Linguagem C

Eliane Pozzebon

Material de Apoio de C

- Site da disciplina: http://www.univasf.edu.br/~eliane.pozzebon
  - Slides de aulas
  - Exercícios

- Dev-C++: um ambiente de programação interativo e gratuito para as linguagens C e C++:
  - Site http://www.bloodshed.net/devcpp.html

- Bibliografia:
  - C a linguagem de programação padrão ANSI
  - C completo e total
  - A prática da programação

Material de Apoio e Suporte

- Livros (free)
  - The C Book (Banahan, Brady & Doran)
    http://www.computer-books.us/c_5.php
  - The New C Standard (Jones)
    http://www.knosof.co.uk/cbook/cbook.html

- Outros
  - Material elaborado por outros professores do colegiado Computação
    - Profs. Anderson, Marcelo e Marcus
  - Unicamp: http://www.inf.ufrgs.br/~binsely/tutorialc.pdf
  - PUC: http://www.inf.pucrs.br/~pinho/LaproI/IntroC/IntroC.htm

- Monitoramento

  Monitor: Jesse Nery
  Email: jessenery@hotmail.com
  Locais de atendimento: Laboratórios de Computação 1 e 2

Linguagens de Programação

Uma **linguagem de programação** é um método padronizado para expressar instruções para um computador.

É um conjunto de regras usadas para definir um programa de computador. Uma linguagem permite que um programador especifique precisamente sobre quais dados um computador vai atuar, como estes dados serão armazenados ou transmitidos e quais ações devem ser tomadas sob várias circunstâncias.

Cada linguagem possui um conjunto único de palavras-chaves (palavras que ela reconhece) e uma sintaxe (regras) específica para organizar as instruções dos programas.

Linguagens de Programação

Os programas de computador podem ser escritos em várias linguagens de programação, algumas diretamente compreensíveis pelo computador e outras que exigem passos de tradução intermediária.

As linguagens de programação podem ser divididas em três tipos, com relação à sua similaridade com a linguagem humana:

- **Linguagem de máquina;**
- **Linguagem simbólica;**
- **Linguagem de alto nível.**

Linguagem de máquina

É a linguagem de mais baixo nível de entendimento pelo ser humano e a única, na verdade, entendida pelo processador (CPU).

- É constituída inteiramente de números (0's e 1's), o que torna praticamente impossível entendê-la diretamente. Cada CPU tem seu conjunto único de instruções que definem sua linguagem de máquina, estabelecido pelo fabricante do chip.

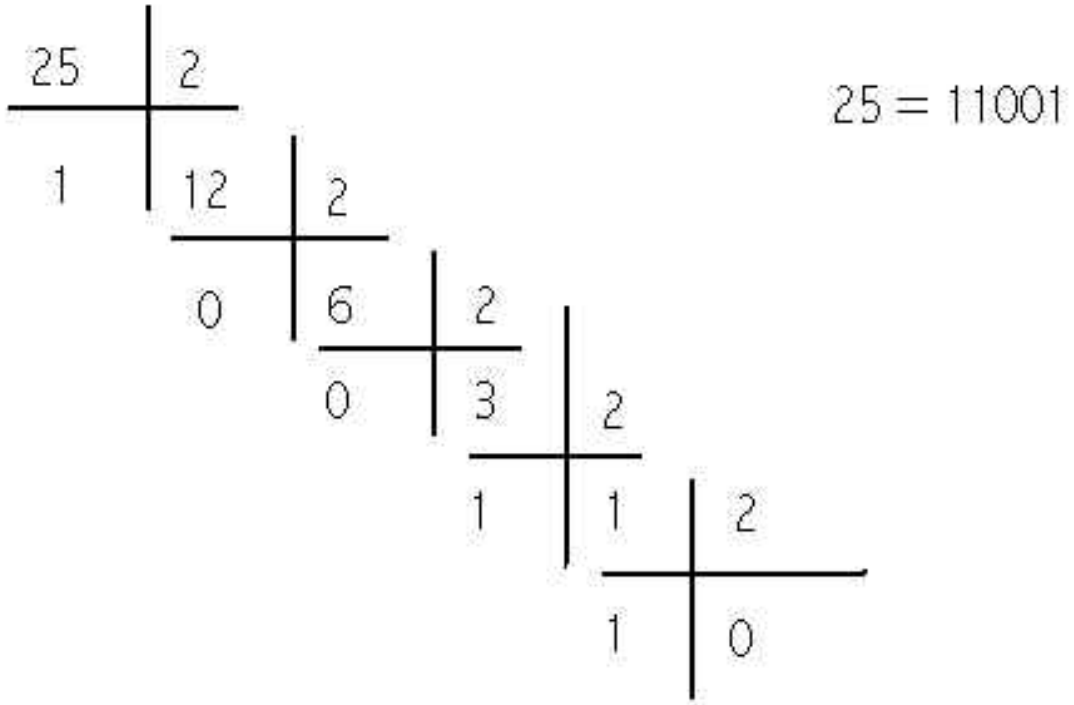
- Uma instrução típica em linguagem de máquina seria algo como:

0100 1111 1010

Essa linguagem é também classificada como uma linguagem de primeira geração.

Linguagem de máquina

EXEMPLO: Vamos converter 25 de decimal para binário.



Linguagem simbólica (Assembly):

É a linguagem de nível imediatamente acima da linguagem de máquina. Ela possui a mesma estrutura e conjunto de instruções que a linguagem de máquina, porém permite que o programador utilize nomes e símbolos em lugar de números.

- A linguagem simbólica é também única para cada tipo de CPU, de forma que um programa escrito em linguagem simbólica para uma CPU poderá não ser executado em outra CPU de uma família diferente.

- Nos primórdios da programação os programas eram escritos nessa linguagem.

Linguagem simbólica (Assembly):

- Hoje a linguagem simbólica, é utilizada quando a velocidade de execução ou o tamanho do programa executável gerado são essenciais. A conversão da linguagem simbólica para a linguagem de máquina se chama montagem, e é feita por um programa chamado montador (ou assembler).

- Uma típica instrução em linguagem simbólica seria:

ADD A, B

- Essa linguagem é também classificada como linguagem de segunda geração, e, assim como a linguagem de máquina, é considerada uma linguagem de baixo nível.

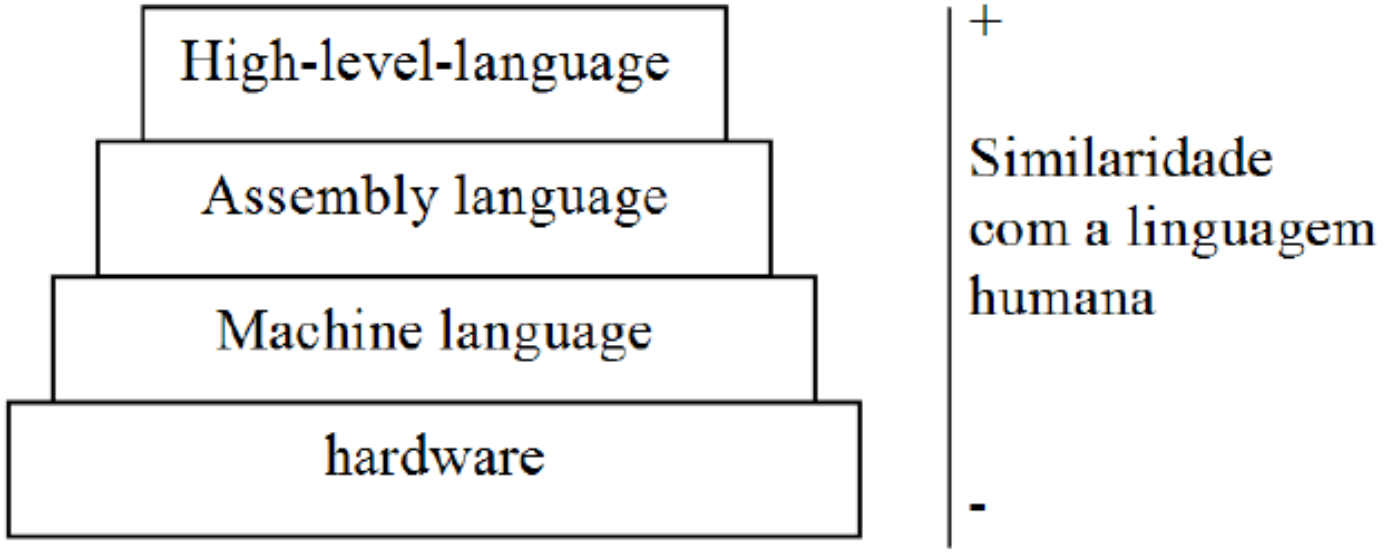
Linguagem de alto nível:

- São as linguagens de programação que possuem uma estrutura e palavras-chave que são mais próximas da linguagem humana. Tornando os programas mais fáceis de serem lidos e escritos. Esta é a sua principal vantagem sobre as linguagens de nível mais baixo.
- Os programas escritos nessas linguagens são convertidos para a linguagem de baixo nível através de um programa denominado compilador ou de um interpretador.
- Uma instrução típica de uma linguagem de alto nível é:

if (A>10) then A:=A-7;

Exemplo: C, Pascal, Fortran, etc

Linguagens de Programação



Breve histórico de “C”

- Criada por Dennis Ritchie;
- Em 1972;
- Centro de Pesquisas da Bell Laboratories;
- Para utilização no S.O. UNIX;
- O C é uma linguagem de propósito geral.

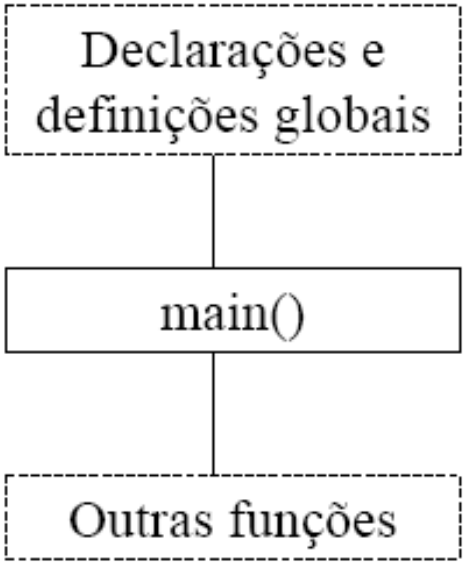
Características básicas da linguagem

- Sensível ao caso (Case sensitive);
- Tipos de dados primitivos: caractere, inteiro e real;
- Possui estruturas de controle de fluxo para viabilizar a programação estruturada;
- Operadores aritméticos, lógicos, relacionais e condicional;
- Todo programa tem uma função principal chamada main();
- Todo linha de instrução em um programa é finalizada com um “;”.

ANSI

Devido à falta de padronização da linguagem C, em 1983, o **Instituto Norte-Americano de Padrões** (**ANSI**) formou um comitê, X3j11, para estabelecer uma especificação do padrão da linguagem C. O padrão foi completo em 1989 e ratificado como ANSI X3.159-1989 “Programming Language C” (**C ANSI**).

Estrutura de um programa em C



Obrigatório
Opcional

Conceitos Básicos – Linguagem C

Exemplo de um programa

```
// Exemplo de programa em C
// Isto é uma linha de comentário

void main()
{
    int a;          // declara a variável "a"
    a = 3 + 2;      // soma 3 com 2
}
```

- Um programa em C é composto por um conjunto de **Funções**. A função pela qual o programa começa a ser executado chama-se ***main***.
- Um programa em C deve ser **Identado** para que possa ser lido com mais facilidade.
- Após cada cada comando em C deve-se colocar um ; (ponto-e-vírgula).

Identificadores

- São os nomes que podem ser dados para variáveis e funções.
- Para a escolha destes nomes é necessário seguir algumas regras:
  - Um identificador deve iniciar por uma letra ou por um "_" (*underscore*);
  - A partir do segundo caractere pode conter letras, números e *underscore*;
  - Deve-se usar nomes significativos dentro do contexto do programa;
  - C é uma linguagem *case-sensitive*, ou seja, faz diferença entre nomes com letras maiúsculas e nomes com letras minúsculas. Peso e peso são diferentes;
  - Costuma-se usar maiúsculas e minúsculas para separar palavras: "PesoDoCarro";
  - Deve ser diferente dos comandos da linguagem;
  - Deve ter no máximo 31 caracteres (no caso do TurboC);
  - Pode conter números a partir do segundo caractere;

Exemplos:
Idade, Contador, PesoDoCarro, Usuario_1, CorDaPagina, RaioDoCirculo

Variáveis

- Uma variável é uma posição de memória que pode ser identificada através de um nome.
- Podem ter seu conteúdo alterado por um ***comando de atribuição***.
- Após a atribuição mudam de valor.

```
int a,b, SomaGeral;
a = 3;                  // a recebe o valor 3
b = a * 2;              // b recebe o dobro do valor de a
c = a + b + 2;          // c recebe 11
```

Tipos de Variáveis

- Todas as variáveis em C tem um tipo;
- Cada tipo define os valores que a variável pode armazenar;
- Cada tipo ocupa uma certa quantidade de memória.

