Programação Imperativa e Funcional

Apresentação

erico.souza.teixeira
est@cesar.school

O objetivo

- **Apresentar os conceitos de programação funcional e imperativa**
  - **linguagem C e Haskell**

Informações

- **Horário**
  - **segundas e quintas 1o.3o - 12.oo**
  - **Chama é obrigatória**
- **Nota**
  - **Avaliações: 12.o9, 17.1o e 28.11 (8o%)**
  - **Listas: 12.o9, 17.1o e 28.11 (2o%)**
  - **Segunda Chamada: o9.12**
  - **Final: 16.12**
- **Slack:** `imp-e-func`
- **Google classroom:** `osq70xb`

Programação Imperativa e Funcional

Introdução à programação imperativa

erico.souza.teixeira
est@cesar.school

O conceito

- Ações que mudam o estado de um programa
- Uma sequência da comandos
- Como deve ser feito e não o que deve ser feito
- Programação estruturada
  - sequência, decisão e iteração

Por que Linguagem C?

- **Portabilidade -> Performance/Eficiência**
- **Estável**
- **Manipulação eficiente de memória**
- **Fácil iteração com *hardware***
- **https://www.tiobe.com/tiobe-index/**

Olá Mundo

Olá Mundo

```c
#include <stdio.h>

int main() {

        printf("Olá Mundo!”);
        return 0;

}
```

Estilo de codificação

- **Padrão do *kernel* do Linux**
  - **indentação**
  - **longas linhas**
  - **localização das chaves**

Referências

- Peter Van Roy , *Programming Paradigms for Dummies: What Every Programmer Should Know*
- Schildt, H. , *C Completo e Total*
- Deitel, H. e Deitel, P., *C Como Programar*
- Jamsa, K., *Programando em C/C++ "A bíblia"*
- Kernighan, B.W., *C: a Linguagem de Programação*
- http://learnyouahaskell.com
- https://www.kernel.org/doc/html/v4.10/process/coding-style.html

Programação Imperativa e Funcional

Comandos de Seleção

erico.souza.teixeira
est@cesar.school

Olá Mundo ou/*or Hello World*

Operador ?

```
(condicional)?(faça algo):(outro algo)
```

- **Retornar o maior valor**

```
(a > b) ? a : b;
```

switch

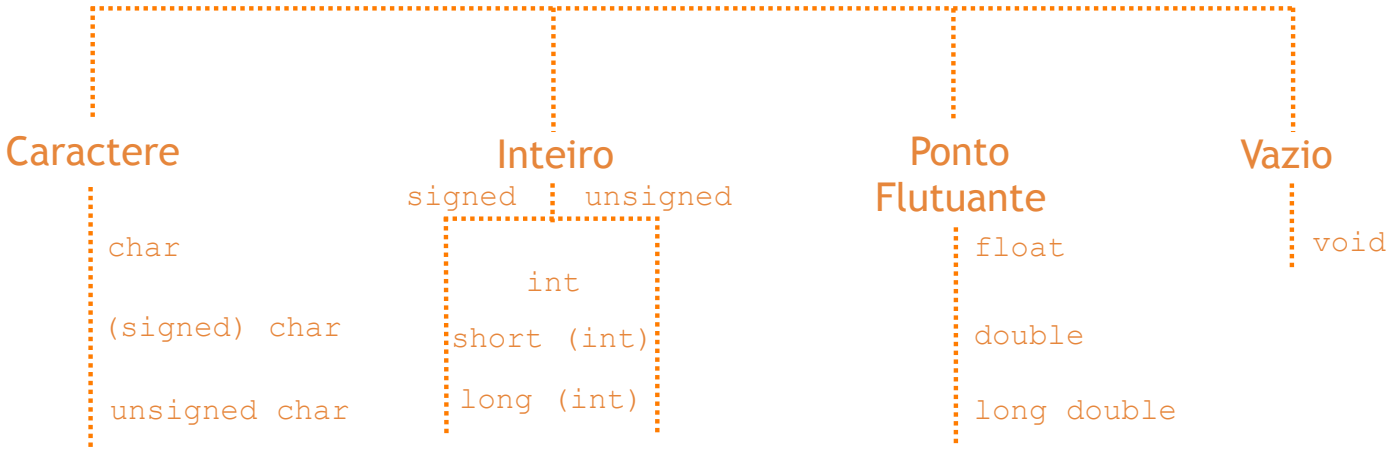
```
switch (/* int ou enum ou char */) {
case /* valor em potencial*/:
  /* código */
case /* um diferente valor em potencial */:
  /* código */
default:
  /* código */
}
```

