

Fundamentos de Programação

Curso de Gestão da Tecnologia da Informação

Professora: Esp. Sibele Mueller

E-mail: sibele.gti@seifai.edu.br

Compilador X Interpretador

- Um programa é a maneira de se comunicar com um computador e a única linguagem que o computador entende é a linguagem de máquina.
- Todos os programas que se comunicam com o computador devem estar em linguagem de máquina.
- O computador deve converter os comandos dados em *linguagem de alto nível* para *linguagem de máquina* (códigos binários). A tradução é feita por um programa chamado compilador.

Compilador X Interpretador

- A forma como os programas são traduzidos da linguagem de alto nível para a linguagem de máquina (baixo nível) é classificada em duas categorias:
 - **interpretados e;**
 - **compilados.**

Interpretador

- Lê a primeira instrução do programa, faz a consistência de sua sintaxe (vê se está escrito corretamente) e se não houver erro converte-a para linguagem de máquina e depois ordena ao computador para que a execute;
- Depois repete o processo seguindo para a próxima instrução, repetindo o processo até a execução da última instrução.
- No momento em que a segunda instrução é executada, a primeira é perdida, pois apenas uma instrução fica na memória em casa instantânea;
 - Se o mesmo programa for executado uma segunda vez, haverá uma nova tradução, comando por comando, pois os comandos não ficam armazenados para futuras execuções.
- O interpretador precisa estar presente todas as vezes que é executado o programa, portanto o trabalho de checagem da sintaxe e tradução é repetitivo.

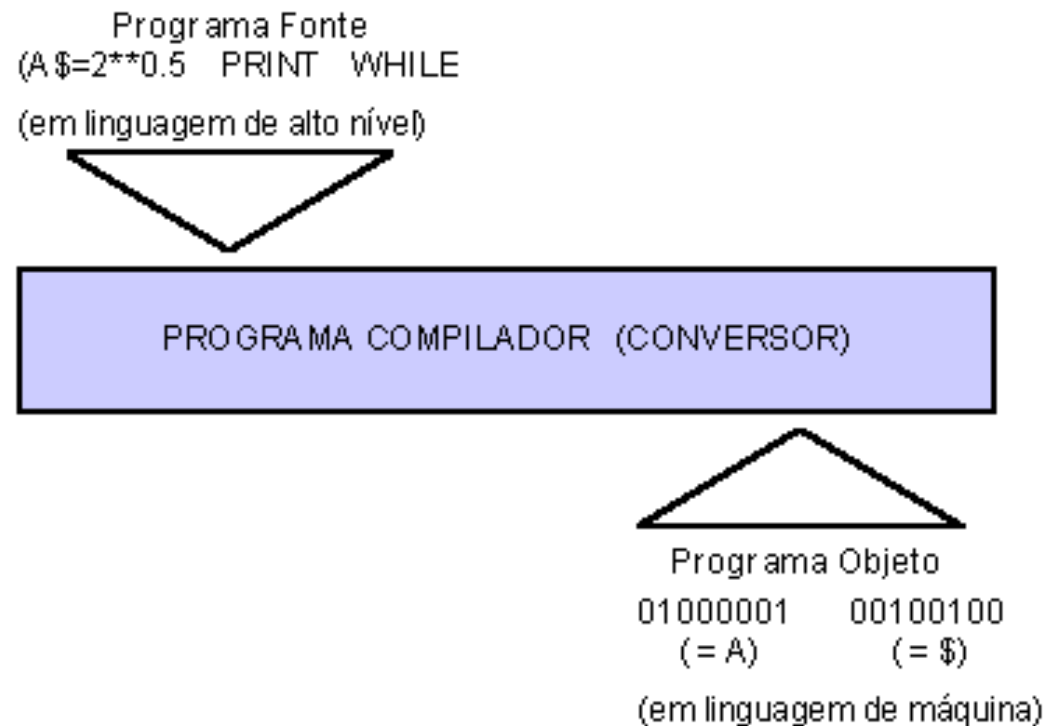
Compilador

- Quando programamos em uma linguagem de programação de alto nível primeiramente criamos um arquivo de texto comum contendo a lógica do programa, ou seja, é onde falamos ao computador como deve ser feito o que queremos. Este arquivo de texto é chamado de **código-fonte**.
- O arquivo de código-fonte deve ser traduzido para linguagem binária usando o compilador correspondente com a linguagem na qual estamos programando.
- É gerando um arquivo com o sufixo .OBJ com as instruções já traduzidas para linguagem de máquina.

