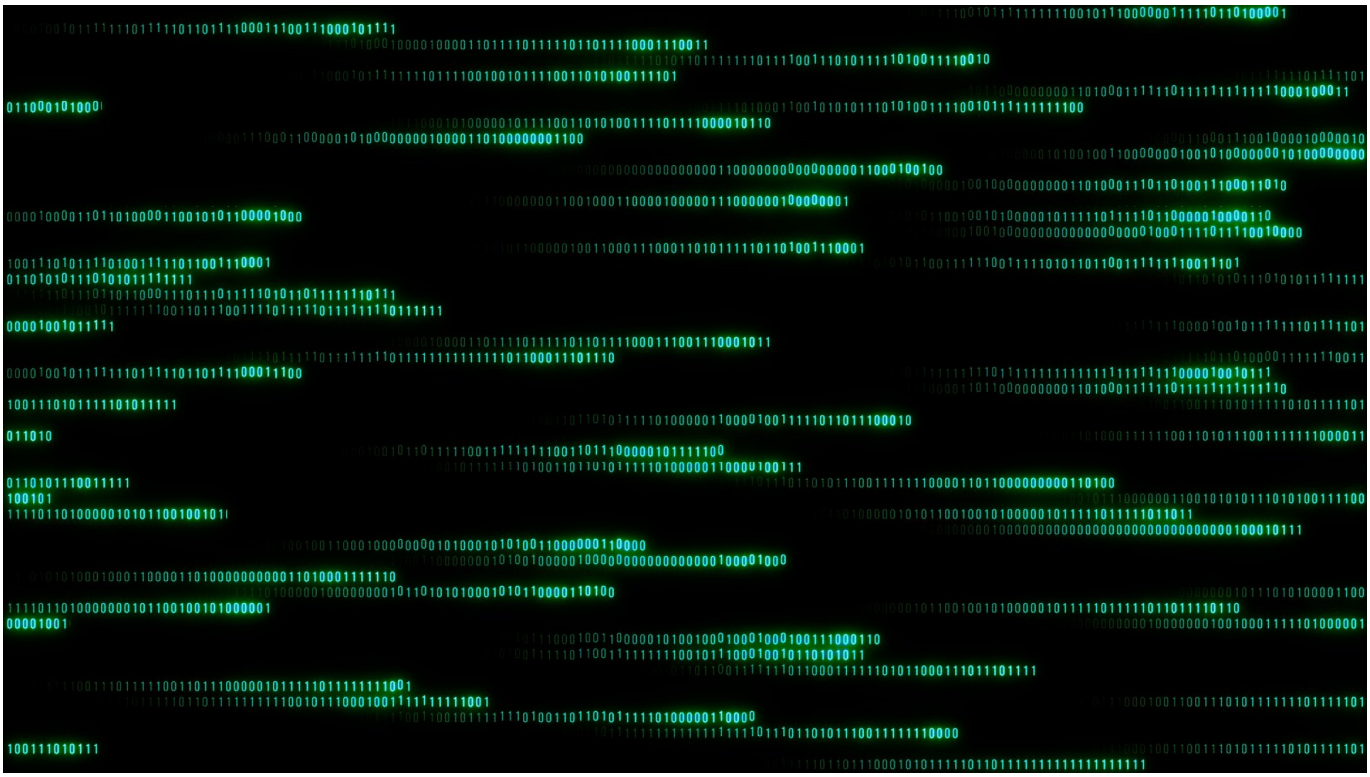


Konwersja liczb z systemu szesnastkowego na binarny w języku C++

Źródło: Pixabay, domena publiczna.

> W tym e-materiale powtarzamy wiadomości ze szkoły podstawowej.

W e-materiale Konwersja liczb z systemu szesnastkowego na binarny poznaliśmy dwa sposoby konwersji liczb zapisanych w postaci szesnastkowej do ich odpowiedników w systemie binarnym.

W tym e-materiale napiszemy program w języku C++ realizujący te algorytmy.

Ciekawi cię, jak wyglądają implementacje w innych językach programowania? Możesz się z nimi zapoznać w pozostałych e-materiałach z tej serii:

- Konwersja liczb z systemu szestnastkowego na binarny w języku Java,
- Konwersja liczb z systemu szestnastkowego na binarny w języku Python.

Więcej zadań? Sięgnij do Konwersja liczb z systemu szestnastkowego na binarny – zadania maturalne.

> **Twoje cele**
>
> - Zastosujesz w praktyce wiedzę na temat pozycyjnych systemów liczbowych.
> - Zaimplementujesz w języku C++ algorytm konwersji liczby zapisanej w systemie szesnastkowym do systemu dwójkowego oraz dziesiętnego.

- Przeanalizujesz kod programu zapisującego część ułamkową liczby szesnastkowej w postaci binarnej.

