Programação Estruturada Aula 1 - Introdução

Prof. Luis Carlos Retondaro

Técnico em Telecomunicações
2^o Ano

CEFET/RJ - Centro Federal de Educação Tecnológica Celso Suckow da Fonseca

Campus Petrópolis

2017

Sumário

O que vamos aprender

- Programação em C;
- História da linguagem;
- Introdução à linguagem C;
  - Variáveis, constantes, expressões aritméticas e lógicas;
  - Tipos de dados, E/S de dados;
  - Operadores;
- Comandos de controle: seleção, repetição;

O que vamos aprender

- Funções;
- Matrizes e strings;
- Estruturas;
- Ponteiros;
- Alocação dinâmica;
- Arquivos.

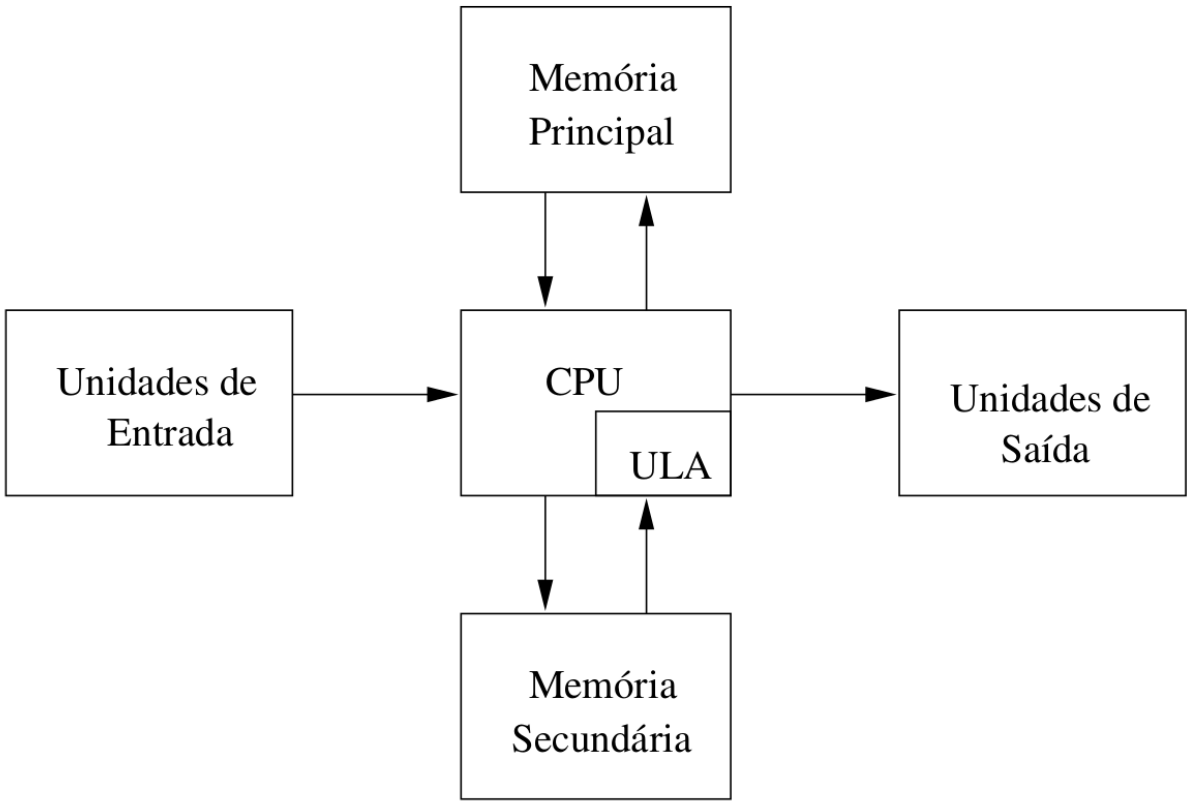
Organização básica de um computador

- Um computador é constituído de 4 unidades básicas: entrada, saída, processamento central e memória;
- **Entrada:** A unidade de entrada são os dispositivos que permitem que o usuário interaja com o computador, fornecendo dados que serão processados;
- **Saída:** São dispositivos que fornecem ao usuário os resultados do processamento realizado;

Organização básica de um computador

- **CPU:** É responsável por todo processamento requerido;
  - **ULA:** É a unidade lógica e aritmética, responsável pelos cálculos matemáticos;
- **Memória:** Armazena dados e informações que serão utilizados no processamento;
  - **Memória Principal:** Usada pela CPU para armazenar instruções e informações enquanto o computador está ligado (memória RAM);
  - **Memória Secundária:** Usada pelo computador para armazenar instruções e informações por tempo indeterminado independente do estado do computador, com capacidade maior e acesso mais lento que a RAM.

