wyznaczamy ją następująco:

- Sposób 1:

$$\begin{array}{rr} & 1\ 0000\ 0000\ 0000\ 0000 \\ - & \quad 1111\ 1111\ 1010\ 1110 \\ \hline & \quad 0000\ 0000\ 0101\ 0010 \end{array}$$

, co daje nam 82, ale że pierwszy bit był równy 1, zatem kodowaną liczbą jest -82.

[2] Analogicznie przechowuje się np. zmienne typu `int`, tylko że w czterech bajtach, czyli 32 bitach.

■ Sposób 2 (zapis „zaprzeczamy” i dodajemy „1”):

$$\begin{array}{r} 1111\ 1111\ 1010\ 1110 \\ \hline 0000\ 0000\ 0101\ 0001 \\ +\qquad\qquad\qquad\quad 1 \\ \hline 0000\ 0000\ 0101\ 0010 \end{array}$$

, co daje nam 82, ale że pierwszy bit był równy 1, zatem kodowaną liczbą jest -82.

■ Sposób 3 (odejmujemy „1” od zapisu i „zaprzeczamy”):

$$\begin{array}{r} 1111\ 1111\ 1010\ 1110 \\ -\qquad\qquad\qquad\quad 1 \\ \hline 1111\ 1111\ 1010\ 1101 \\ \hline 0000\ 0000\ 0101\ 0010 \end{array}$$

, co daje nam 82, ale że pierwszy bit był równy 1, zatem kodowaną liczbą jest -82.

ZADANIE 1.22. Korzystając z opisanych wyżej trzech różnych sposobów, zapisz w `int` następujące liczby:

a) 131 oraz -131;
b) 76 oraz -76;
c) 32100 oraz -32100.

ZADANIE 1.23. Korzystając z opisanych wyżej trzech różnych sposobów, zapisz w systemie dziesiętnym następujące liczby zapisane w `int`:

a) 0000 0000 1111 0011 oraz 1111 1111 0000 1100;
b) 0000 0000 0110 0110 oraz 1111 1111 1001 1001;
c) 0010 1100 0000 1001 oraz 1010 1100 0000 1001.

ZADANIE 1.24. Dla par liczb 3 oraz -3, 3 oraz -2, -2 oraz 3, -2 oraz -2 zapisanych w systemie U2 na 5 bitach wykonaj operacje dodawania, odejmowania oraz mnożenia, a następnie sprawdź poprawność wyników wykonując odpowiednie operacje oraz zamiany w systemie dziesiętnym.

- Jeżeli wynik dodawania/mnożenia ma w zapisie więcej niż 5 bitów, to przycinamy go do 5 najmniej znaczących bitów (czyli od prawej do lewej).
- Należy założyć, że zawsze możliwe jest odejmowanie, tzn. zawsze można pożyczyć bit, dla przykładu:

$$\begin{array}{r} 1112 \\ \hline 00001 \\ -\ 10011 \\ \hline 01110 \end{array}$$

KODY GRAYA

Zakładając, że dostępne są jedynie operacje na bitach, a dokładnie nie mamy możliwości skorzystania z operacji dodawania, odejmowania, mnożenia i dzielenia, zaimplementuj procedurę, która dla danej dodatniej liczby całkowitej k wygeneruje (jakiś) kod Graya k-bitowych słów kodowych. •••

Kod Graya, zwany również kodem *refleksyjnym*, jest to (uporządkowany) kod binarny (bezwagowy i niepozycyjny), w którym dwa kolejne słowa kodowe różnią się od siebie tylko stanem jednego bitu, a ponadto także ostatni i pierwszy wyraz tego kodu spełniają tę własność (a zatem kod Graya jest także kodem *cyklicznym*).

Algorytm rozszerzania kodu Graya

Załóżmy, że ciąg C jest kodem Graya, w którym słowa kodowe są długości n. Aby otrzymać kod Graya o słowach długości $n+1$, rozszerzamy kod C o jeden bit w nastepujący sposób:

1. Dopisz do ciągu C jego odbicie lustrzane C^R (czyli te same słowa kodowe, ale w odwrotnej kolejności).
2. W tak otrzymanym ciągu, do wyrazów z ciągu C dopisz na początku bit o wartości zero, natomiast do tych z ciągu C^R — bit o wartości 1.

