Bases numéricas e codificação de texto

Sumário

Números binários e hexadecimais

Números decimais

- Estamos acostumados com números decimais, que são números expressos na base 10.
- Qualquer número decimal pode ser expresso pela multiplicação de seus dígitos e potências do valor 10, que é a base do sistema.
- Por exemplo:

$$367 = 3 * 10^2 + 6 * 10^1 + 7 * 10^0$$

Números decimais

- Estamos acostumados com números decimais, que são números expressos na base 10.
- Qualquer número decimal pode ser expresso pela multiplicação de seus dígitos e potências do valor 10, que é a base do sistema.
- Por exemplo:

$$367 = 3 * 10^2 + 6 * 10^1 + 7 * 10^0$$

que é equivalente a

$$367 = 3 * 100 + 6 * 10 + 7$$

que é equivalente a

$$367 = 300 + 60 + 7$$

Números decimais

- De forma geral, um número de n dígitos é representado como

  $a_{n-1} \ldots a_3 a_2 a_1 a_0$

- Cada a_i corresponde a um dígito do número.
- Por exemplo, o valor 367 possui 3 dígitos, sendo $a_2 = 3, \mathrm{a}_1 = 6$ e $a_0 = 7$

Números decimais

- Portanto, qualquer número inteiro na base decimal pode ser representado como

$$a_{n-1} \dots a_2 a_1 a_0 = a_{n-1} * 10^{n-1} + \cdots + a_2 * 10^2 + a_1 * 10^1 + a_0 * 10^0$$

Números binários

- Um computador digital somente entende os valores 0 e 1. Por isso, números precisam ser representados utilizando apenas esses dois valores
- Números representados nessa base possuindo 2 valores (0 e 1) são chamados de números binários
- A representação é exatamente a mesma do que números decimais, com exceção da base, que ao invés de 10 é o valor 2:

$$a_{n-1} \dots a_2 a_1 a_0 = a_{n-1} * 2^{n-1} + \cdots + a_2 * 2^2 + a_1 * 2^1 + a_0 * 2^0$$

Números binários

- Exemplos
- O valor decimal 5 é representado em binário como 101, pois

$$101 = 1 * 2^2 + 0 * 2^1 + 1 * 2^0 = 5$$

- O valor 14:

$$1110 = 1 * 2^3 + 1 * 2^2 + 1 * 2^1 + 0 * 2^0 = 14$$

- O valor 100:

$$1100100 = 1 * 2^6 + 1 * 2^5 + 0 * 2^4 + 0 * 2^3 + 1 * 2^2 + 0 * 2^1 + 0 * 2^0 = 100$$

Números binários

Primeiros 10 valores decimais e o equivalente em binário:

Decimal	Binário
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010

Números binários

- Para identificarmos o equivalente de um número decimal em binário, é útil pensarmos na fórmula

$$\ldots a_6 a_5 a_4 a_3 a_2 a_1 a_0 = \cdots + a_6 * 64 + a_5 * 32 + a_4 * 16 + a_3 * 8 + a_2 * 4 + a_1 * 2 + a_0 * 1$$

- Por exemplo, qual a representação do número 40 em binário?
  - $40 = 32 + 8$, portanto, sua representação em binário é 101000

Números hexadecimais

- Uma outra base numérica muito utilizada em computação é a hexadecimal

$$a_{n-1} \dots a_2 a_1 a_0 = a_{n-1} * 16^{n-1} + \cdots + a_2 * 16^2 + a_1 * 16^1 + a_0 * 16^0$$

- Por exemplo, o número 20 decimal é igual a 14 em hexadecimal, pois

$$14 = 1 * 16^1 + 4 * 16^0 = 20$$

hexadecimal

decimal

Números hexadecimais

- Como números hexadecimais possuem dígitos similares a números decimais, é adequado utilizarmos uma notação que indica a base na qual o número está sendo representado:

$$20_{10} = 14_{16} = 10100_2$$

Números hexadecimais

- A base decimal possui 10 dígitos possíveis (números de 0 a 9)
- A base binária possui 2 dígitos possíveis (números 0 e 1)
- A base hexadecimal possui 15 dígitos possíveis (números de 0 a 15)

