



Programação de Computadores I

Introdução a Linguagem C
Princípios de Programação em C

Gil Eduardo de Andrade





Conceitos Iniciais: Linguagem C

Introdução

- É uma linguagem estruturada;
- Foi desenvolvida na década de 70 – permitiu a criação do sistema UNIX ;
- Serviu como base para criação de outras linguagens: C++ e Java;





Conceitos Iniciais: Linguagem C

Características:

- Estruturalmente simples, pós-compilação gera códigos executáveis pequenos e rápidos;
- Portável, programas escritos em C para uma plataforma (Linux) pode ser facilmente adaptável para outra (Windows);
- Linguagem de nível relativamente baixo, possuindo elementos Assembly;





Conceitos Iniciais: Linguagem C

Características:

- Por ser estruturada, permite codificação dividida em módulos;
- Possibilita a inclusão de várias rotinas definidas pelo programador;
- Permite ao programador criar suas bibliotecas de forma customizada;





Conceitos Iniciais: Linguagem C

Estrutura básica de um arquivo “.c”

```
#include <stdio.h> →
```

Declaração das Bibliotecas Utilizadas

```
int main() { →
```

Declaração da função principal – indica onde se inicia o código-fonte em C

```
return 0; →
```

Retorna o valor “0” – final da execução

```
} ↘
```

Indica onde termina o código-fonte em C relativo a função principal





Conceitos Iniciais: Linguagem C

Bibliotecas do Sistema:

- Possuem um conjunto específico de funções já implementadas e que podem ser utilizadas pelo programador;
- A biblioteca **`<stdio.h>`** (standard in/out) é um exemplo, possui funções que permitem utilizar a entrada (teclado) e saída (monitor) padrão do computador;





Conceitos Iniciais: Linguagem C

Função Principal “main()”:

- Todo arquivo “.c” deve, obrigatoriamente, possuir uma função *main()*;
- A execução do código se inicia nela (*main()*);
- As chaves “{” e “}” servem para indicar onde a codificação da função *main()* inicia e acaba, respectivamente;





Princípios de Programação em C

Variáveis em C:

- Variáveis são espaços na memória RAM, reservados para que valores possam ser armazenados e resgatados posteriormente;
- A linguagem C possui dois tipos de variáveis, locais (nesta aula) e globais (futuramente);
- A linguagem C é dita fortemente tipada, pelo fato de suas variáveis possuírem tipos (int, char);





Princípios de Programação em C

Tipos básicos de variáveis em C:

- ***int***: armazena um valor inteiro, por exemplo: *0, 2, -56, 200*;
- ***char***: armazena um caractere, por exemplo: *a, z, D, X, q*;
- ***float***: armazena um valor ponto flutuante (com vírgula), por exemplo: *3.14, -12.6, 0, 100.56*;





Princípios de Programação em C

Declaração de variáveis em C:

- A declaração das variáveis em C deve ser feita no início do código, abaixo da função *main()*;
- É obrigatório declarar todas as variáveis que serão utilizadas ao longo da codificação;
- Na declaração deve-se especificar o tipo da variável (*int*, *char*) e depois seu nome;





Princípios de Programação em C

Declaração de variáveis em C:

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

```
int main() {
```

Indica o tipo da
variável que está
sendo declarada.

```
int nomeA, nomeB;  
char nomeC;  
float nomeD, nomeE;
```

Indica o nome da
variável que está
sendo declarada.

```
return 0;
```

```
}
```

Observação: o nome das variáveis não pode conter caracteres especiais como acento e espaço, nem iniciar com números.





Princípios de Programação em C

Final de linha/comando em C (“;”):

- Na linguagem C é necessário indicar ao seu compilador o término de uma linha de comando;
- Para tal existe o caractere reservado “;”;
- Por isso, após declarar uma variável, por exemplo, utiliza-se o ponto-vírgula;





Princípios de Programação em C

Atribuindo valores a variáveis (“=”):

- Variáveis são utilizadas para armazenar valores, sendo assim foi preciso definir um operador para tal;
- Este operador é o caractere reservado “=”;
 - *Ex.: valorA = 12;*



Princípios de Programação em C

Atribuindo valores a variáveis (“=”):

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

```
int main() {
```

```
    int nomeA, nomeB;  
    char nomeC;  
    float nomeD, nomeE;
```

```
    nomeA = 12;  
    nomeB = 21;  
    nomeC = 'A';  
    nomeD = 3.14;  
    nomeE = 84.2;
```

```
    return 0;
```

```
}
```

Observe que variáveis do tipo *int* recebem valores inteiros, assim como variáveis do tipo *float* valores com vírgula. A variável do tipo *char* recebe caracteres, e observe que estes devem estar entre aspas simples.



