

## Anotações

### Codificações

Yuri Kaszubowski Lopes

UDESC

YKL (UDESC)

Codificações

1 / 15

---

---

---

---

---

### Revisão: Ponto Flutuante

- Para armazenar um valor binário normalizado arbitrário na memória, como  $1,1111_2 \times 2^3$ , quais campos são importantes?
  - ▶ Não precisamos armazenar o 1 antes do ponto binário, nem a base.
  - ▶ Devemos armazenar:
    - \* Sinal (0 positivo, 1 negativo)
    - \* Exponente
    - \* mantissa (parte à direita do ponto binário)
- Os valores então sempre tem o formato:
  - ▶  $\pm 1, mmmmmmm \times 2^{eeeeee}$
  - ★ *mmmmmmmm* é a **mantissa** (ou fração)
  - ★ *eeeeee* é o **exponente**
  - ★ Armazenamos apenas a mantissa, o expoente, e o sinal

## Anotações

---

---

---

---

---

YKL (UDESC)

Codificações

2 / 15

### Exercícios extras de ponto flutuante

- Qual o maior e o menor valor que podem ser representados em ponto flutuante de precisão dupla e simples (desconsiderando infinito)? Quais são seus equivalentes em decimal?
- Exiba os seguintes valores em ponto flutuante. Quando necessário trunque (ignore os bits que não couberem) os valores. Faça o desenho da memória como nos exemplos e coloque os endereços dos bits (para deixar claro a ordem dos bits).
  - ▶  $-16,015625_{10}$ 
    - \* Em precisão simples
    - \* Em precisão dupla
  - ▶  $-0,1_{10}$ 
    - \* Em precisão simples
    - \* Em precisão dupla
  - ▶  $0,1_{10}$ 
    - \* Em precisão simples
    - \* Em precisão dupla
  - ▶  $0,125_{10}$ 
    - \* Em precisão simples
    - \* Em precisão dupla
    - \* Em meia precisão: 10 bits para mantissa, 5 para expoente e 1 para sinal

## Anotações

---

---

---

---

---

YKL (UDESC)

Codificações

3 / 15

## Codificações

- Quando representamos letras, palavras, números ... por um grupo especial de símbolos, estamos criando uma codificação
  - ▶ Cada letra tem seu próprio grupo, que podemos chamar de código
  - ▶ Exemplo de codificação: código Morse
- Precisamos de codificações para representar caracteres, letras especiais, símbolos, ... em binário
- Como as letras, símbolos, ... são representados em nossos computadores?

### Anotações

---

---

---

---

---

---

## ASCII

- Uma das formas mais simples de representações é o código ASCII:
  - ▶ American Standard Code for Information Interchange
- Código de 7 bits
  - ▶ Em um byte há 8 bits, mas ASCII define código com 7 bits
  - ▶ Quantos códigos diferentes temos?
    - \*  $2^7 = 128$  códigos
- Representar todos os caracteres do seu teclado, além de códigos de controle (e.g. pulo de linha \n)
- Veja <https://www.rapidtables.com/code/text/ascii-table.html>

### Anotações

---

---

---

---

---

---

## Tabela ASCII

```
1 int main() {
2     char c1 = '\0'; //caractere NULL - 0000 0000 na memória
3     char c2 = 'a'; //caractere a - 0110 0001 na memória
4     char c3 = '9'; //caractere 9 - 0011 1001 na memória
5
6     return 0;
7 }
```

- Note que o caractere '9' (0011 1001) tem um valor diferente do número  $9_{10}$  ( $1001_2$ )

### Anotações

---

---

---

---

---

---

## Observação

- Nossas máquinas são comumente endereçadas a byte
  - ▶ Cada endereço de memória suporta exatamente 1 byte
  - ▶ Você aprenderá detalhes na disciplina de Arquitetura de Computadores
- Sendo assim, comumente um `char` vai ocupar 1 byte (8 bits), e não 7 bits
  - ▶ Os valores padrão sempre vão começar com um 0
  - ▶ O bit extra é muitas vezes utilizado para se criar extensões da tabela ASCII
    - ★ Contendo por exemplo caracteres específicos de determinados alfabetos, como o 'ç'
    - ★ Ou para utilização de um sistema de paridade: Detecção de erros
- O código ASCII foi um dos primeiros padrões a serem adotados em larga escala
  - ▶ Possibilitou que as máquinas se comuniquem
  - ▶ Se cada máquina utiliza sua própria codificação, fica difícil a comunicação entre elas

## Anotações

---

---

---

---

---

---

---

---

## Truques

### Converter de ASCII para 0-9

- A tabela ASCII foi criada para que conversões sejam feitas de maneira simples e rápida
- Como converter de binário para ASCII, e vice-versa?