Compilador

- O compilador irá gerar um segundo arquivo que chamamos de executável ou programa, este arquivo gerado é interpretado diretamente pelo computador, isto é feito pelo linkeditor, que além de juntar as rotinas cria o arquivo executável.
 - Ao programa original, em linguagem de alto nível, dá-se o nome de *Programa Fonte (código-fonte)* e ao resultado, em linguagem de máquina, de *Programa Objeto* (executável).
- Existem algumas linguagens de programação que não necessitam de compiladores, como o PHP, uma linguagem dedicada à produção de websites dinâmicos. As instruções em PHP são compiladas e executadas ao mesmo tempo.

Compilador



- Fonte: <https://www.inf.ufsc.br/~j.barreto/cca/arquitet/arq3.htm>

Comparativo: Compilador X Interpretador

- Velocidade de execução do programa compilado chega a ser 20 vezes mais rápida do que o programa interpretado;
- O programa compilado e linkeditor pode ser executado diretamente sobre o sistema operacional não necessitando de nenhum outro software;
- Programas .exe não podem ser alterados, o que protege o código fonte.

Programação



- *“Tudo o que há dentro de um computador, além das funções que ele realiza, foi criado pelo homem. Pode, portanto, ser explicado e entendido.”*
- *“A única forma de aprender uma linguagem de programação ou uma nova linguagem de programação é escrever programas nesta linguagem”*

Importante!

- Não existe receita pronta!
- Programar é dedicação, persistência, errar, acertar, mas o mais importante é não desistir!
- É necessário conhecer os comandos da linguagem.



Linguagem C

- A primeira versão de C :
 - criada por Dennis Ritchie em 1972;
- A linguagem C é o resultado da linguagem BCPL;
- BCPL influenciou a linguagem B;
- A linguagem C é a evolução da linguagem B.



Linguagem C

- Necessidades do surgimento da linguagem C:
 - prover acesso de baixo nível ao hardware;
 - se entender bem com o Assembly;
 - ter um bom desempenho e otimizar o uso de memória;
 - ser de fácil compreensão.



Linguagem C

- Limite da linguagem C :
 - tamanho de um projeto ultrapassa 25.000 a 100.000 linhas de código.
- Solução:
 - Desenvolvimento de nova linguagem(1980): C++
 - Linguagem C++ evoluiu (suporta Programação Orientada a Objetos).
- Linguagens como Java e C# foram influenciadas pela linguagem C.
- Grandes sistemas operacionais são construídos em C/C++.

Linguagem C

- A linguagem C :
 - se adapta a praticamente qualquer tipo de projeto;
 - altamente portátil (diferentes SO);
 - extremamente rápida em tempo de execução;



Instalação Code::Blocks

- <http://www.planetaunix.com.br/2014/10/instalando-o-codeblocks-no-linux-ubuntu.html>
- Ir no konsole e digitar os seguintes comandos:
 - **sudo apt-get update**
 - **sudo apt-get install g++**
 - **sudo apt-get install codeblocks**
- Ir na pasta Pessoal e na pasta 'Documentos'. Dentro de Documentos criar a pasta 'C'.