Programação Imperativa e Funcional

Variáveis e Tipos de Dados

erico.souza.teixeira
est@cesar.school

Tipos básicos de dados



- **Na linguagem C não há o tipo *booleano***
  - **0 == Falso**
  - **Qualquer outro valor será Verdadeiro**
  - `stdbool.h`

Caracteres de Barra Invertida

Código	Significado
\b	Retroceso
\f	Alimentação de formulário
\n	Nova linha
\r	Retorno de carro
\t	Tabulação horizontal
\”	Aspas duplas
\’	Aspas simples
\0	Nulo
\\	Barra invertida
\v	Tabulação veritcal
\a	Alerta
\N	Constante octal
\xN	Constante hexadecimal

Tipos básicos de dados

Tipo	Tamanho	Interval	Precisão
`float`	4 byte	1.2E-38 to 3.4E+38	6 casas decimais
`double`	8 byte	2.3E-308 to 1.7E+308	15 casas decimais
`long double`	10 byte	3.4E-4932 to 1.1E+4932	19 casas decimais

Formatos

Tipo	Formato
char	%c
signed char	%c
unsigned char	%c
short int	%hd
unsigned long int	%lu
int	%d
unsigned int	%u

Identificador

- **Regras**
  - **Caracteres alfanuméricos (`a-z A-Z 0-9`) ou sublinhado (_)**
  - **Primeiro caractere não pode ser numérico**
    - **Mas evitar _**
  - ***Case sensitive***
    - `NOME ≠ nome ≠ NoMe`
  - **Tamanho máximo depende do compilador**
  - **Não pode usar palavras reservadas**
    - **Palavras chaves C**
    - **Funções definidas pelo usuário ou bibliotecas**

Palavras Chaves

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
const	extern	return	union
char	float	short	unsigned
continue	for	signed	volatile
default	goto	sizeof	void
do	if	static	while

Programação Imperativa e Funcional

Operadores Aritméticos, Lógicos e Relacionais

erico.souza.teixeira
est@cesar.school

Par ou Ímpar

Operadores Aritméticos

- **Adição (+) Subtração (-) Multiplicação (*) Divisão (/) e Módulo (%)**
  - `(-5) % 2 == -1`
  - `(-5) / 3 == -1`
- `math.h`
  - **Operado de potência (`pow`)**
  - **Módulo para float (`fmod`)**

Operadores Aritméticos

```
x += y      ==      x = x+y
x -= y      ==      x = x-y
x *= y      ==      x = x*y
x /= y      ==      x = x/y
x %= y      ==      x = x%y
```

Operadores Aritméticos

- **Incremento (++) Decremento (--)**

```
x = x + 1 == ++x == x++
x = x - 1 == --x == x--
```

- **Mas quando aplicados em uma expressão ...**

```
x = 10;
y = ++x;
   ≠
x = 10;
y = x++
```

Operadores Relacionais e Lógicos

- > >= < <= == !=
- && || !

Avaliação por circuito curto

```
int a = 20;
if (0 < a || a < 10){
  printf (”%d”, a);
}
```

Precedência

- `x = y=3, y+1`
- `x = (y=3, y+1)`

Precedência

Maior	() [] ++ -- (sufixo) .	Esq.-Dir.
	! ++ -- (prefixo) -	Dir.-Esq.
	* / %	Esq.-Dir.
	+ -	Esq.-Dir.
	< <= > >=	Esq.-Dir.
	== !=	Esq.-Dir.
	&&	Esq.-Dir.
	||	Esq.-Dir.
	= += -= *= /=	Dir.-Esq.
Menor	,	Esq.-Dir.

Exercício

Escreva um programa que determina se o aluno foi ou não aprovado na disciplina. Não esqueça de considerar a situação de segunda chamada e final. Considere que uma nota negativa indica que o aluno faltou a avaliação.