

Aula 2: Introdução a Linguagem C

Prof. Sérgio Montazzoli Silva
smsilva@uel.br

Sumário

- História
- Ponto de entrada de um programa (função **main**)
- Bibliotecas
- Blocos, encerramento de linha e indentação
- Função de impressão na tela (**printf** - parte 1)
- Modelo básico

História

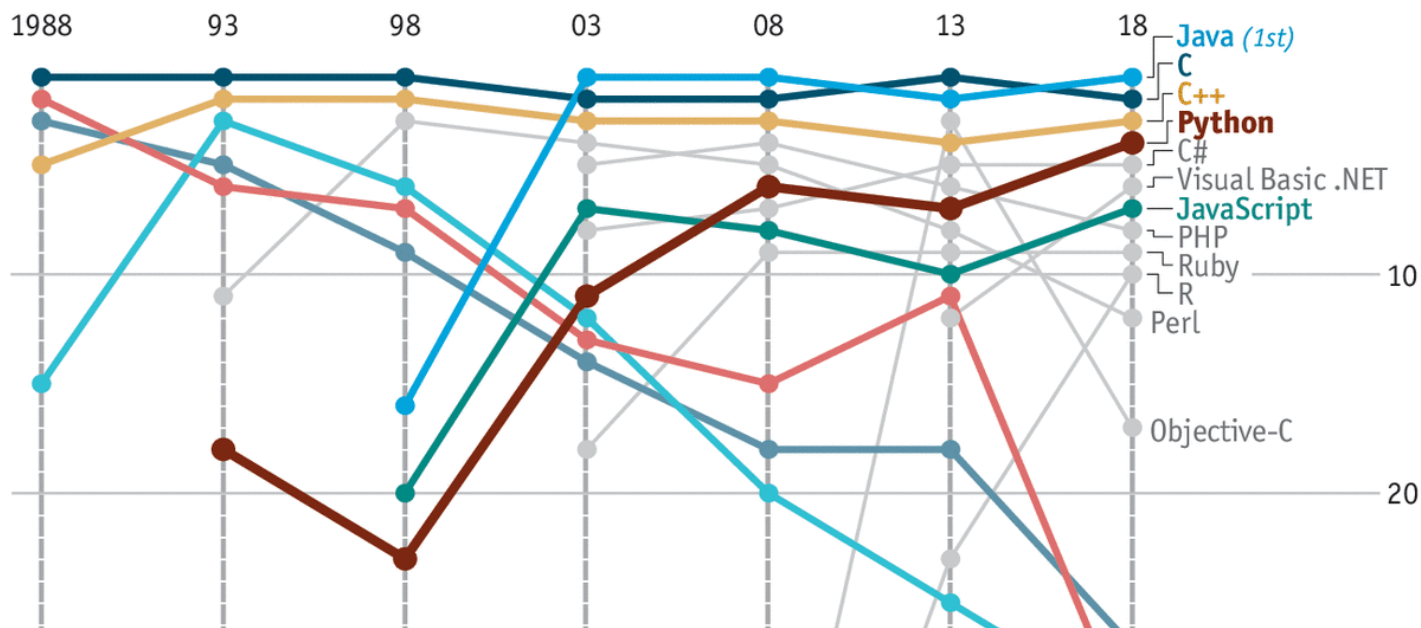
- A linguagem C foi criada em 1972 nos laboratórios da BELL
 - C foi o nome dado ao aprimoramento da linguagem B
- Entre 1972 e 1989, houveram várias versões do C
 - Neste período, a linguagem ganhou muita popularidade
- Em 1989, o Instituto Nacional Americano de Padrões ou ANSI (sigla em inglês) formou um comitê para estabelecer um padrão a linguagem
 - A versão ficou popularmente conhecida como ANSI C
 - ... ou também C89
- Desde então, outras versões foram criadas, aumentando o número de recursos na linguagem:
 - C99 (1999), C11 (2011), C18 (2018)

História

- Popularidade em gráficos:

Code of conduct

Ranking of programming languages*



Fonte: The Economist

História

- Considerando que o C++ é uma variante do C, então podemos dizer que:
 - **C é a linguagem de programação mais utilizada e popular no mundo**
- Serve de base para os principais Sistemas Operacionais:
 - Windows, Linux, Mac e Android
- Está presente em servidores web, bancos de dados, eletrodomésticos, carros, aviões, barcos...
- Dennis Ritchie, criador do C, morreu em 2011
 - ... uma semana após a morte de Steve Jobs

Exemplo Básico

- Programa que soma duas variáveis tipo inteiro (a e b) e mostra o resultado na tela:

```
#include <stdlib.h>
#include <stdio.h> } Bibliotecas incluídas
```

```
int main() ← Ponto de entrada
{ ← Início do bloco
```

```
int a = 10;
int b = 20;
int r; } Criação de variáveis inteiras
```

```
r = a + b; ← Operações de soma e atribuição
```

```
printf("%d", r); ← Função que escreve na tela
} ← Fim do bloco
```

Todo arquivo C deve ser salvo utilizando a extensão .c
Ex: programa.c



Função main()