Primeiro Programa em C

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

```
int main ()
```

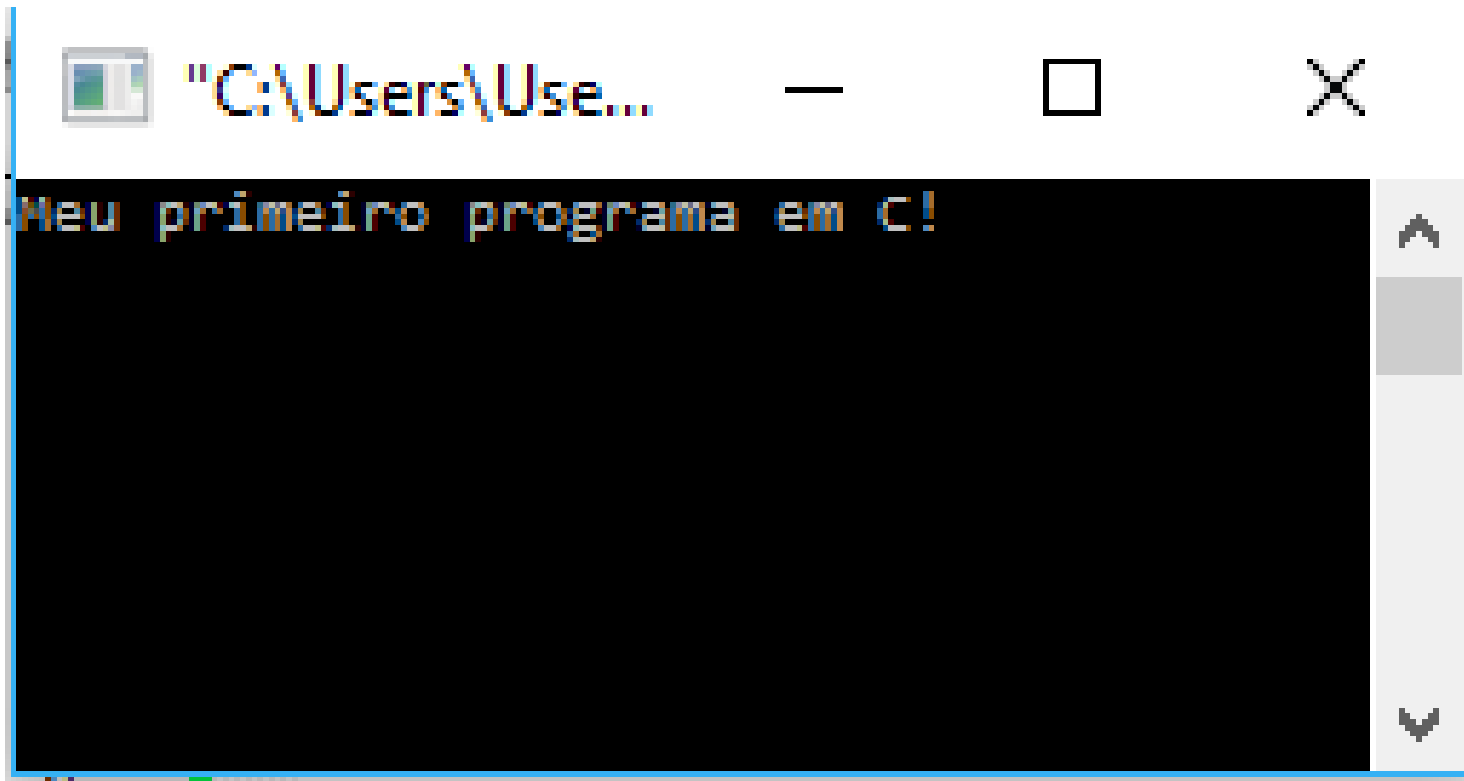
```
{
```

```
    printf("Meu primeiro programa em C!");
```

```
    getchar();
```

```
}
```


Resultado



A screenshot of a Windows command prompt window. The title bar shows the file path "C:\Users\Use...". The command prompt displays the output of a C program: "Neu primeiro programa em C!". The text is in a monospaced font, and the background is black.