Organização básica de um computador



Linguagem de máquina

- As unidades que compõem o computador precisam se comunicar umas com as outras;
- **Ex.:** Dado fornecido pelo teclado é armazenado na memória. Quando a CPU vai realizar uma operação aritmética ela busca os valores armazenados na memória, etc;
- Para que ocorra essa comunicação é necessário estabelecer uma linguagem;
- Sere humanos: linguagem escrita e a fala (cada qual com suas regras);
- **Escrita:** parágrafos, períodos, frases, palavras, letras.

Linguagem de máquina

- Uma letra é um ente indivisível da linguagem escrita e, por isto, é chamada símbolo básico desta linguagem;
- Toda linguagem requer a existência de símbolos básicos, para a fala os fonemas;
- A linguagem entre as unidades deve ser um fenômeno físico e possuir dois símbolos básicos dada a facilidade de representar fisicamente dois estados distintos e não confundíveis.

Linguagem de máquina

- A linguagem usada para a comunicação interna de um computador é chamada de linguagem de máquina e possui dois símbolos;
- Cada um desses símbolos é denominado de bit (**b**inary digi**t**), representados por 0 e 1;
- bit em inglês significa fragmento, dessa forma as palavras na linguagem de máquina são sequências de bits, ou seja sequências de 0 e 1.

O código ASCII

- Para que seja possível a comunicação entre homem e computador, é necessário que as palavras da linguagem escrita sejam traduzidas para a linguagem de máquina e vice-versa;
- Então, é necessário estabelecer qual sequência de bits que corresponde a cada caractere usado na linguagem escrita, ou seja, uma codificação em bits para cada um dos caracteres;
- Uma codificação muito utilizada é o código ASCII (*American Standard Code for Information Interchange*, estabelecido pelo ANSI (American National Standards Institute);
- Nesta codificação, cada caractere é representado por uma sequência de oito bits (byte);

O código ASCII

Tabela 1 Códigos ASCII de alguns caracteres

Caractere	Código ASCII
Espaço em branco	00100000
!	00100001
"	00100010
. . .	. . .
0	00110000
1	00110001
. . .	. . .
A	01000001
B	01000010
. . .	. . .
Z	01011010
. . .	. . .
a	01100001
. . .	. ..

O código ASCII

- Cada sequência de 0 e 1 pode ser vista como a representação de um número inteiro no sistema binário de numeração [Evaristo, J 2002];
- Pode-se, até para facilitar a manipulação, associar a cada código ASCII o inteiro correspondente (código ASCII decimal);
- **Exemplo:** 1000001 é a representação do número (decimal) 65 no sistema binário de numeração, dizemos que o código ASCII decimal de A é 65.

Alguns termos técnicos

- **Dados:** Qualquer tipo de informação ou instrução que pode ser manipulada pelo computador;
- **Bit:** Unidade básica para armazenamento, processamento e comunicação de dados;
- **Byte:** Um conjunto de 8 bits;
- **Palavra:** Um conjunto de n bytes (e.g. $n = 1, 2, 4, 8$). Os dados são organizados em palavras de n bytes. Os processadores mais modernos processam os dados em palavras de 16 bytes ou 128 bits;
- **Comandos:** São as instruções que fazem com que o computador execute tarefas.

Alguns termos técnicos

- **Programa:** É uma sequência de instruções com alguma finalidade;
- **Arquivo:** Conjunto de bytes que contém dados (programa, imagem, lista de nomes, ...);
- **Software:** É um conjunto de programas com um propósito global comum;
- **Hardware:** Parte física do computador;
- **Sistema Operacional:** Conjunto de programas que gerenciam e alocam recursos de hardware e Software.

Alguns termos técnicos

- **Linguagem de programação:** Consiste da sintaxe (gramática) e semântica (significado) utilizada para escrever (codificar) um programa;
  - **Alto Nível:** Linguagem de codificação de programa independente do tipo de máquina e, de fácil utilização pelo ser humano (Pascal ,C, Java, Python, C++, ...);
  - **Baixo Nível:** Linguagem de codificação baseada em mnemônicos depende do tipo da máquina e é de fácil tradução para a máquina (Assembly);
- **Linguagem de Máquina:** Conjunto de códigos binários que são compreendidos pela CPU de um computador (depende do tipo da máquina).

