Bases Numéricas na Computação

Vinícius Silva Madureira Pereira

bit.ly/lp280818

Definição

Uma base numérica é um conjunto de símbolos, derivado de um sistema numérico, para a representação de valores. Para cada base um mesmo valor é expressado com diferentes algarismo.

Conceituação

Na eletrônica digital, os dispositivos trabalham apenas com valores binários em seus níveis mais baixos de abstração.

Um microprocessador, por exemplo, trabalha com apenas dois estados, formalmente representados por 0 e 1, como por exemplo $1010010001010001_{(2)}$.

Representar valores em uma base com menos algarismos é consideravelmente mais fácil para se entender e, consequentemente, trabalhar, sendo o número anterior equivalente a $A451_{(16)}$.

Aplicações

- Identificação das cores no sistema RGB
- Endereços de memória
- Transistor Transistor Logic (TTL)
- Tabelas de codificação de formato (ASCII)

Bases Numéricas

- Binária (base 2): 0, 1
- Ternária (base 3): 0, 1, 2
- Octal (base 8): 0, 1, 2, 3, 4, 5, 6, 7
- Decimal (base 10): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Duodecimal (base 12): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B
- Hexadecimal (base 16): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Conversões entre Bases

Uma das formas de converter um valor da base 10 para a base n, pode utilizar o seguinte algoritmo:

- Divide-se o valor por n, obtém-se o quociente (valor inteiro da divisão) e o resto;
- O quociente da divisão anterior torna-se o dividendo da nova divisão. Repete-se, então, a divisão até que o quociente da mesma seja 0;
- Os algarismos restos de cada uma das divisões são reunidos invertidamente.

Conversões entre Bases

$19_{(10)} \cong 201_{(3)}$

Conversões entre Bases

Uma das formas de converter um valor da base n para a base 10, pode utilizar o seguinte algoritmo:

- Multiplica-se o algarismo mais a direita pela base n elevada à potência x (n^x) – x inicia-se com o valor 0;
- É incrementado 1 ao valor de x. O resultado da multiplicação anterior é somado com o segundo algarismo mais a direita multiplicado por n^x novamente;
- O procedimento é repetido até que o último algarismo seja alcançado.