Należy podkreślić, że w/w metoda tworzy tylko jeden z możliwych kodów Graya. Równie dobrze można dopisywać 0/1 na innej (ustalonej) pozycji. Ponadto zastosowanie tej samej permutacji na każdym ze słów kodowych dla kodu wygenerowanego w w/w sposób także da w wyniku kod Graya.

W oparciu o algorytm rozszerzania kodu Graya, wygeneruj (jakiś) kod Graya o słowach kodowych długości trzy. **◀ PRZYKŁAD**

Zaczynamy od przykładowego (jednego z dwóch) kodu Graya $C_1 = (0, 1)$ dla 1-bitowych słów kodowych. Przebieg algorytmu rozszerzania kodu C_1 wygląda następująco.

C_1	C_1 oraz jego odbicie lustrzane C_1^R	dopisanie 0 oraz 1
0	0	**0**0
1	1	**0**1
	1	**1**1
	0	**1**0

A zatem mamy (przykładowy) kod Graya dla 2-bitowych słów kodowych: $C_2 = (00, 01, 11, 10)$. Następnie rozszerzamy kod C_2.

C_2	C_2 oraz jego odbicie lustrzane C_2^R	dopisanie 0 oraz 1
00	00	**0**00
01	01	**0**01
11	11	**0**11
10	10	**0**10
	10	**1**10
	11	**1**11
	01	**1**01
	00	**1**00

Otrzymany (przykładowy) kod Graya dla słów 3-bitowych: $C_3 = (000, 001, 011, 010, 110, 111, 101, 100)$.

ZADANIE 1.25. W oparciu o algorytm rozszerzania kodu Graya, wygeneruj (jakiś) kod Graya o słowach kodowych długości cztery.

(Przykładowy) kod Graya n-bitowych słów kodowych można też wygenerować bezpośrednio z naturalnego kodu binarnego, tj. z kolejnych n-bitowych zapisów liczb $0, 1, 2, \ldots, 2^n - 1$.

Algorytm wyznaczania i-tego wyrazu n-bitowego kodu Graya

1. Zapisz pomniejszony o jeden numer wyrazu kodu Graya w naturalnym kodzie dwójkowym na zadanej liczbie bitów (brakujące bity uzupełnij bitem 0).
2. Przesuń kopię ciągu z kroku 1 o jeden bit w prawo (najmniej znaczący bit odrzuć, a na początku dopisz bit o wartości 0, czyli podziel całkowicie przez 2).
3. Wykonaj `XOR` na odpowiednich bitach liczb z kroków 1 oraz 2; wynik jest wyrazem w kodzie Graya.

W oparciu o algorytm wyznaczania i-tego wyrazu n-bitowego kodu Graya wyznacz 10-ty wyraz 4-bitowego (przykładowego) kodu Graya. **◀ PRZYKŁAD**

Interesuje nas 10-ty wyraz. Mamy $(9)_{10} = (1001)_2$, a przesunięcie tego ciągu binarnego w prawo o jeden bit daje w wyniku ciąg $(0100)_2$.

$$\begin{array}{rr} & 1001 \\ \texttt{XOR} & 0100 \\ \hline & 1101 \end{array}$$

A zatem 10-ty wyraz ciągu Graya to 1101.

ZADANIE 1.26. W oparciu o algorytm wyznaczania i-tego wyrazu n-bitowego kodu Graya wyznacz kod Graya o słowach kodowych długości cztery.

ZADANIE 1.27. W oparciu o algorytm wyznaczania i-tego wyrazu n-bitowego kodu Graya wyznacz $3, 5, 9, \ldots, (2^{n-1} + 1)$-ty wyraz n-bitowego kodu Graya.

ZADANIE 1.28.

a) Utwórz wszystkie[3] możliwe 3-bitowe kody Graya.

b)* Ile różnych kodów Graya da się utworzyć w n bitowym kodzie?

Algorytm konwersji wyrazów kodu Graya na wyrazy w naturalnym kodzie binarnym

1. Przyjmij pierwszą (najbardziej znaczącą) cyfrę kodu naturalnego równą pierwszej cyfrze kodu Graya.
2. Każdą kolejną cyfrę oblicz jako `XOR` odpowiedniej cyfry kodu Graya i poprzednio wyznaczonej cyfry kodu naturalnego.