Princípios de Programação em C

Operações matemáticas com variáveis :

- **Adição (+):** permite somar valores, sejam eles estáticos ou armazenados em variáveis;
 - *Ex.: nomeA = nomeB + 10;*
- **Subtração (-):** permite subtrair valores, sejam eles estáticos ou armazenados em variáveis;
 - *Ex.: nomeB = nomeB – nomeA;*



Princípios de Programação em C

Operações matemáticas com variáveis :

```
#include <stdio.h>

int main() {

    int nomeA, nomeB;
    char nomeC;
    float nomeD, nomeE;

    nomeA = 12;
    nomeB = 21;
    nomeA = nomeB + 10;

    return 0;
}
```

Observe que antes de efetuar qualquer operação matemática sobre as variáveis foram atribuídos valor as mesmas. Isso permite concluir que a programação é uma tarefa sequencial e que a ordem com que essa sequência de comandos é colocada influencia diretamente o funcionamento do código durante a sua execução.

Princípios de Programação em C

Operações matemáticas com variáveis :

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

```
int main() {
```

```
    int nomeA, nomeB;  
    char nomeC;  
    float nomeD, nomeE;
```

```
    nomeA = 12;  
    nomeB = 21;  
    nomeB = nomeB - nomeA;
```

```
    return 0;
```

```
}
```

Observe que é possível efetuar uma operação matemática utilizando o valor armazenado em uma variável, ao mesmo tempo em que o resultado dessa operação é armazenado na mesma variável o qual utilizamos seu valor para efetuar o cálculo. Nesse caso a variável *nomeB* recebe a subtração do seu próprio valor pelo valor armazenado em *nomeA*.



Princípios de Programação em C

Operações matemáticas com variáveis :

- **Multiplificação (*)**: permite multiplicar valores, sejam eles estáticos ou armazenados em variáveis;
 - *Ex.: nomeA = nomeB * 10;*
- **Divisão (-)**: permite dividir valores, sejam eles estáticos ou armazenados em variáveis;
 - *Ex.: nomeB = nomeB / nomeA;*





Princípios de Programação em C

Operações matemáticas com variáveis :

```
#include <stdio.h>

int main() {

    int nomeA, nomeB;
    char nomeC;
    float nomeD, nomeE;

    nomeA = 12;
    nomeB = 21;
    nomeA = nomeB * 10;

    return 0;
}
```

```
#include <stdio.h>

int main() {

    int nomeA, nomeB;
    char nomeC;
    float nomeD, nomeE;

    nomeA = 12;
    nomeB = 21;
    nomeB = nomeB / nomeA;

    return 0;
}
```





Princípios de Programação em C

Comandos de Saída (*printf*):

- Os comandos de saída permitem que informações sejam enviadas para dispositivos computacionais (monitores, impressoras);
- O comando de saída mais importante na linguagem C, é o ***printf()***, disponível na biblioteca ***<stdio.h>***;





Princípios de Programação em C

Sintaxe do comando *printf()*:

- O comando *printf()*, permite que textos e valores contidos em variáveis sejam apresentados ao usuário;
- O texto que deseja-se apresentar deve ser escrito, entre aspas duplas, dentro dos parênteses do comando;
 - *Ex.: printf("Gil Eduardo de Andrade");*





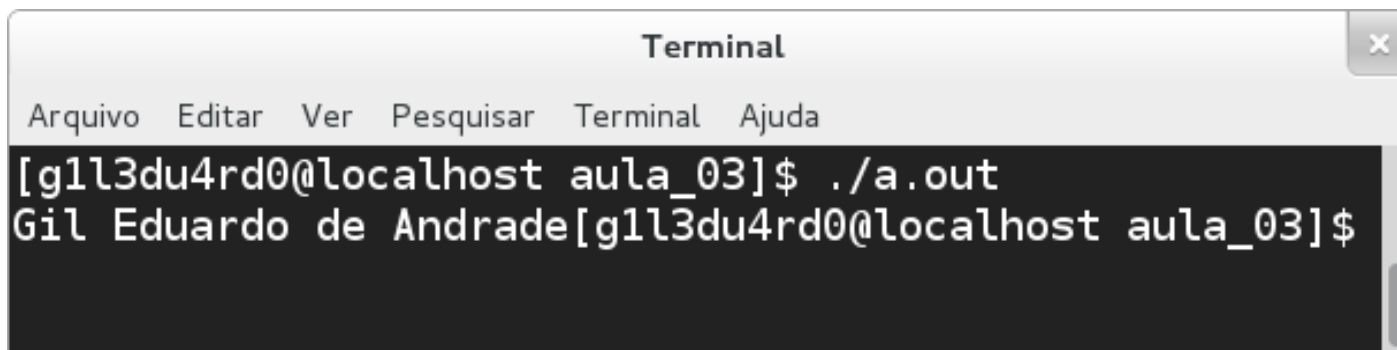
Princípios de Programação em C

Sintaxe do comando *printf()*:

```
#include <stdio.h>

int main() {

    printf("Gil Eduardo de Andrade");
    return 0;
}
```