Tipo	Tamanho	Valores Válidos
char	1 byte	letras e símbolos: 'a', 'b', 'H', '^', '*','1','0'
int	2 bytes	de -32767 até 32767 (apenas números inteiros)
float	4 bytes	de -3.4 x 1038 até +3.4 x 10+38 com até 6 dígitos de precisão
double	8 bytes	de -1.7 x 10308 até +1.7 x 10+308 com até 10 dígitos de precisão

Declaração de Variáveis

- Todas as variáveis tem que ser declaradas antes de serem usadas;
- Não há uma inicialização implícita na declaração

```
// Exemplo de programa em C

#include <stdio.h>                          // Arquivo de cabeçalho (header)
void main()
{
      int contador; // declarações simples
      float PrecoDoQuilo;
      double TaxaDeCambio;
       char LetraDigitada;
       int IdadeManoel, IdadeJoao, IdadeMaria;
                                  // Pode colocar mais de uma variável na // na mesma linha
      Double   TaxaDoDolar,
                  TaxaDoMarco,
                  TaxaDoPeso,              // Também pode trocar de linha no meio
                  TaxaDoFranco;
 ..... }
```

Inicialização de Variáveis na Declaração

```
// Exemplo de programa em C

#include <stdio.h>                          // Arquivo de cabeçalho (header)
void main()
{
    int NroDeHoras = 0;                     // declara e inicializa com Zero
    float PrecoDoQuilo = 10.53;             // declara e inicializa com 10.53
    double TaxaDoDolar = 1.8,
           TaxaDoMarco = 1.956,
           TaxaDoPeso = 1.75,
           TaxaDoFranco = 0.2;
.......
}
```

Constantes

- Constantes são identificadores que não podem ter seus valores alterados durante a execução do programa.
- Para criar uma constante existe o comando #define que, em geral é colocado no início do programa-fonte.

Exemplos:

```
#define LARGURA_MAXIMA 50                    // Não se coloca ponto-e-vírgula após o valor
#define NRO_DE_DIAS_DA_SEMANA 7
#define NRO_DE_HORAS_DO_DIA 24
#define VALOR_DE_PI 3.1415
void main ()
{
        int TotalDeHoras;
        TotalDeHoras = 10 * NRO_DE_DIAS_DA_SEMANA * NRO_DE_HORAS_DO_DIA;
.....
}
```

Strings

Uma String é uma seqüência de caracteres entre aspas duplas:

"exemplo de uma string em C".

A função **printf** exibe um ou mais dados na tela. Para tanto ele deve receber pelo menos dois parâmetros, separados por vírgula.

Por exemplo:
printf("%s","teste");

"%s" : é a string de formato
"teste" : é o dado a ser impresso.

A ***string de formato*** define quais os tipos dos dados a serem impressos. O símbolo **%s** será substituído pelo dado que vem após a vírgula.
Os **dados** definem quais os valores a serem impressos.

Strings

- Se for necessário, um string de formato pode definir que mais de um dado será impresso. Para tanto, dentro da string de formato deve haver mais de um **%**, um para cada dado a ser impresso.
  Neste caso, os dados devem vir após a string de formato separados por vírgulas.

Por exemplo:

printf("%s %s","teste1", "outra string");

Isto irá imprimir o string **teste1** deixar 1 espaço em branco e imprimir ao lado o string **outra string**, assim :

teste1 outra string

Exemplo

```
#include <stdio.h>      // Necessário para usar a função printf
                        // A função printf exibe um ou mais dados na tela

void main ()

{
    printf("%s","Isto é uma string ....\n");     // note o '\n' no final da string;
    printf("%s","Outra string ....");
    printf("%s","Terceira string\n");

}
```