W oparciu o algorytm konwersji wyrazów kodu Graya na wyrazy w naturalnym kodzie binarnym dokonaj konwersji wyrazu 1110 (4-bitowego kodu Graya). **◀ PRZYKŁAD**

Zgodnie z algorytmem w kroku pierwszym otrzymujemy:

$$\begin{array}{r} \mathbf{1}110 \\ \hline \mathbf{1}??? \end{array}$$

Natomiast etapy ustalania kolejnych bitów w kroku drugim przedstawiają się następująco:

[3] Z dokładnością do operacji cyklicznego przesunięcia, tzn., dla przykładu, 2-bitowe kody Graya (00,01,11,10) oraz (01,11,10,00) uznajemy za takie same, a także z dokładnością do operacji odbicia lustrzanego, tzn., dla przykładu, 2-bitowe kody Graya (00,01,11,10) oraz (10,11,01,00) uznajemy także za takie same.

	1	**1**	1	0		1	**1**	1	0		1	1	**1**	0
XOR		**1**			XOR		**0**			XOR			**1**	
	1	**0**	?	?		1	**0**	1	?		1	0	**1**	1

A zatem wyrazem naturalnego 4-bitowego kodu binarnego, który posłużył do utworzenia wyrazu 1110 kodu Graya, jest 1011.

ZADANIE 1.29. W oparciu o algorytm konwersji wyrazów kodu Graya na wyrazy w naturalnym kodzie binarnym dokonaj konwersji wyrazów 00001, 00011, 000111, 01111 oraz 11111 (5-bitowego kodu Graya).

WAGA

Rozważmy wagę szalkową, na której lewej szalce kładziemy jakiś przedmiot do zważenia, a następnie na obu szalkach kładziemy odważniki. Jeżeli waga jest w równowadze, wówczas ważony przedmiot ma wagę równą sumie wag odważników położonych na prawej szalce minus suma wag odważników położonych na lewej szalce obok ważonego przedmiotu. Zakładamy, że zarówno odważniki jak i sam ważony przedmiot posiadają wagi będące liczbami naturalnymi.

ZADANIE 1.30. Mając do dyspozycji po dwa odważniki każdego rodzaju z $1, 3, 9, 27$ wyznaczyć ułożenie odważników na szalkach tak, aby odważyć ciężar 35.

Wskazówka. Rozważyć zapis liczby 35 w systemie o podstawie trzy.

◀ PRZYKŁAD

Mając do dyspozycji po jednym odważniku każdego rodziaju z $1, 3, 9, 27$ wyznaczyć ułożenie odważników na szalkach tak, aby odważyć ciężar 35.

W ogólności, rozłożenie k odważników przy odważaniu ciężaru W odpowiada przedstawieniu W w postaci $W = \sum_{i=0}^{k-1} d_i \cdot 3^i$, gdzie $d_i \in \{-1, 0, 1\}$. Aby przedstawić ciężar W w tej postaci, należy najpierw przedstawić liczbę $W^* = W + \frac{3^k-1}{2}$ w systemie trójkowym: $W^* = (e_{k-1} \dots e_0)_3$, a następnie za d_i podstawić $e_i - 1$. Zatem w rozważanym przykładzie, $W^* = 35 + \frac{3^4-1}{2} = 35 + 40 = 75 = 2 \cdot 27 + 2 \cdot 9 + 1 \cdot 3 + 0 \cdot 1 = (2210)_3$, stąd $d_0 = -1, d_1 = 0, d_2 = 1, d_3 = 1$. Zatem rozłożenie jest następujące: odważnik o nominale 1 na lewej szalce, odważnik o nominale 3 pozostaje na stole, a odważniki o nominałach 9 i 27 na prawej szalce $(35 + 1 = 27 + 9)$.

ZADANIE 1.31. Jak ułożyć na szalkach odważniki o nominałach $1, 3, 9, 27, 81$, aby odważyć ciężar:

a) 92;
b) 111?

Analogiczne rozumowanie jak w powyższym przykładzie można zastosować np. dla odważników innego rodzaju będącego potęgą jakiejś liczby p. Wówczas potrzebujemy odważników nie po jednym z każdego rodzaju, lecz po większej liczbie: wynika to z zapisu w systemie o żądanej podstawie. Jeśli np. rozważymy system odważników o nominałach czterech kolejnych potęg $p = 5$, tzn. $1, 5, 25, 125$, wówczas kolejne cyfry w zapisie liczby $W^* = W + \frac{5^k-1}{2}$ w systemie o podstawie 4 należą do zbioru $\{0, \dots, 4\}$. Aby otrzymać żądany rozkład odważników na szalce, podstawiamy $d_i = e_i - \lfloor \frac{p}{2} \rfloor = e_i - 2$. Jako że $d_i \in \{-2, -1, 0, 1, 2\}$, potrzebujemy po dwa odważniki z każdego rodzaju.