Conversões entre Bases

Algarismo Base Algarismo Base Algarismo Base

$2 \times 3^{2} + 0 \times 3^{1} + 1 \times 3^{0}$

Potência Potência Potência

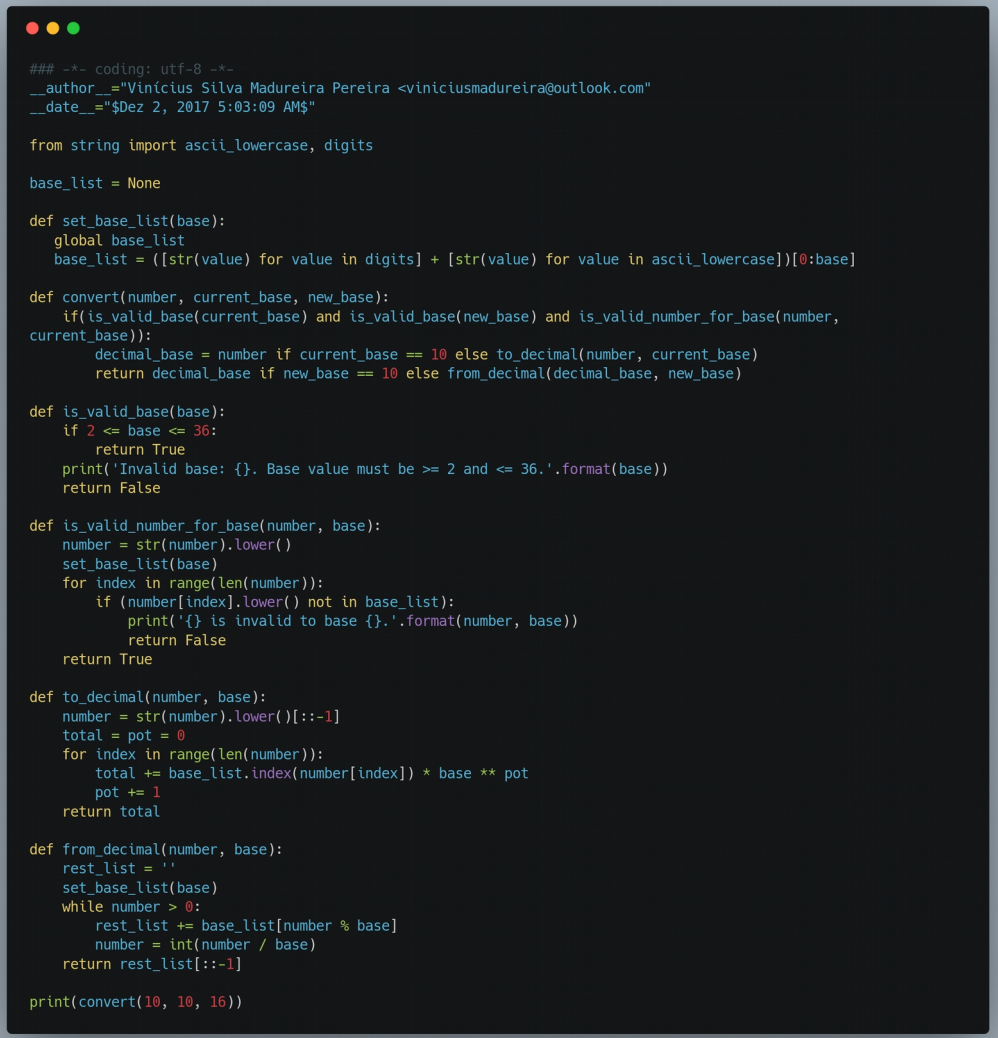
$2 \times 9 + 0 \times 3 + 1 \times 1$

$18 + 0 + 1$

19

$201_{(3)} \cong 19_{(10)}$

Algoritmo de Conversão de Bases



```
### -*- coding: utf-8 -*-
__author__="Vinícius Silva Madureira Pereira <viniciusmadureira@outlook.com"
__date__="$Dez 2, 2017 5:03:09 AM$"

from string import ascii_lowercase, digits

base_list = None

def set_base_list(base):
   global base_list
   base_list = ([str(value) for value in digits] + [str(value) for value in ascii_lowercase])[0:base]

def convert(number, current_base, new_base):
    if(is_valid_base(current_base) and is_valid_base(new_base) and is_valid_number_for_base(number,
current_base)):
        decimal_base = number if current_base == 10 else to_decimal(number, current_base)
        return decimal_base if new_base == 10 else from_decimal(decimal_base, new_base)

def is_valid_base(base):
    if 2 <= base <= 36:
        return True
    print('Invalid base: {}. Base value must be >= 2 and <= 36.'.format(base))
    return False

def is_valid_number_for_base(number, base):
    number = str(number).lower()
    set_base_list(base)
    for index in range(len(number)):
        if (number[index].lower() not in base_list):
            print('{} is invalid to base {}.'.format(number, base))
            return False
    return True

def to_decimal(number, base):
    number = str(number).lower()[::-1]
    total = pot = 0
    for index in range(len(number)):
        total += base_list.index(number[index]) * base ** pot
        pot += 1
    return total

def from_decimal(number, base):
    rest_list = ''
    set_base_list(base)
    while number > 0:
        rest_list += base_list[number % base]
        number = int(number / base)
    return rest_list[::-1]

print(convert(10, 10, 16))
```

Vinícius Silva Madureira Pereira

bit.ly/lp280818

Referências

- CARVALHO, A. C. P. L. F.; LORENA, A. C. Introdução à computação: **hardware, software e dados**. 1 ed. Rio de Janeiro: LTC, 2016. 200 p.

- SIPSER, M. Introdução à teoria da computação. 2. ed. São Paulo: Cengage CTP, 2005. 488 p.

- VIERIA, N. Introdução aos fundamentos da computação: **linguagens e máquinas**. 1 ed. São Paulo: Cengage CTP, 2006. 334 p.

Dúvidas

Obrigado...

Boa noite
para
todos!!!