- Mas seria muito confuso utilizar os números 10 a 15 como dígitos. Com isso, são utilizadas as letras A, B, C, D, E, F para representar esses números.
- Por exemplo, o número 28_{10} é igual a $1C_{16}$, pois

$$1C = 1 * 16^1 + 12 * 16^0$$

Números hexadecimais

- Exemplos
- O valor decimal 5_{10} é representado em hexadecimal como 5_{16}, pois

  $5_{16} = 5 * 16^0$
- O valor 172_{10} é representado como AC_{16}, pois

  $\mathrm{AC}_{16} = 10 * 16^1 + 12 * 16^0$
- O valor 100_{10} é representado como 64_{16}, pois:

  $64_{16} = 6 * 16^1 + 4 * 16^0$

Números hexadecimais

- A utilidade de números hexadecimais é que os dígitos da base representam todas as combinações possíveis de 4 dígitos da base binária

Números hexadecimais

Binário	Hexadecimal	Decimal
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

Números hexadecimais

- Uma sequência de dígitos binários pode ser representada pelos dígitos hexadecimais equivalentes:

  1010 1110 0010 0110 -> AE26

Números binários e hexadecimais

- Em Python (e em C), podemos representar números nas bases binária e hexadecimal:

```
# Número 15 representado em decimal
num = 15
# Número 15 representado em binário
num_bin = 0b1111
# Número 15 representado em hexadecimal
num_hex = 0xf
```

- Não há nada de especial sobre o termo `0b1111`. Ele é exatamente a mesma coisa que o número 15.

Números binários e hexadecimais

- Python fornece funções para conversão de um número inteiro em uma string representando o número na base binária e hexadecimal:

```
num = 15
# Gera uma string representando o número em binário
num_bin = bin(num)
# Gera uma string representando o número em hexadecimal
num_hex = hex(num)
```

- Uma string representando um número binário e hexadecimal pode ser convertida de volta para inteiro utilizando a função int, que recebe como segundo argumento a base utilizada para a conversão:

```
# Converte a string para um inteiro usando base 2 (binário)
num = int(num_bin, 2)
# Converte a string para um inteiro usando base 16 (hexadecimal)
num = int(num_hex, 16)
```

Exercício 1 - Binário para decimal

- Faça uma função que receba como entrada uma string representando um número binário. A função retorna o respectivo número decimal.

  *Não utilizar a função int

- Exemplo:

  Entrada: "010101"

  Saída: 21

Solução

```
def bin_para_decimal(str_num):

    n = len(str_num)
    res = 0
    for i in range(0, n):
        if str_num[i]=="1":
            res += 2**(n-i-1)

    return res

def testes():

    str_num = "010101"
    print(bin_para_decimal(str_num))
    print(int(str_num, 2))

    str_num = "11011001"
    print(bin_para_decimal(str_num))
    print(int(str_num, 2))

testes()
```

Exercício 2 - Conversor

- Faça uma função que receba como entrada uma string representando um número na base decimal, binária ou hexadecimal. A função também recebe um valor inteiro indicando a base do número. A função imprime na tela o número em decimal, binário e hexadecimal.

  *Utilize as funções int, bin e hex para converter o número

- Exemplo:

  Entrada:

  ```
  converte("010101", 2)
  ```

  Saída:

  ```
  Decimal: 21
  Binário: 0b10101
  Hexadecimal: 0x15
  ```

Solução

```
def converte(str_num, base):

    num_dec = int(str_num, base)
    num_bin = bin(num_dec)
    num_hex = hex(num_dec)

    print(f"Decimal: {num_dec}")
    print(f"Binário: {num_bin}")
    print(f"Hexadecimal: {num_hex}")

def testes():

    converte('010101', 2)
    converte('17', 10)
    converte('af', 16)

testes()
```