ZADANIE 1.32. Mając do dyspozycji po dwa odważniki każdego rodzaju z $1, 5, 25, 125$ wyznaczyć ułożenie odważników na szalkach tak, aby odważyć ciężar 164.

PRZESZUKIWANIA BINARNE

Załóżmy, że mamy „czarne pudełko", które przechowuje równianie prostej $y = f(x)$ (której nie znamy), zwanej dalej *murem*. Jedyny możliwy sposób korzystania z pudełka to zapytanie o wartość $f(x)$ dla podanego na wejściu argumentu x.

Rozważmy punkt $P = P(t)$, zwany dalej *żabą*, który porusza się, tj. *skacze*, w czasie o wektor $\mathbf{v} = [2, 1]$, startując z punktu $P(0) = (0, 5)$ (czyli $P(0) = (0, 5), P(1) = (2, 6), P(2) = (4, 7), \ldots, P(i + 1) = P(i) + \mathbf{v}, \ldots$). Zaimplementuj efektywny algorytm wyznaczający największą liczbę skoków, które jest w stanie wykonać żaba, aby nie uderzyć w mur. •••

Rozważmy zbiór $A = \{x_0, x_1, x_2, x_3, \ldots, x_{15}\}$. Załóżmy, że w grze przeciwnik wybiera element x z A, a my musimy za pomocą jak najmniejszej liczby pytań odgadnąć ten element. Wówczas sposób postępowania może być następujący:

> Dzielimy zbiór A na dwa rozłączne podzbiory $A_1 = \{x_0, x_1, \ldots, x_7\}$ i $A_2 = \{x_8, x_9 \ldots, x_{15}\}$ i pytamy przeciwnika, do którego podzbioru należy wybrany przez niego element — niech B będzie tym podzbiorem. Następnie w podobny sposób dzielimy zbiór B na „połowy" i powtarzamy pytanie, itd.

Dla przykładu, niech $A = \{0, 1, 2, \ldots, 15\}$ i niech $x = 10$.

$\{0,1,2,3,4,5,6,7\}$	$\{8,9,10,11,12,13,14,15\}$
Nie	Tak
$\{8,9,10,11\}$	$\{12,13,14,15\}$
Tak	Nie
$\{8,9\}$	$\{10,11\}$
Nie	Tak
$\{10\}$	$\{11\}$
Tak	Nie

,

czyli ostatecznie $x = 10$. Zadaliśmy 4 pytania.

Powyższy sposób rozumowania można rozszerzyć na dowolny n-elementowy zbiór A, przy czym w najgorszym przypadku minimalna liczba pytań, jaką należy zadać, to $\lceil \log_2 n \rceil$. Zatem np. mając do dyspozycji k pytań można odgadnąć całkowitą liczbę z przedziału od 0 do $2^k - 1$ (czyli element ze zbioru o mocy 2^k).

W szczególności metodę przeszukiwań binarnych można zastosować do stwierdzenia, czy jakaś liczba naturalna n jest kwadratem innej liczby naturalnej, tzn. czy istnieje naturalna liczba k taka, że $k^2 = n$.

Algorytm(`int` n)

1. $k_d := 1$; $k_g := n$.
2. Powtarzaj aż do skutku:
   - 2.a. Jeżeli $k_g - k_d \leq 1$, to Koniec: n nie ma pierwiastka.
   - 2.b. $j := \lfloor \frac{k_g + k_d}{2} \rfloor$;
   - 2.c. Jeżeli $j^2 = n$, to Koniec: n jest kwadratem j;
   - 2.d. Jeżeli $j^2 > n$, to $k_g := j$, w przeciwnym wypadku $k_d := j$.