Número	ASCII	Binário
0	011 0000	0
1	011 0001	1
2	011 0010	10
3	011 0011	11
4	011 0100	100
5	011 0101	101
6	011 0110	110
7	011 0111	111
8	011 1000	1000
9	011 1001	1001

## Anotações

---

---

---

---

---

---

---

---

## Truques

### Converter de ASCII para 0-9

- Basta ligar/desligar os bits indicados! Podemos usar uma operação lógica, o que pode ser mais rápido (veremos adiante) ou somar/subtrair 48

Número	ASCII	Binário
0	011 0000	0
1	011 0001	1
2	011 0010	10
3	011 0011	11
4	011 0100	100
5	011 0101	101
6	011 0110	110
7	011 0111	111
8	011 1000	1000
9	011 1001	1001

## Anotações

---

---

---

---

---

---

---

---

## Truques

### Maiúsculo/Minúsculo

- Como converter entre maiúsculo ou minúsculo?
- Basta ligar/desligar os bits indicados! Podemos usar uma operação lógica, o que pode ser mais rápido ou somar/subtrair 32
  - ▶ Note que  $2^5 = 32$

Códigos ASCII		Códigos ASCII	
a : 01100001	A : 01000001	a : 01100001	A : 01000001
b : 01100010	B : 01000010	b : 01100010	B : 01000010
c : 01100011	C : 01000011	c : 01100011	C : 01000011
d : 01100100	D : 01000100	d : 01100100	D : 01000100
e : 01100101	E : 01000101	e : 01100101	E : 01000101
f : 01100110	F : 01000110	f : 01100110	F : 01000110
g : 01100111	G : 01000111	g : 01100111	G : 01000111
h : 01101000	H : 01001000	h : 01101000	H : 01001000
...	...	...	...
x : 01111000	X : 01011000	x : 01111000	X : 01011000
y : 01111001	Y : 01011001	y : 01111001	Y : 01011001
z : 01111010	Z : 01011010	z : 01111010	Z : 01011010

### Anotações

---

---

---

---

---

---

---

---

---

## Outras codificações

- Existem diversas outras codificações que utilizamos nos nossos dia a dia
- Exemplos:
  - ▶ Compatíveis com o ASCII (adicionam novos códigos, mas são compatíveis com o ASCII)
    - ★ ASCII estendido
    - ★ UTF-8
    - ★ CP1252
  - ▶ Código BCD
  - ▶ Gray Code

### Anotações

---

---

---

---

---

---

---

---

---

## Observações

- Se você está comunicando dois dispositivos, uma das coisas que você deve levar em consideração é a codificação
- Exemplo: um dispositivo utiliza BCD para representar números, e outro ASCII
  - ▶ Você vai precisar converter

### Anotações

---

---

---

---

---

---

---

---

---

## Exercícios extras

- 1 Crie um programa em C que exibe todos os caracteres ASCII, e seus códigos ASCII em binário e em decimal.
- 2 Pesquise sobre UTF-8. Como ele é formado? Quantos bits ocupa? Como ele é compatível com ASCII?

Anotações

---

---

---

---

---

---

## Referências

- TOCCI, R.J.; WIDMER,N.S. **Sistemas digitais: princípios e aplicações**. 11a ed, Prentice-Hall, 2011.
- RUGGIERO, M.; LOPES, V. da R. **Cálculo numérico: aspectos teóricos e computacionais**. Makron Books do Brasil, 1996.
- NULL, L.; LOBUR, J. **Princípios Básicos de Arquitetura e Organização de Computadores**. 2014. Bookman, 2009. ISBN 9788577807666.

Anotações

---

---

---

---

---

---

Anotações

---

---

---

---

---

---