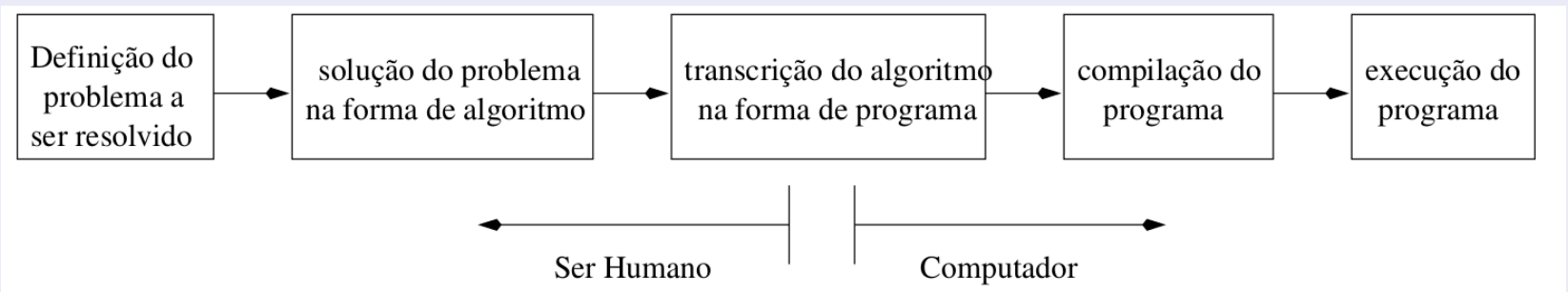
Alguns termos técnicos

- **Compilador:** Traduz programas codificados em linguagem de alto nível e baixo nível (código fonte) para linguagem de máquina (código executável);
- **Interpretador:** Traduz o código fonte para código de máquina diretamente em tempo de execução (Basic, Python, Lisp);
- **Algoritmos:** São procedimentos ou instruções escritos em linguagem humana antes de serem codificados usando uma linguagem de programação.

Alguns termos técnicos

Objetivo do curso

- Etapas de solução de um problema utilizando computador.



Programas de computador

- Para que um computador tenha alguma utilidade, ele deve executar um programa que tenha uma finalidade específica;
- **Exemplos:**
  - **Games:** são programas com o objetivo propiciar entretenimento aos seus usuários;
  - **Processadores de texto:** são programas que permitem que textos sejam digitados, impressos e armazenados;
  - **Planilhas eletrônicas:** são programas que oferecem recursos para manipulação de tabelas de valores numéricos;
  - **Navegadores:** permitem acesso à páginas da internet.

Programas de computador

- Estes programas são para usuários finais, pessoas que vão utilizar o computador, usando um programa que aprendeu a usar, sem preocupação relativa com o funcionamento interno do sistema (computador/programa);
- **Exemplo:** um usuário de um processador de texto deve aprender os passos para que o processador destaque em negrito alguma parte do texto ou localize uma palavra, sem a necessidade de saber como o programa realiza estas ações;
- De fato para que um processador de texto execute qualquer tarefa é necessária a execução de muitas instruções com objetivos bem mais específicos.

Programas de computador

- Um programa de computador é um conjunto de instruções que podem ser executadas pelo computador;
- A área da computação que trata do desenvolvimento de programas "menores" é a programação de computadores.

Lógica de programação

- O desenvolvimento de um programa requer a utilização de um raciocínio ímpar em relação aos raciocínios utilizados na solução de problemas de outros campos do saber;
- **Exemplo simplificado:**
- **Problema de Mecânica Newtoniana:** para tentar resolver deve-se procurar capturar a especificação, as grandezas físicas envolvidas e aplicar as fórmulas que relacionam estas grandezas;
- **Por programação:** para se desenvolver um programa que resolva um determinado problema é necessário que encontremos uma sequência de instruções que cujas execuções resultem na solução da questão (algoritmo).

Lógica de programação

- A lógica de programação pode ser entendida como o conjunto de raciocínios utilizados para o desenvolvimento de algoritmos (e, portanto, de programas).

Exemplo:

Um senhor, está numa das margens de um rio com uma raposa, uma dúzia de galinhas e um saco de milho. O senhor pretende atravessar o rio com suas cargas, num barco a remo que só comporta o senhor e uma das cargas.