Dla podanych niżej liczb wyznacz pierwiastki stopnia drugiego. **◀ PRZYKŁAD**

a) 49

k_d	k_g	$?(k_g - k_d \leq 1)$	j	$?(j^2 = n)$	$?(>, <)$
1	49	$?(49 - 1 \leq 1)$	25	$?(25^2 = 49)$	$>$
1	25	$?(25 - 1 \leq 1)$	13	$?(13^2 = 49)$	$>$
1	13	$?(13 - 1 \leq 1)$	7	$?(7^2 = 49)$	Koniec

,

czyli ostatecznie istnieje $k = 7$ takie, że $k^2 = 49$.

b) 59

k_d	k_g	$?(k_g - k_d \leq 1)$	j	$?(j^2 = n)$	$?(>, <)$
1	59	$?(59 - 1 \leq 1)$	30	$?(30^2 = 59)$	$>$
1	30	$?(30 - 1 \leq 1)$	15	$?(15^2 = 59)$	$>$
1	15	$?(15 - 1 \leq 1)$	8	$?(8^2 = 59)$	$>$
1	8	$?(8 - 1 \leq 1)$	4	$?(4^2 = 59)$	$<$
4	8	$?(8 - 4 \leq 1)$	6	$?(6^2 = 59)$	$<$
6	8	$?(8 - 6 \leq 1)$	7	$?(7^2 = 59)$	$<$
7	8	$?(8 - 7 \leq 1)$	KONIEC		

,

czyli ostatecznie nie istnieje k takie, że $k^2 = 59$. Jednakże z warunków zatrzymania algorytmu wynika, że otrzymaliśmy przybliżenie: $\sqrt{59} \in (7, 8)$.

ZADANIE 1.33. Zastosuj algorytm wyznaczania pierwiastków dla znalezienia pierwiastka stopnia drugiego z następujących liczb:

a) 144;
b) 123;
c) 625;
d) 517.

Odpowiedzi do zadań

1.1.

a) 11.
b) 61.
c) 94.

1.2.

a) $111 = (1101111)_2$.
b) $169 = (10101001)_2$.
c) $411 = (110011011)_2$.

1.3.

a) 111111.
b) 10100.
c) 1000000.

1.4.

a) $1111 < 10001$.
b) $11010 > 10111$.
c) $1111001 < 1111011$.

1.5.

a) $10011 + 1100 = 11111$.
b) $1111 + 1110 = 11101$.
c) $110111 + 110011 = 1101010$.
d) $101 + 111 + 111 = 10011$.
e) $1011 + 1011 + 111 = 11101$.

1.6.

a) $110111 - 110011 = 100$.
b) $10011 - 1100 = 111$.
c) $1010001 - 101110 = 100011$.
d) $1011100 - 1010111 = 101$.

1.7.

a) $10011 \cdot 1100 = 11100100$.
b) $101 \cdot 111 = 100011$.
c) $1111 \cdot 111 = 1101001$.
d) $111000 \cdot 111 = 110001000$.

1.8.

a) $11000 : 1000 = 11$.
b) $100011 : 101 = 111$.
c) $1010001 : 1001 = 1001$.
d) $110001 : 111 = 111$.
e) $1001101 : 111 = 1011$.

1.9.

a) \$A91 = 2705.
b) \$C2 = 194.
c) \$FCA = 4042.

1.10.

a) 199 = \$C7.
b) 541 = \$21D.
c) 855 = \$357.

1.11.

a) $(100010)_2 = (42)_8 = \$22$.
b) $(1011101)_2 = (135)_8 = \$5D$.
c) $(111110110)_2 = (766)_8 = \$1F6$.

1.12.

a) $\$C2 = (11000010)_2 = (302)_8$.
b) $\$A91 = (101010010001)_2 = (5221)_8$.
c) $\$FCA = (100011001010)_2 = (7712)_8$.

1.13.

a) $(713)_8 = (111001011)_2$ i $(713)_8 = 459$.
b) $(1027)_8 = (1000010111)_2$ i $(1027)_8 = 535$.
c) $(37700)_8 = (11111111000000)_2$ i $(37700)_8 = 16320$.

1.14. $(132)_8$ i $(240)_8$.

1.15.

a) 5 cyfr.
b) Jeśli liczba $x \in \{(1000000)_4, (1000001)_4, \ldots, (1333333)_4\}$, to w zapisie dwójkowym ma ona 13 cyfr, w przeciwnym wypadku, jeśli liczba $x \in \{(2000000)_4, \ldots, (3333333)_4\}$, to w zapisie dwójkowym ma ona 14 cyfr.

1.16. Jeśli liczba $x \in \{(10000)_8, (10001)_8, \ldots, (37777)_8]$, to w zapisie czwórkowym ma ona 7 cyfr, w przeciwnym wypadku, jeśli liczba $x \in \{(40000)_8, \ldots, (77777)_8\}$, to w zapisie czwórkowym ma ona 8 cyfr.