Przeczytaj

Realizacja algorytmu w języku C++

Do napisania w języku C++ programu przekształcającego liczbę szesnastkową do postaci binarnej wykorzystamy dwuetapowy algorytm omówiony w e-materiale Konwersja liczb z systemu szesnastkowego na binarny.

Zacznijmy od zdefiniowania funkcji `hex2Dec()`, która dokona konwersji liczby szesnastkowej do postaci dziesiętnej:

```
int hex2Dec(string liczba_hex) {
    int wynik = 0;
    int cyfra_hex = 0;
    int waga = 1;

    for (int i = liczba_hex.length() - 1; i >= 0; i--) {
        cyfra_hex = liczba_hex[i];

        if (liczba_hex[i] >= '0' && liczba_hex[i] <= '9') {
            cyfra_hex = liczba_hex[i] - '0';
        } else if (liczba_hex[i] >= 'A' && liczba_hex[i] <= 'F')
            cyfra_hex = liczba_hex[i] - 55;
        }

        wynik = wynik + cyfra_hex * waga;
        waga = waga * 16;
    }

    return wynik;
}
```

W programie deklarujemy trzy zmienne typu całkowitego:

- `wynik` – suma obliczonych iloczynów częściowych,
- `cyfra_hex` – zmienna przechowująca kolejne cyfry przekształcanej liczby,
- `waga` – kolejne potęgi liczby 16, odpowiadające wagom poszczególnych cyfr liczby szesnastkowej.

Zmienna `waga` na początku ma wartość 1, ponieważ najmłodszemu symbolowi odpowiada waga 16^0 (czyli 1).

W pętli `for` kolejne cyfry szesnastkowe, zaczynając od najmniej znaczącej, będą mnożone przez wagi odpowiadające ich pozycji. Parametrem funkcji jest zmienna typu `string`, czyli tekst zawierający liczbę szesnastkową.

Trzeba zatem znaleźć sposób przekształcenia kolejnych cyfr liczby szesnastkowej na odpowiadające im wartości liczbowe w systemie szesnastkowym.

Skorzystamy z instrukcji warunkowej `if`, aby sprawdzić, czy odczytywana cyfra liczby szesnastkowej należy do zbioru cyfr (od 0 do 9), czy też liter (od A do F). Najpierw zapiszmy poniższy warunek.

```
if (liczba_hex[i] >= '0' && liczba_hex[i] <= '9') {
    cyfra_hex = liczba_hex[i] - '0';
```

Sprawdza on, czy zmienna `liczba_hex[i]` zawiera znaki ASCII odpowiadające cyfrom ('0', '1', '2', ... '9'). Jeśli tak, to w zmiennej `cyfra_hex` (typu `int`) zapisujemy liczbę odpowiadającą kodowi ASCII znaku cyfry, odejmując od niej kod ASCII znaku zera (wynosi on 48).
W rezultacie zmienna `cyfra_hex` przyjmie wartość od 0 do 9.

> **Przykład 1**
>
> Jeżeli `liczba_hex[i]` będzie znakiem „6”, zostanie wykonane działanie $54 - 48$, czyli zmienna `cyfra_hex` przyjmie wartość 6; gdy `liczba_hex[i]` będzie znakiem „0”, zostanie obliczona różnica $48 - 48$, więc `cyfra_hex` przyjmie wartość 0.

Rozpatrzmy teraz drugi przypadek, kiedy sprawdzany znak nie będzie cyfrą, lecz literą (od A do F). Wtedy od zmiennej `liczba_hex[i]` trzeba odjąć liczbę 55. Dlaczego właśnie tyle? Otóż symbole literowe ze zbioru $\langle \mathrm{A}, \ldots, \mathrm{F} \rangle$ mają przyporządkowane w kodzie ASCII wartości od 65 do 70. Po odjęciu od nich liczby 55 otrzymamy dziesiętne odpowiedniki cyfr szesnastkowych.

> **Przykład 2**
>
> Znakowi „C” w kodzie ASCII odpowiada liczba 67. Jeżeli odejmiemy od niej 55, uzyskamy wynik 12, czyli dziesiętną wartość $\mathrm{C}_{(16)}$.

Uzupełniamy zatem instrukcję `if`:

```
else if (liczba_hex[i] >= 'A' && liczba_hex[i] <= 'F') {
    cyfra_hex = liczba_hex[i] - 55;
}
```

Następnym etapem jest zdefiniowanie funkcji `dec2Bin()`, która posłuży do przekształcenia liczby dziesiętnej do systemu binarnego. Funkcja ta zwraca łańcuch znaków (`string`) – reprezentację binarną liczby:

```
string dec2Bin(int liczba_dec) {
    string liczba_bin = "";
    int reszta = 0;

    while (liczba_dec > 0) {
        reszta = liczba_dec % 2;
        liczba_dec = liczba_dec / 2;
        liczba_bin = to_string(reszta) + liczba_bin;
    }

    return liczba_bin;
}
```

Konwersji dokonujemy poprzez konkatenację zmiennej `liczba_bin` i kolejnych reszt z dzielenia liczby dziesiętnej przez 2. Aby móc wykonać konkatenację zmiennej `liczba_bin` typu `string` oraz liczby całkowitej `reszta`, używamy funkcji wbudowanej `to_string()`, która dla liczby `int` zwraca jej tekstową reprezentację (`string`).