Evidentemente, o senhor não pode deixar em uma das margens, sozinhos, a raposa e a galinha, nem a galinha e o milho.

A questão é escrever um algoritmo que oriente o senhor a realizar o seu intento.

Lógica de programação

Algoritmo de travessia

1. Atravesse as galinhas.
2. Retorne sozinho.
3. Atravesse a raposa.
4. Retorne com as galinhas.
5. Atravesse o milho.
6. Retorne sozinho.
7. Atravesse as galinhas.

O que é sintaxe?

- Um programa escrito em linguagem de alto nível é traduzido para a linguagem de máquina por um compilador;
- Para que o compilador consiga fazer a tradução, é necessário que cada instrução seja escrita de acordo com regras preestabelecidas;
- Estas regras são chamadas sintaxe da linguagem e quando não são obedecidas dizemos que existe erro de sintaxe no programa;
- Se o programa fonte contém algum erro de sintaxe o compilador não consegue compilar o programa e indica qual o tipo de erro cometido.

O que é semântica?

- Cada instrução tem uma finalidade específica. Ou seja, a execução de um instrução resulta na realização de alguma ação parcial;
- É a sequência das ações parciais que redunda na realização da tarefa para a qual o programa foi escrito;
- A ação resultante da execução de uma instrução é chamada semântica da instrução;
- Infelizmente, um programa pode não conter erros de sintaxe (e, portanto,pode ser executado), mas a sua execução não fornecer como saída o resultado esperado para alguma entrada.

O que é semântica?

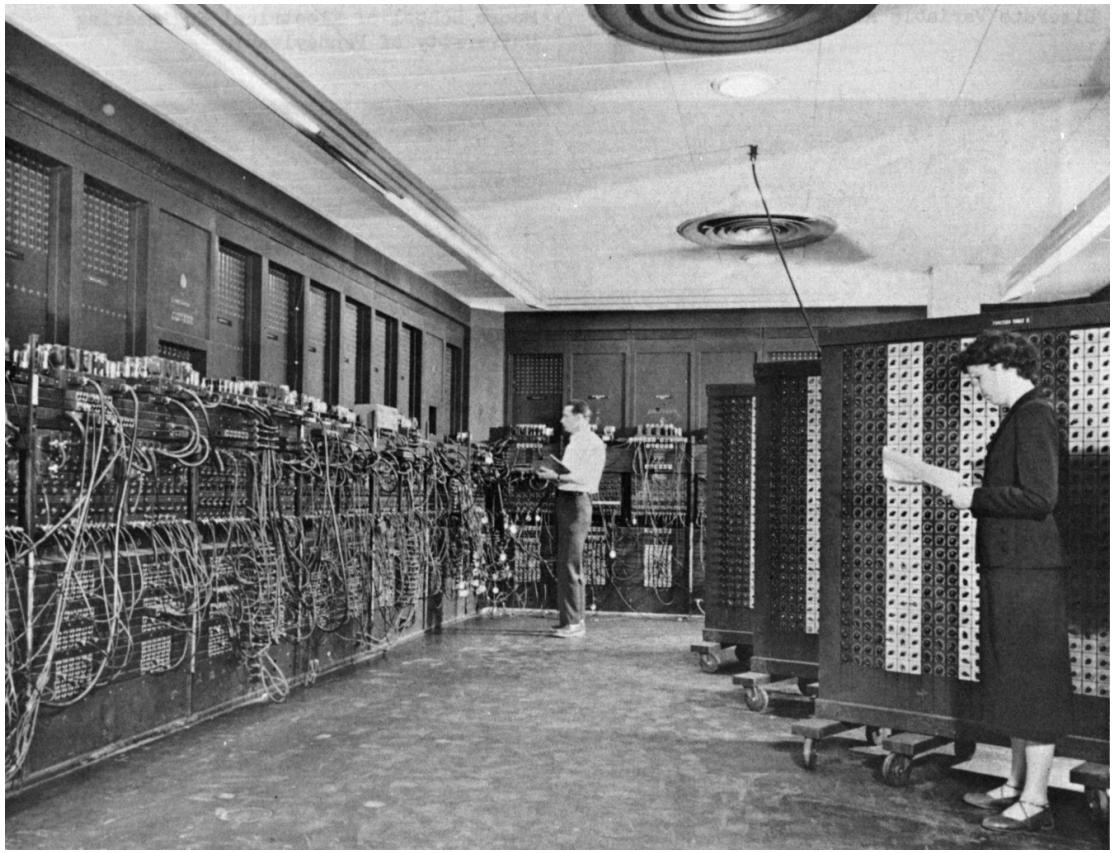
- Para aprender a programar numa determinada linguagem é necessário que se aprenda a sintaxe de cada instrução e suas semânticas;
- Também ter uma boa lógica de programação para que se escolha as instruções necessárias e a sequência segundo a qual estas instruções devem ser escritas;
- Felizmente ou infelizmente, para cada tarefa que se pretende não existe apenas uma sequência de instruções que a realize (não existe apenas um programa);
- Dado um problema, devemos procurar o melhor programa, ou seja, um programa que tenha boa legibilidade, demande o menor tempo possível e que use o mínimo da memória.