The image shows a terminal window titled "Terminal" with a menu bar containing "Arquivo", "Editar", "Ver", "Pesquisar", "Terminal", and "Ajuda". The terminal content shows the execution of a C program:

```
[g1l3du4rd0@localhost aula_03]$ ./a.out
Gil Eduardo de Andrade[g1l3du4rd0@localhost aula_03]$
```





Princípios de Programação em C

Caracteres especiais para o comando *printf()*:

- “\n”: o ‘barra-n’ permite pular linhas momento em que utilizamos o comando *printf()*;
 - *Ex.: printf(“Gil\nEduardo\nAndrade\n”);*
- “\t” o ‘barra-n’ permite que tabulações sejam criadas quando utilizamos o comando *printf()*;
 - *Ex.: printf(“\tGil\n\t\tEduardo\n\t\t\tAndrade\n”);*





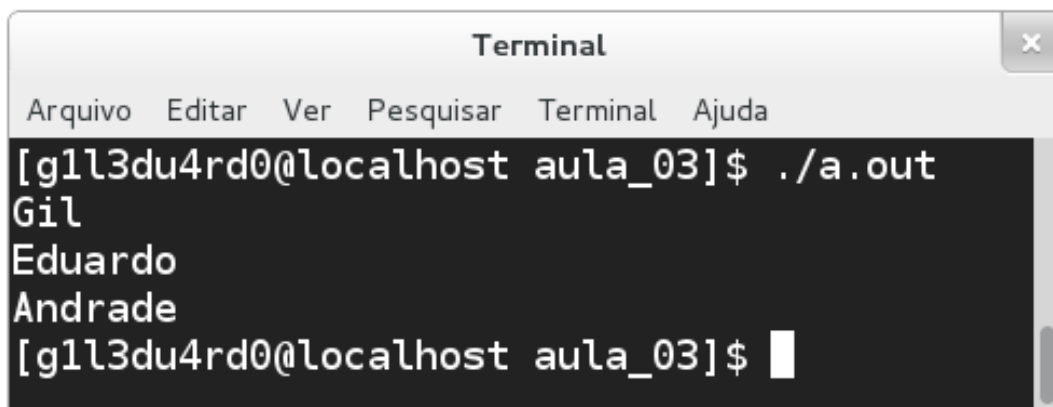
Princípios de Programação em C

Caracteres especiais – `printf("\\n")`:

```
#include <stdio.h>

int main() {

    printf("Gil\\nEduardo\\nAndrade\\n");
    return 0;
}
```



```
Terminal
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
[g1l3du4rd0@localhost aula_03]$ ./a.out
Gil
Eduardo
Andrade
[g1l3du4rd0@localhost aula_03]$
```





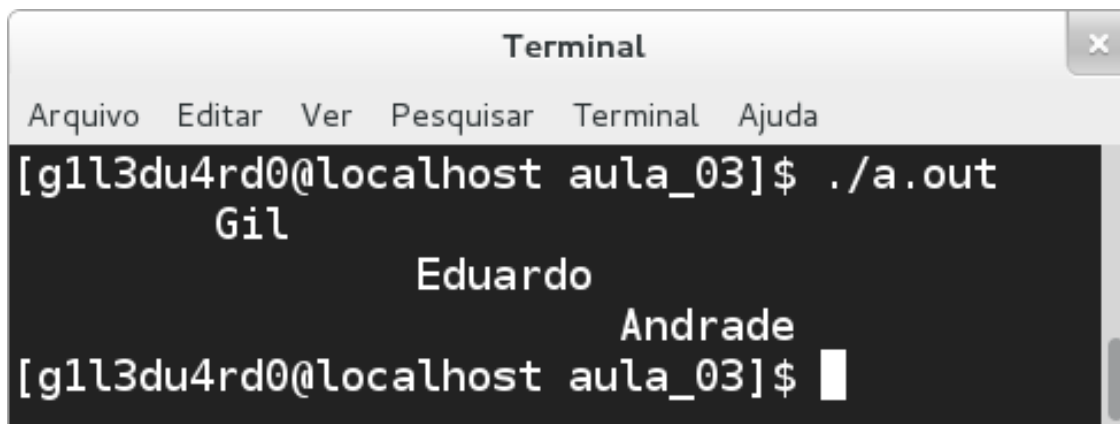
Princípios de Programação em C

Caracteres especiais – `printf("\t")`:

```
#include <stdio.h>

int main() {

    printf("\tGil\n\t\tEduardo\n\t\t\tAndrade\n");
    return 0;
}
```



```
Terminal
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
[g1l3du4rd0@localhost aula_03]$ ./a.out
    Gil
          Eduardo
                Andrade
[g1l3du4rd0@localhost aula_03]$
```