Codificação de texto

Representação de texto

- Caracteres de texto são representados por números. Afinal, qualquer informação em um computador deve ser representada por números
- O mapeamento de cada caractere para o respectivo número é chamado de codificação (encoding)
- O mapeamento de cada número para o respectivo caractere é chamado de decodificação (decoding)
- Um arquivo de texto nada mais é do que um conjunto de números armazenados no computador
- Existem diferentes codificações. A mais simples é a chamada ASCII

Representação de texto

- A codificação ASCII possui 128 caracteres possíveis
- Cada caractere é representado por 7 bits
- De forma equivalente, cada caractere é representado por um número hexadecimal entre 0 e 7F

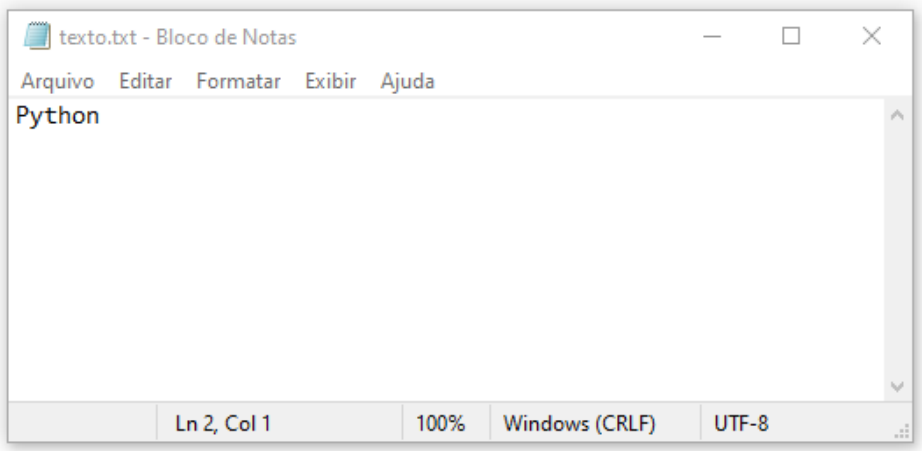
Representação de texto

- Exemplos de códigos ASCII (procure por tabela ASCII ou ASCII table no Google para ver todos):

Símbolo	Decimal	Hexadecimal	Binário
\n	10	0A	0001010
espaço	32	20	0100000
0	48	30	0110000
1	49	31	0110001
2	50	32	0110010
a	97	61	1100001
b	98	62	1100010
c	99	63	1100011
d	100	64	1100100
e	101	65	1100101
f	102	66	1100110
g	103	67	1100111

Representação de texto

Considere um arquivo de texto contendo apenas a palavra "Python\n":



Representação de texto

- Códigos ASCII dos caracteres "Python\n"

Símbolo	Decimal	Hexadecimal	Binário
P	80	50	01010000
y	121	79	01111001
t	116	74	01110100
h	104	68	01101000
o	111	6F	01101111
n	110	6E	01101110
\n	10	0A	00001010

- Portanto, esse arquivo de texto é representado por

  01010000 01111001 01110100 01101000 01101111 01101110 00001010

- Em hexadecimal:

  50 79 74 68 6F 6E 0A

Representação de texto

- A extensão do VSCode "Hex Editor" permite vermos o conteúdo de um arquivo em hexadecimal

Representação de texto

- Python fornece duas funções para codificar e decodificar caracteres de texto:

```
# Retorna um valor inteiro com o código ascii da letra "a"
ascii = ord("a")
# Retorna o caractere correspondente ao código ascii
carac = chr(ascii)
```

Exercício 3 - Minúscula

- Escreva uma função que receba como entrada um caractere e retorne True se o caractere for minúsculo.

  *Não utilizar o método islower() ou similares

Solução

```
def minuscula(carac):

    # Códigos ASCII dos caracteres a e z
    ascii_a = 97
    ascii_z = 122

    codigo = ord(carac)
    if ascii_a<=codigo and codigo<=ascii_z:
        return True
    else:
        return False

def testes():

    print(minuscula("a"))
    print(minuscula("A"))
    print(minuscula("5"))
    print(minuscula("/"))

testes()
```