Oto pełny kod programu realizującego konwersję liczby szesnastkowej do postaci binarnej:

```
#include <iostream>
#include <string>
using namespace std;

int hex2Dec(string liczba_hex) {
    int wynik = 0;
    int cyfra_hex = 0;
    int waga = 1;

    for (int i = liczba_hex.length() - 1; i >= 0; i--) {
        cyfra_hex = liczba_hex[i];

        if (liczba_hex[i] >= '0' && liczba_hex[i] <= '9') {
            cyfra_hex = liczba_hex[i] - '0';
        } else if (liczba_hex[i] >= 'A' && liczba_hex[i] <= 'F')
            cyfra_hex = liczba_hex[i] - 55;
```

```
        }

        wynik = wynik + cyfra_hex * waga;
        waga = waga * 16;
    }

    return wynik;
}

string dec2Bin(int liczba_dec) {
    string liczba_bin = "";
    int reszta = 0;

    while (liczba_dec > 0) {
        reszta = liczba_dec % 2;
        liczba_dec = liczba_dec / 2;
        liczba_bin = to_string(reszta) + liczba_bin;
    }

    return liczba_bin;
}


int main() {
    string liczba_test = "1A";

    cout << hex2Dec(liczba_test) << endl;
    int liczbaDec = hex2Dec(liczba_test);
    cout << dec2Bin(liczbaDec);

    return 0;
}
```

Pozostałe materiały z serii

- Konwersja liczb z systemu szesnastkowego na binarny
- Konwersja liczb z systemu szesnastkowego na binarny w języku Java
- Konwersja liczb z systemu szesnastkowego na binarny w języku Python
- Konwersja liczb z systemu szesnastkowego na binarny - zadania maturalne

Słownik

argument

konkretna wartość przekazywana do funkcji lub metody, przy jej wywołaniu

parametr

element składni w określonym języku programowania umożliwiający komunikację pomiędzy podprogramem wywołanym a programem wywołującym; parametry określa się wraz z deklaracją określonego podprogramu w jego nagłówku

Film samouczek

Problem 1

Napisz program, który przekształci liczbę zapisaną w systemie szesnastkowym do postaci w systemie binarnym. Działanie programu przetestuj dla liczby 1B.

Specyfikacja problemu:

Dane:

- `liczbaSzesnastkowa` – łańcuch znaków; liczba zapisana w systemie szesnastkowym

Wynik:

Program na wyjściu standardowym wypisuje wartość podanej liczby szesnastkowej w systemie binarnym.

```
#include <iostream>
#include <string>
#include <algorithm>

using namespace std;

// tutaj dopisz funkcje

int main() {
    string liczbaSzesnastkowa = "1B";

    // tutaj dopisz kod
}
```

```
1
```

Polecenie 1

Porównaj swoje rozwiązanie z przedstawionym w filmie.

Film dostępny pod adresem /preview/resource/RVqblTjVISR9z

Film nawiązujący do treści materiału: algorytm zamiany liczby hex na bin.

Plik z kodem źródłowym do pobrania:

Plik o rozmiarze 1.10 KB w języku polskim

Polecenie 2

Przeanalizuj sposób konwersji części ułamkowej liczby szesnastkowej do postaci binarnej; w przypadku tego algorytmu wykorzystuje się fakt, że podstawy obydwu systemów są bazami skojarzonymi.

1

Napiszemy program przekształcający część ułamkową liczby szesnastkowej do postaci binarnej.

2

Zaczniemy od zbudowania szkieletu programu, czyli zadeklarowania funkcji głównej – `main()`:

```
#include <iostream>
using namespace std;

int main() {
    return 0;
}
```

3

Następnie zadeklarujemy funkcję `konwertuj_Ulamek()`, która zwróci wynik działania programu. Będzie nim wartość typu `string` zawierająca ułamek zapisany w systemie binarnym.