1.17.

a) $(0.1101)_2 = 0.8125$.
b) $(0.0011)_2 = 0.1875$.
c) $(0.01101)_2 = 0.40625$.

1.18.

a) $0.5625 = (0.1001)_2$.
b) $0.78125 = (0.11001)_2$.
c) $0.15625 = (0.00101)_2$.
d) $0.328125 = (0.010101)_2$.
e) $7.5625 = (111.1001)_2$.
f) $11.15625 = (1011.00101)_2$.
g) $13.328125 = (1101.010101)_2$.

1.19.

a) 112.
b) 1220.
c) 1012
d) 223.
e) 1363.
f) 111.
g) 130563.

1.20. 19 oraz 127.

1.21. $(2222)_3$ i $(143)_7$; $(11110)_3$ i $(231)_7$.

1.22.

a) 0000 0000 1000 0011, 1111 1111 0111 1101.
b) 0000 0000 0100 1111, 1111 1111 1011 0001.
c) 0111 1101 0110 0100, 1000 0010 1001 1100.

1.23.

a) 243, -244,
b) 102, -103,
c) 11273, -21495.

1.24. $(3)_{10} = (00011)_{\mathrm{U2}}, (-3)_{10} = (11101)_{\mathrm{U2}}, (-2)_{10} = (11110)_{\mathrm{U2}}$

a) $3 + (-3) = 0$, a $00011 + 11101 = 00000$, co daje $(00000)_{\mathrm{U2}} = (0)_{10}$;
$3 - (-3) = 6$, a $00011 - 11101 = 00110$, co daje $(00110)_{\mathrm{U2}} = (6)_{10}$;
$3 \cdot (-3) = -9$, a $00011 \cdot 11101 = 10111$, co daje $(10111)_{\mathrm{U2}} = (-9)_{10}$;

b) $3 + (-2) = 1$, a $00011 + 11110 = 00001$, co daje $(00001)_{\mathrm{U2}} = (1)_{10}$;
$3 - (-2) = 5$, a $00011 - 11110 = 00101$, co daje $(00101)_{\mathrm{U2}} = (5)_{10}$;
$3 \cdot (-2) = -6$, a $00011 \cdot 11110 = 11010$, co daje $(11010)_{\mathrm{U2}} = (-6)_{10}$;

c) $(-3) + 2 = 1$, a $11101 + 00010 = 11111$, co daje $(11111)_{\mathrm{U2}} = (-1)_{10}$;
$(-3) - 2 = -5$, a $11101 - 00010 = 11011$, co daje $(11011)_{\mathrm{U2}} = (-5)_{10}$;
$(-3) \cdot 2 = -6$, a $11101 \cdot 00010 = 11010$, co daje $(11010)_{\mathrm{U2}} = (-6)_{10}$;

d) $(-2) + (-2) = -4$, a $11110 + 11110 = 11100$, co daje $(11100)_{\mathrm{U2}} = (-4)_{10}$;
$(-2) - (-2) = 0$, a $00010 - 00010 = 00000$, co daje $(00000)_{\mathrm{U2}} = (0)_{10}$;
$(-2) \cdot (-2) = 4$, a $11110 \cdot 11110 = 00100$, co daje $(00100)_{\mathrm{U2}} = (4)_{10}$.

1.25. 0000, 0001, 0011, 0010, 0110, 0111, 0101, 0100, 1100, 1101, 1111, 1110, 1010, 1011, 1001, 1000.

1.26. 0000, 0001, 0011, 0010, 0110, 0111, 0101, 0100, 1100, 1101, 1111, 1110, 1010, 1011, 1001, 1000.

1.27. 00000…000011, 00000…000110, 00000…001100, …, 00110…000000, 01100…000000, 11000…000000.

1.28.

a) 6;
b) A066037 (zobacz https://oeis.org/A066037).

1.29. 00001, 00010, 00101, 01010, 10101.

1.30.

Lewa szalka — 0, prawa szalka — 2×27+1×9+2×1.

1.31.

a) Lewa szalka — 1, prawa szalka — 3+9+81.
b) Lewa szalka — 0, prawa szalka — 3+27+81.

1.32. Lewa szalka — 1×1+2×5, prawa szalka — 1×125+2×25.

1.33.

a) $k = 12$;
b) $k_d = 11$, $k_g = 12$, $\sqrt{123} \in (11, 12)$;
c) $k = 25$;
d) $k_d = 22$, $k_g = 23$, $\sqrt{517} \in (22, 23)$.