O que vamos aprender

- Criar algoritmos simples;
- Usar a linguagem C para descrever os algoritmos;
- Testar o programa criado;
- Ter um executável para realizar as tarefas especificadas.

Um pouco de história

- **Primórdios da programação:** programação em código binário;
- ENIAC:



Um pouco de história

- **Melhoria na programação:** Linguagem Assembly;
- Criou-se uma linguagem de baixo nível, para representar as instruções em código binário;
- Um programa chamado montador ou Assembler, faz a transformação em código de máquina;
- **Exemplo:**

$$\begin{aligned} LOOP: \quad & MOV\ A, 3 \\ & INC\ A \\ & JMP\ LOOP \end{aligned}$$

Um pouco de história

- Criação de linguagens de alto nível e compiladores para estas;
- Mais distantes da máquina e mais próximas de linguagens naturais;
- Um compilador as transforma em código executável;
- **Exemplo:**
  - C, $C++$, ...
  - Pascal
  - Java
  - Python

História do C

- A primeira versão de C foi criada por Dennis Ritchie em 1972 nos laboratórios Bell;
- O objetivo era incluir a linguagem como um dos softwares a serem distribuídos juntamente com o sistema operacional Unix do computador PDP-11;
- O foco da linguagem C inicialmente foi o desenvolvimento de sistemas operacionais e compiladores.

História do C

- O surgimento do C iniciou com a linguagem ALGOL 60, definida em 1960. ALGOL era uma linguagem de alto nível;
- A linguagem ALGOL permitia ao programador trabalhar "longe da máquina", sem se preocupar com os aspectos de como cada comando ou dado era armazenado ou processado;
- Criado para substituir o FORTRAN, a ALGOL não teve sucesso, (muito alto nível) em uma época em que a maioria dos SOs exigiam do usuário grande conhecimento de hardware.

História do C

- Em 1967 surgiu CPL (*Combined Programming Language*) nas universidades de Londres e Cambridge com o objetivo, de "trazer ALGOL à terra", ou "manter contato com a realidade de um computador real";

- Da mesma forma de ALGOL, CPL não foi bem aceita, em especial pelos projetistas de sistemas operacionais que a consideravam difícil de implementar.

História do C

- Em 1970, Ken Thompson, chefe da equipe que projetou o UNIX para o PDP-11 (Bell Labs) implementou um compilador para uma versão mais reduzida do CPL. Batizou a linguagem de B;
- A linguagem B mostravam-se muito limitada, podendo ser utiizada apenas para certas classes de problemas;
- O grupo responsável por reescrever o UNIX em uma linguagem de alto nível percebeu que a linguagem B era considerada lenta.

História do C

- Estes problemas levaram o projetista Dennis Ritchie, do Bell Labs, a ser encarregado de projetar uma nova linguagem, sucessora do B, que viria então, a ser chamada de C;
- A linguagem C buscou manter o "contato com o computador real" e ainda sim dar ao programador novas condições para o desenvolvimento de programas em áreas diversas, como comercial, científica e de engenharia;
- C foi usada com grande êxito na construção de uma nova versão do sistema operacional Unix, que inicialmente foi escrito em Assembly.