Pamiętajmy o dołączeniu biblioteki pozwalającej przetwarzać łańcuchy znaków. Robimy to, wydając polecenie `#include <string>`:

```
#include <iostream>
using namespace std;
#include <string>
```

```
4
5  string konwertuj_Ulamek()
   {
6
7  }
8
9  int main() {
10     return 0;
11 }
```

4

Parametrem przyjmowanym przez funkcję będzie zmienna `liczba_hex`, czyli szesnastkowa liczba (ułamek), którą przekształcimy do postaci dwójkowej. Parametr podamy jako łańcuch znaków (`string`), ponieważ w systemie szesnastkowym oprócz cyfr od 0 do 9 wykorzystuje się sześć symboli literowych (znaki od 'A' do 'F'):

```
1  #include <iostream>
2  using namespace std;
3  #include <string>
4
5  string
   konwertuj_Ulamek(string
   liczba_hex) {
6
7  }
8
9  int main() {
10     return 0;
11 }
```

Zdefiniujemy teraz łańcuch znaków (`string`) o nazwie `wynik`. Jej początkowymi elementami

5

będą "0" oraz ",", ponieważ przekształcamy część ułamkową liczby:

```
#include <iostream>
using namespace std;
#include <string>

string
konwertuj_Ulamek(string
liczba_hex) {
    string wynik = "0,";
}

int main() {
    return 0;
}
```

6

Teraz utworzymy pętlę `for`, dzięki której będziemy przetwarzać kolejne elementy zmiennej `liczba_hex`, zaczynając od cyfry o wadze 2^{-1}. Jest nią pierwszy element po przecinku, czyli `liczba_hex[2]` (znakami zapisanymi na pozycjach `liczba_hex[0]` i `liczba_hex[1]` są – odpowiednio – "0" oraz ","):

```
#include <iostream>
using namespace std;
#include <string>

string
konwertuj_Ulamek(string
liczba_hex) {
    string wynik = "0,";

    for (int i = 2; i <
liczba_hex.length(); i++)
{

    }
```

```
}

int main() {
    return 0;
}
```

7

Zdefiniujemy zmienną o nazwie `bit`, która będzie przechowywać kolejne elementy łańcucha `liczba_hex`:

```
#include <iostream>
using namespace std;
#include <string>

string konwertuj_Ulamek(string liczba_hex) {
    string wynik = "0,";

    for (int i = 2; i < liczba_hex.length(); i++) {
        char bit = liczba_hex[i];
    }
}

int main() {
    return 0;
}
```

8

Jak wiemy, podstawa systemu szesnastkowego (16), jest czwartą potęgą liczby 2, czyli podstawy systemu dwójkowego. Oznacza to, że każdą cyfrę szesnastkową można przedstawić za

pomocą złożonej z czterech bitów liczby dwójkowej.

9

Każdemu elementowi szesnastkowemu, przechowywanemu w zmiennej `bit`, przypiszemy odpowiadający mu ciąg czterech bitów. Wykorzystamy w tym celu instrukcję wielokrotnego wyboru `switch`.

Jako warunek (przełącznik) instrukcji `switch` podamy kolejne wartości zmiennej `bit`.
W rezultacie otrzymamy 4-bitowy ciąg, który dopiszemy do zmiennej `wynik`:

```
#include <iostream>
using namespace std;
#include <string>

string konwertuj_Ulamek(string liczba_hex) {
    string wynik = "0,";

    for (int i = 2; i < liczba_hex.length(); i++) {
        char bit = liczba_hex[i];

        switch(bit) {
            case '0':
                wynik = wynik + "0000";
                break;
            case '1':
                wynik = wynik + "0001";
                break;
            case '2':
                wynik = wynik + "0010";
```

```
                break;
            case '3':
                wynik =
wynik + "0011";
                break;
            case '4':
                wynik =
wynik + "0100";
                break;
            case '5':
                wynik =
wynik + "0101";
                break;
            case '6':
                wynik =
wynik + "0110";
                break;
            case '7':
                wynik =
wynik + "0111";
                break;
            case '8':
                wynik =
wynik + "1000";
                break;
            case '9':
                wynik =
wynik + "1001";
                break;
            case 'A':
                wynik =
wynik + "1010";
                break;
            case 'B':
                wynik =
wynik + "1011";
                break;
            case 'C':
                wynik =
wynik + "1100";
                break;
            case 'D':
                wynik =
wynik + "1101";
                break;
```

```
54              case 'E':
55                  wynik =
    wynik + "1110";
56                  break;
57              case 'F':
58                  wynik =
    wynik + "1111";
59                  break;
60          }
61      }
62
63      return wynik;
64  }
65
66  int main() {
67      return 0;
68  }
```

Ostatnią instrukcją w ciele funkcji `konwertuj_Ulamek()` jest polecenie zwrócenia zmiennej `wynik`, czyli liczby zapisanej w systemie dwójkowym.