Princípios de Programação em C

printf() para exibir o conteúdo de variáveis:

- “%i ou %d”: o ‘percentual-i ou d’ permite exibir o conteúdo de variáveis inteiras (*int*);
 - *Ex.: printf(“%i”, nomeA);*
- “%c” o ‘percentual-c’ permite exibir o conteúdo de variáveis caractere (*char*);
 - *Ex.: printf(“%c”, nomeC);*





Princípios de Programação em C

printf() para exibir o conteúdo de variáveis:

- “%f”: o ‘percentual-f’ permite exibir o conteúdo de variáveis ponto flutuante (*float*);
 - *Ex.:* `printf(“%f”, nomeD);`
- Observe (em todos os exemplo) que após as aspas duplas, é colocada uma vírgula e logo após esta o nome da variável que deseja-se exibir o conteúdo;





Princípios de Programação em C

printf() para exibir o conteúdo de variáveis:

```
#include <stdio.h>

int main() {

    int nomeA, nomeB;
    char nomeC;
    float nomeD, nomeE;

    nomeA = 12;
    nomeC = 'A';
    nomeD = 3.14;

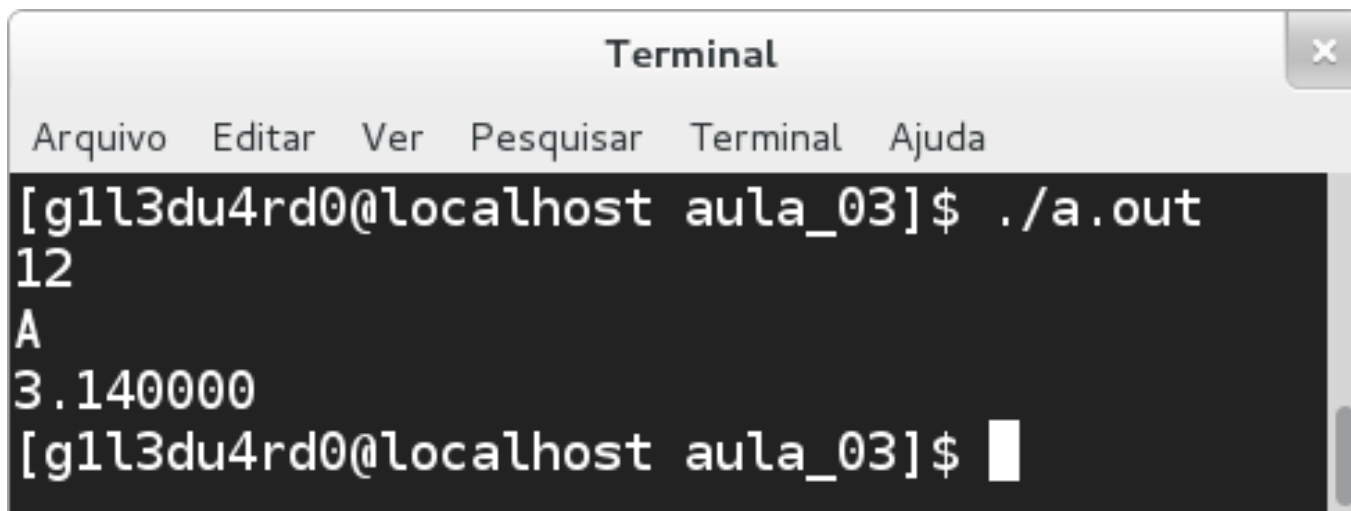
    printf("%i\n%c\n%f\n", nomeA, nomeC, nomeD);
    return 0;
}
```





Princípios de Programação em C

printf() para exibir o conteúdo de variáveis:



```
Terminal
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
[g1l3du4rd0@localhost aula_03]$ ./a.out
12
A
3.140000
[g1l3du4rd0@localhost aula_03]$
```





Princípios de Programação em C

Exemplos Utilizados no Documento

http://www.gileduardo.com.br/ifpr/pci/downloads/pc_exdoc03.zip

Mais Exemplos sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/pci/downloads/pc_ex03.zip

Exercícios sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/pci/downloads/pc_pratica03.dpf