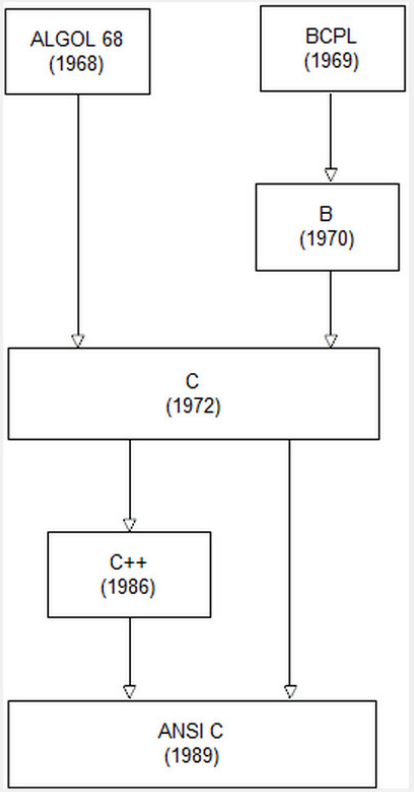
História do C

- O grande sucesso obtido no mundo do Unix fez com que a linguagem ganhasse cada vez mais adeptos;
- Atualmente, quase todos os grandes sistemas operacionais são construídos em C/C++.

Padronização do C

- No ano de 1978 foi publicado o livro: *The C Programming Language* por Kernigham e Ritchie. Este livro fez grande sucesso e ajudou muito a divulgar a linguagem;
- Nos anos 80 C passou a ser reconhecida como uma linguagem de propósito geral e contava com diversos compiladores desenvolvidos por vários fabricantes;
- Com uma série de compiladores C semelhantes, eles frequentemente apresentavam várias discrepâncias e eram incompatíveis entre si, tornando a padronização da linguagem uma necessidade;
- A padronização foi iniciada pela ANSI – *American National Standard Institute* – em 1983 e finalizada em 1989.

Origem do C



Origem e evolução da Linguagem C

Padronização do C

- Atualmente ainda há versões de compiladores próprias de cada fabricante, porém a maioria dos fabricantes oferece uma opção de compatibilidade com o padrão ANSI C;
- A linguagem C é uma linguagem de propósito geral, o que quer dizer que se adapta a praticamente qualquer tipo de projeto, altamente portável e extremamente rápida em tempo de execução;
- Linguagens como Java, C++ e C# foram influenciadas pela linguagem C.

Linguagem de médio nível

- Um tipo de dados define os valores que uma variável pode armazenar e as operações que podem ser executadas com essa variável;
- Embora C tenha 5 tipos de dados internos, ela não é uma linguagem rica em tipos de dados como Pascal e Ada;
- Permite quase todas as conversões de tipos de dados.

Linguagem de médio nível

- Sendo uma linguagem de nível médio alguns detalhes de programação ficam sob a responsabilidade do programador (limites de matrizes, utilização de ponteiros, alocação e liberação de memória);
- O C tem apenas 32 palavras-chaves, que são os comandos que compõem a linguagem e as linguagens de alto nível possuem algumas vezes mais essa quantidade de palavras reservadas;
- Ex.: A maioria das versões do Basic possuem bem mais de 100 palavras-chaves.

Linguagem estruturada

- Embora o termo de linguagem estruturada em blocos não seja rigorosamente aplicável ao C, ela é normalmente referida como linguagem estruturada;
- **Razão:** As linguagens estruturadas em blocos permitem que procedimentos e funções sejam declarados dentro de procedimentos e funções;
- A característica principal de uma linguagem estruturada é a compartimentalização do código e dos dados (uso de sub-rotinas);
- Com o uso de variáveis locais é possível escrever sub-rotinas de forma que os eventos que ocorrem dentro delas não causem nenhum evento inesperado nas outras partes do programa.

Linguagem estruturada

- Uma linguagem estruturada permite muitas possibilidades na programação, suporta diretamente diversas construções de laços;
- Permite que o programador insira sentenças em qualquer lugar de uma linha;
- Exemplo de linguagens estruturadas: Pascal, Ada, C, Modula2, C++, Java;
- Exemplo de linguagens não estruturadas: Fortran, basic, Cobol.

Linguagem estruturada

- Linguagens estruturadas tendem a ser modernas;
- O principal componente estrutural do C é a função;
- As funções em C são blocos em que toda atividade do programa ocorre;
- As funções permitem o programador codificar separadamente as diferentes tarefas de um programa (modularização do programa);
- Outra forma de estruturar o código é através de blocos de código, grupos de comandos conectados logicamente (comandos condicionais, de repetição).

Referências

1. C completo e Total. 3a ed. Revista e Atualizada, Hebert schildt, Makron Books Ltda.
2. Linguagem C. DAMAS, Luis. 10a. Edição. LTC, 2014.
3. Aprendendo a programar, programando na linguagem C. 3a. Edição. Edição digital, Jaime Evaristo.
4. Notas de aula prof. Alexandre Xavier Falcão, UNICAMP, 2005.