10

Zwróć uwagę, jak w instrukcji `switch`

kolejne 4-bitowe bloki były dołączane do zmiennej `wynik`. Wykorzystaliśmy operator dodawania (+), pamiętając o odpowiedniej kolejności argumentów. Bity były dopisywane (dodawane) do zmiennej `wynik`, a nie odwrotnie. Jest to ważne: nie wykonywaliśmy operacji dodawania na liczbach, lecz łączyliśmy ze sobą łańcuchy znaków. Zależało nam na tym, aby kolejne 4-bitowe bloki były dopisywane z prawej strony ciągu `wynik`.

Program jest prawie gotowy. Trzeba jeszcze dopisać kod, dzięki któremu użytkownik poda

11

ułamek do przekształcenia. Skorzystamy ze strumienia `cin`. Musimy także zdefiniować zmienną `liczba_test`, która przechowa wpisaną liczbę:

```
#include <iostream>
using namespace std;
#include <string>

string konwertuj_Ulamek(string liczba_hex) {
    string wynik = "0,";

    for (int i = 2; i < liczba_hex.length(); i++) {
        char bit = liczba_hex[i];

        switch(bit) {
            case '0':
                wynik = wynik + "0000";
                break;
            case '1':
                wynik = wynik + "0001";
                break;
            case '2':
                wynik = wynik + "0010";
                break;
            case '3':
                wynik = wynik + "0011";
                break;
            case '4':
                wynik = wynik + "0100";
                break;
            case '5':
                wynik = wynik + "0101";
                break;
```

```
            case '6':
                wynik = wynik + "0110";
                break;
            case '7':
                wynik = wynik + "0111";
                break;
            case '8':
                wynik = wynik + "1000";
                break;
            case '9':
                wynik = wynik + "1001";
                break;
            case 'A':
                wynik = wynik + "1010";
                break;
            case 'B':
                wynik = wynik + "1011";
                break;
            case 'C':
                wynik = wynik + "1100";
                break;
            case 'D':
                wynik = wynik + "1101";
                break;
            case 'E':
                wynik = wynik + "1110";
                break;
            case 'F':
                wynik = wynik + "1111";
                break;
        }
    }

    return wynik;
}
```

```
65
66 int main() {
67     string liczba_test;
68
69     cout << "Podaj część
   ułamkową liczby w systemie
   szesnastkowym, którą
   chcesz przekonwertować do
   systemu binarnego" <<
   endl;
70     cin >> liczba_test;
71
72     return 0;
73 }
```

12

Trzeba jeszcze wywołać funkcję `konwertuj_Ulamek()` z parametrem`liczba_test` i wyświetlić wynik.

Oto pełny kod programu:

```
1  #include <iostream>
2  using namespace std;
3  #include <string>
4
5  string
   konwertuj_Ulamek(string
   liczba_hex) {
6      string wynik = "0,";
7
8      for (int i = 2; i <
   liczba_hex.length(); i++)
   {
9          char bit =
   liczba_hex[i];
10
11         switch(bit) {
12             case '0':
13                 wynik =
   wynik + "0000";
```

```
                break;
            case '1':
                wynik =
wynik + "0001";
                break;
            case '2':
                wynik =
wynik + "0010";
                break;
            case '3':
                wynik =
wynik + "0011";
                break;
            case '4':
                wynik =
wynik + "0100";
                break;
            case '5':
                wynik =
wynik + "0101";
                break;
            case '6':
                wynik =
wynik + "0110";
                break;
            case '7':
                wynik =
wynik + "0111";
                break;
            case '8':
                wynik =
wynik + "1000";
                break;
            case '9':
                wynik =
wynik + "1001";
                break;
            case 'A':
                wynik =
wynik + "1010";
                break;
            case 'B':
                wynik =
wynik + "1011";
                break;
```

```
            case 'C':
                wynik = wynik + "1100";
                break;
            case 'D':
                wynik = wynik + "1101";
                break;
            case 'E':
                wynik = wynik + "1110";
                break;
            case 'F':
                wynik = wynik + "1111";
                break;
        }
    }

    return wynik;
}

int main() {
    string liczba_test;

    cout << "Podaj część ułamkową liczby w systemie szesnastkowym, którą chcesz przekonwertować do systemu binarnego" << endl;
    cin >> liczba_test;

    cout << konwertuj_Ulamek(liczba_test);

    return 0;
}
```

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Napisz program, który będzie dokonywał konwersji liczby całkowitej z systemu o podstawie 16 do systemu binarnego, z użyciem konwersji liczby szesnastkowej do liczby dziesiętnej i z liczby dziesiętnej do liczby binarnej. Uzupełnij podany kod. Wynik działania programu przetestuj dla liczby 1A.