- Programas complexos costumam ter milhares e milhares de linhas de código
 - Linux tem 12 milhões
- Como saber por onde iniciar a execução?
- Todo programa necessita de um ponto de entrada!
- Em C, C++ e Java (... e muitas outras) uma função com nome **main** é usada para esse propósito
- Apenas UMA função main é permitida por programa
- O compilador irá considerar esta função como entrada do seu programa

Função main()

- Em C, a função **main** é definida da seguinte forma:
 - ```
int main() {
 ...Código...
}
```
- Mas podem haver outras variações, como:
  - ```
int main(void) { ...Código... }
```
 - ```
int main(int argc, int **argv) { ...Código... }
```

    - Esta aceita uma lista de argumentos como entrada
    - Não será visto nesse curso



# Função main()

```
#include <stdlib.h>
#include <stdio.h>
```

```
int funcao1() {
 // ...
}
```

```
int funcao2() {
 // ...
}
```

```
int main() {
 // ...
 // Código
 // ...
}
```

Ponto de entrada



**Compilador: qual o ponto de entrada do seu programa?**



# Compilar sem a função main()

- Meu programa sem função **main**:

```
#include <stdio.h>

int outro_nome() {
 printf("Vai dar erro!");
 return 0;
}
```

- Erro de compilação:

```
In function `main':
undefined reference to `WinMain'
[Error] ld returned 1 exit status
```



# main() em pseudocódigo

- Pseudocódigo pode muitas vezes ser um pouco subjetivo
- Nessa disciplina vamos adotar “**algoritmo principal**” para representar a função main.
  - Exemplo:

```
algoritmo principal
início
 $x \leftarrow 1$
 $y \leftarrow 2$
 resultado $\leftarrow x + y$
 imprimir(resultado)
fim
```

# Outros aspectos da linguagem C

- Inclusão de bibliotecas:
  - Use a macro `#include <nome_da_biblioteca.h>` no topo do seu código C
    - `stdlib.h`
      - Standard Library (Biblioteca padrão)
        - Funções de alocação de memória, conversão de tipos de dados, controle de processos, definições de macros, etc...
          - Exemplos: `atof`, `atoi`, `system`, `NULL`, `EXIT_SUCCESS`
    - `stdio.h`
      - Standard Input/Output (Biblioteca padrão de entrada e saída)
        - Define funções para leitura e escrita de dados
          - Exemplo: `printf`, `scanf`

# Outros aspectos da linguagem C

- TODA LINHA DENTRO DE UM BLOCO { } TERMINA

COM ;

```
#include <stdlib.h>
#include <stdio.h>
```

```
int main() {
```

```
int x = 10;
```

```
int y = 20;
```

```
int r = x*y;
```

```
printf("Resultado: %d", r);
```

```
}
```

# Outros aspectos da linguagem C

- Sempre que iniciar um bloco, use a tecla TAB para indentação;

```
#include <stdlib.h>
#include <stdio.h>

int main() {
 ← int x = 10;
 ← int y = 20;
 ← int r = x*y;

 ← printf("Resultado: %d", r);
}
```

# Função printf() – Parte 1

- Função utilizada para **imprimir** informações na tela
- Faz parte da biblioteca stdio
  - Usar no início: **#include <stdio.h>**
- printf() possui um ou mais parâmetros:
  - Um parâmetro:
    - `printf("Texto simples para impressão!");`
  - Dois ou mais parâmetros:
    - `printf("Valor 1: %d", var1);`
    - `printf("Valor 1: %d, Valor 2: %d", var1, var2);`
    - `printf("Valor 1: %d, Valor 2: %d, Valor 3: %d", var1, var2, var3);`
    - .....

# Função printf() – Parte 1

- Símbolo de % (porcentagem) indica o tipo de dado
- Sempre que for utilizado dentro do primeiro parâmetro, a variável correspondente deve ser passada nos argumentos
- Exemplo:
  - Supondo que existam as variáveis inteiras:
    - `int x = 10;`
    - `int y = 20;`
    - Int (ou tipo inteiro) é representado por %d
  - Então:
    - Para imprimir x: `printf("Variavel x: %d", x);`
    - Para imprimir y: `printf("Variavel x: %d", y);`
    - Para imprimir ambos:
      - `printf("Variavel x: %d, variavel y: %d", x, y);`



# Função printf() – Parte 1

- Alguns caracteres especiais que podem ser usados:
  - \n : quebra de linha
  - \t : tab
  - \" : aspas duplas
  - \\ : contrabarra
- Em pseudocódigo:
  - Utilize a palavra “imprimir”
    - imprimir “Ola mundo!”

# Modelo (template)

- Modelo padrão:

```
#include <stdlib.h>
#include <stdio.h>

int main() {

 // Seu código vai aqui!!!
 // ...

 system("PAUSE");
}
```

# Exercícios em aula

1. Escreva um programa, em **pseudocódigo**, que crie as variáveis tipo inteiro  $x=4$ ,  $y=40$  e  $z=20$ , coloque o resultado da operação  $(x*z)/y$  em uma quarta variável  $r$ , e imprima o resultado
2. Agora escreva o mesmo programa anterior, **em C**
3. Escreva um programa em C, que imprima na tela exatamente o seguinte texto:

```
Quebra de linha
 TAB
“uma frase
 com quebra e tab”
Equacao $x*z/y$
```

# Próxima aula

- Tipos e Variáveis
- Função **printf()** – Parte 2
- Operações com variáveis