Specyfikacja problemu:

Dane:

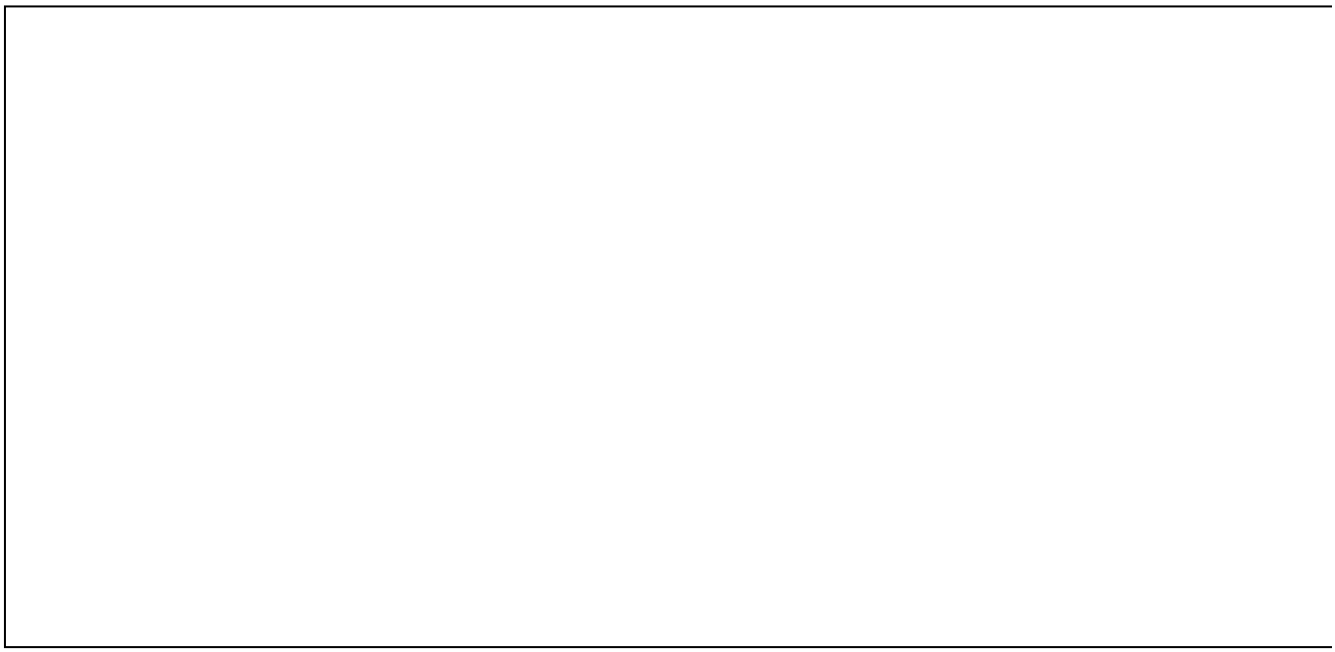
- `podstawa` – liczba naturalna
- `liczba` – łańcuch znaków

Wynik:

- `liczbaDec` – liczba naturalna

Twoje zadania

1. Program powinien wyświetlić liczbę $1A_{(16)}$ w postaci dwójkowej.



Ćwiczenie 2

Uzupełnij program, aby otrzymać algorytm konwersji liczby całkowitej z systemu szesnastkowego do systemu binarnego. Przetestuj działanie programu dla liczby $\mathtt{ABBA}_{(16)}$. Wykorzystaj bazy skojarzone.

Dane:

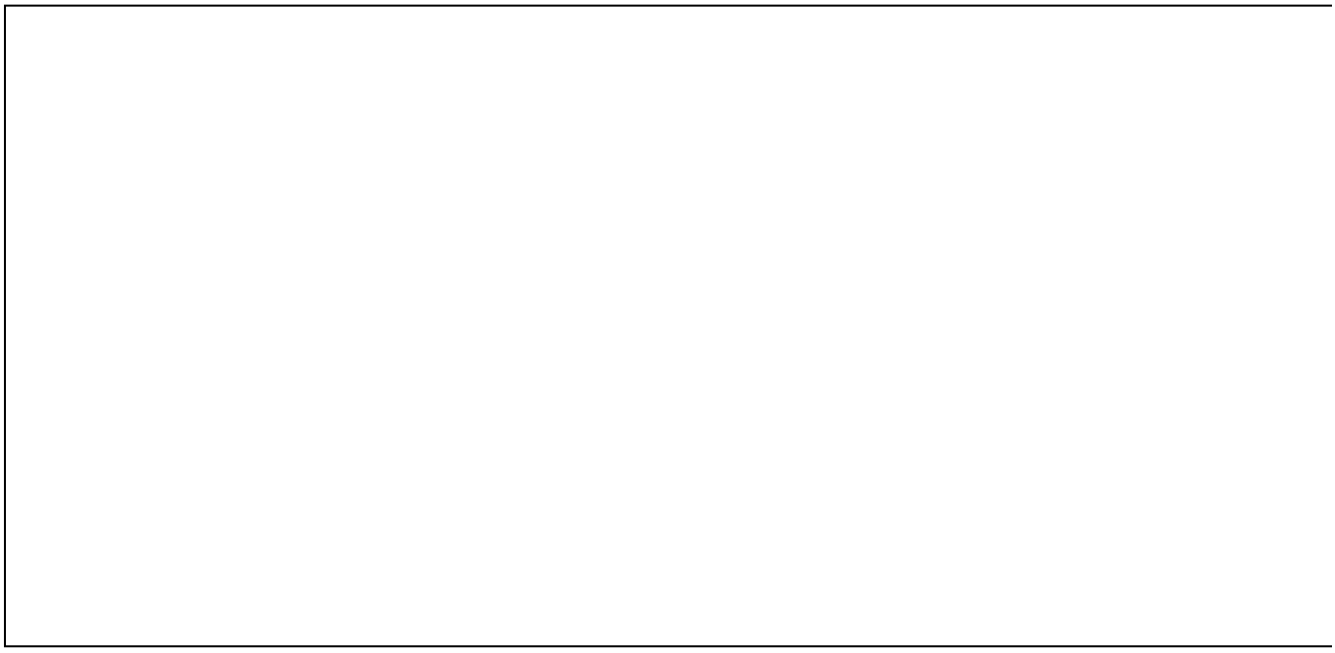
- `liczba` – łańcuch znaków

Wynik:

Program wyświetla liczbę zapisaną w systemie szesnastkowym po konwersji do systemu dwójkowego.

Twoje zadania

1. Program przekształca liczbę zapisaną za pomocą systemu szesnastkowego do postaci dwójkowej.



Ćwiczenie 3

Napisz program, który dokona konwersji ułamka zapisanego w systemie szesnastkowym na system binarny. W implementacji wykorzystaj fakt, że podstawa 16 jest czwartą potęgą liczby 2. Przetestuj działanie programu dla liczby $\texttt{0,6B2}_{(16)}$.

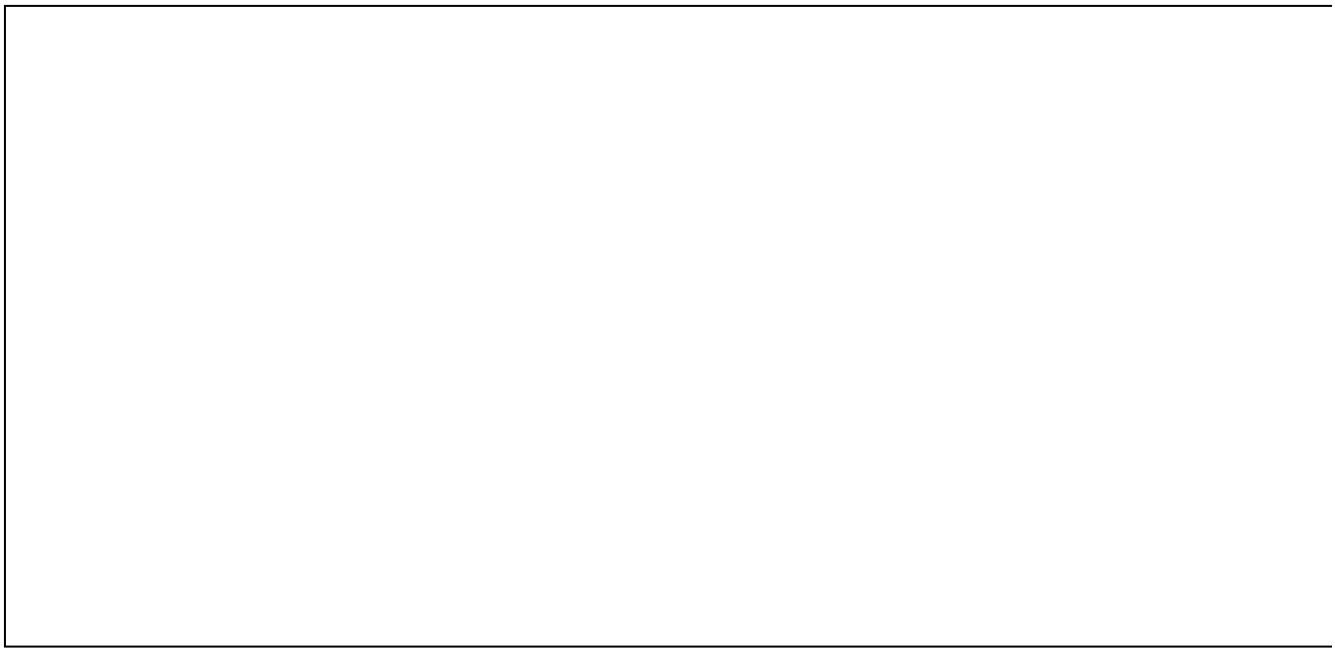
Dane:

Wynik:

- `liczba` – łańcuch znaków
- `wynik` – liczba rzeczywista

Twoje zadania

1. Program ma przekonwertować liczbę zapisaną w systemie szesnastkowym do systemu dwójkowego.



Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Konwersja liczb z systemu szesnastkowego na binarny w języku C++

Grupa docelowa:

Szkoła ponadpodstawowa, liceum ogólnokształcące, technikum, zakres podstawowy i rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

I. Rozumienie, analizowanie i rozwiązywanie problemów na bazie logicznego i abstrakcyjnego myślenia, myślenia algorytmicznego i sposobów reprezentowania informacji.

III. Posługiwanie się komputerem, urządzeniami cyfrowymi i sieciami komputerowymi, w tym: znajomość zasad działania urządzeń cyfrowych i sieci komputerowych oraz wykonywania obliczeń i programów.

Treści nauczania – wymagania szczegółowe

I. Rozumienie, analizowanie i rozwiązywanie problemów.

Zakres podstawowy. Uczeń:

1) planuje kolejne kroki rozwiązywania problemu, z uwzględnieniem podstawowych etapów myślenia komputacyjnego (określenie problemu, definicja modeli i pojęć, znalezienie rozwiązania, zaprogramowanie i testowanie rozwiązania).

2) stosuje przy rozwiązywaniu problemów z różnych dziedzin algorytmy poznane w szkole podstawowej oraz algorytmy:

a) na liczbach: badania pierwszości liczby, zamiany reprezentacji liczb między pozycyjnymi systemami liczbowymi, działań na ułamkach z wykorzystaniem NWD i NWW,

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

3) sprawnie posługuje się zintegrowanym środowiskiem programistycznym przy pisaniu, uruchamianiu i testowaniu programów;

I + II. Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

2) wykorzystuje znane sobie algorytmy przy rozwiązywaniu i programowaniu rozwiązań następujących problemów:

b) wykonywania działań na liczbach w systemach innych niż dziesiętny,

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Zastosujesz w praktyce wiedzę na temat pozycyjnych systemów liczbowych.
- Zaimplementujesz w języku C++ algorytm konwersji liczby zapisanej w systemie szesnastkowym do systemu dwójkowego oraz dziesiętnego.
- Przeanalizujesz kod programu zapisującego część ułamkową liczby szesnastkowej w postaci binarnej.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych.

Formy pracy:

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda;
- oprogramowanie dla języka C++, w tym kompilator GCC/G++ 4.5 (lub nowszej wersji) i Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Konwersja liczb z systemu szesnastkowego na binarny w języku C++". Uczniowie zapoznają się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel wyświetla temat i cele zajęć. Prosi uczniów, by na podstawie wiadomości zdobytych przed lekcją zaproponowali kryteria sukcesu.
2. **Rozpoznanie wiedzy uczniów.** Nauczyciel zadaje uczniom pytania dotyczące ich aktualnego stanu wiedzy w obszarze poruszanego tematu i programowania, np.
   – czym są bazy skojarzone?
   – co oznacza pojęcie „najmniej znacząca cyfra"?
   – jak wygląda dwuetapowy algorytm przekształcania liczby szesnastkowej do postaci binarnej?
   – w jakie sposób dokonujemy konwersji części ułamkowej?
   Chętni uczniowie udzielają na nie odpowiedzi.

Faza realizacyjna:

1. Uczniowie analizują przykład z sekcji „Przeczytaj" i powtarzają zaprezentowane rozwiązanie na swoim komputerze.
2. **Praca z multimedium.** Uczniowie w zespołach dwuosobowych zapoznają się z treścią polecenia nr 1: „Przeanalizuj sposób konwersji części ułamkowej liczby szesnastkowej do postaci binarnej; w przypadku tego algorytmu wykorzystuje się fakt, że podstawy obydwu systemów są bazami skojarzonymi." z sekcji „Film samouczek" i wspólnie analizują kolejne kroki rozwiązania postawionego problemu.
3. **Ćwiczenie umiejętności.** Uczniowie wykonują indywidualnie ćwiczenia nr 1 i 2;. Po wykonaniu każdego z nich następuje omówienie rozwiązania przez nauczyciela.

Faza podsumowująca:

1. Wybrany uczeń podsumowuje zajęcia z programowania w C++, zwracając uwagę na nabyte umiejętności.

Praca domowa:

1. Uczniowie wykonują polecenie nr 2 z sekcji „Film samouczek”.

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka C++.
- Oficjalna dokumentacja techniczna dla kompilatora GCC/G++ 4.5 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio.

Wskazówki metodyczne:

- Treści w sekcji „Film samouczek” można wykorzystać jako materiał służący powtórzeniu materiału.