Primeiro Exemplo

- `#include <stdio.h>`
- `//diz ao compilador para incluir informações sobre a biblioteca padrão de entrada/saída`
- `main ()` //função principal (primeira função a ser executada). É por ela que se inicia a execução do programa
- `{`//início da função main
- `printf("Meu primeiro programa em C!");`
- `//função de biblioteca que imprime uma saída, definida como argumento desta função`
- `getchar();` //função que aguarda a captura do teclado usado sempre no final do programa para que o mesmo possa ser visualizado.
- `}` //término da função main

Comentários

- Podem aparecer em qualquer lugar do programa e são desconsiderados pelo compilador.
- A linguagem C aceita os seguintes tipos de comentários:

`/*comentário de bloco*/`

`//comentário de linha`

- Não é permitido utilizar um comentário dentro do outro.

Bibliotecas

- São os arquivos de cabeçalho que possibilitam o uso das funções em C. Vejamos outras bibliotecas:
 - ***#include <stdlib.h>*** // responsável pela conversão de strings para números, gerenciamento de alocação dinâmica na memória e o uso da função *system()* ;
 - ***#include <ctype.h>*** // classificação e transformação de caracteres
 - ***#include <math.h>*** // uso de operações matemáticas
 - ***#include <string.h>*** // manipulação de strings
 - ***#include <time.h>*** // manipulação de datas e horas
 - ***#include <conio.h>*** // necessário para as funções *clrscr* e *getch*

Comandos / Funções

- Principais comandos ou funções usadas para entrada e saída de informação em programas C/C++
- Exibir algo na tela (saída):
 - `printf(“expressão de controle”, argumentos);`
 - Ex: `printf(“este é o número: %d”, 2);`

Comandos / Funções

- Caracter especial usado com printf:
 - \n - nova linha
- Códigos para impressão formatada com printf.
 - **%c – caracter simples** // Símbolos da tabela ASCII ou numero inteiro entre -128 e +127
 - **%s – cadeia de caracteres** // Símbolos da tabela ASCII ou numero inteiro entre -128 e +127
 - **%d – decimal (inteiros)** // -2.147.483.648 a 2.147.483.648
 - **%f – ponto flutuante** // Números reais com 6 dígitos de precisão

Comandos / Funções

- Receber dados do usuário (entrada):
 - scanf(“expressão de controle”, argumentos);
 - Ex: scanf(“%d”, &num);
 - Onde ‘%d’ indica que deve ser lido um número inteiro decimal. O operador “&” é o operador de endereço indicando logo após ele em qual variável a informação deverá ser armazenada, neste caso em num que é o nome da variável.
 - Após o uso do comando scanf é recomendado usar o comando fflush(stdin); O qual é usado para limpar o buffer do teclado

Comandos / Funções

- Códigos para a leitura formatada com scanf.
 - %c – caracter simples
 - %s – cadeia de caracteres
 - %d – decimal (inteiros)
 - %f – ponto flutuante

Declaração de Variáveis / Tipos de Dados

- Toda variável deve ser declarada e ao ser declarada deve-se atribuir um tipo.
- O tipo determina como valores de dados são representados, que valores pode expressar, e que tipo de operações se pode executar com estas variáveis. Os principais tipos aceitos pela linguagem são:
 - char = character;
 - int = números inteiros;
 - float = ponto flutuante em precisão simples;
 - double = ponto flutuante em dupla precisão;

Constantes

- Tudo que é invariável.
- Valores que permanecem sem alteração durante toda a vida do programa.
- Exemplo:
 - `#define PI 3.1415`
 - `const float teste=12.5;`

Operadores

- Atribuição
 - É feita pelo = (Sinal de igual)
 - Sintaxe: Operador = operando;
 - Semântica: Operador RECEBE valor de operando
- Aritméticos
 - Multiplicação *
 - Divisão /
 - Adição +
 - Subtração –
 - Resto da divisão inteira %

Operadores

- Relacionais

- Maior que >
- Maior ou igual a >=
- Menor que <
- Menor ou igual a <=
- Igual ==
- Não igual !=

- Lógicos

- E (and) &&
- Ou (or) ||
- Negação (not) !

Operadores

- Incremento
 - Incrementa antes ++variável
 - Incrementa após variável++
- Decremento
 - Decrementa antes --variável
 - Decrementa após variável--

Comando para Limpar a tela

- `clrscr()`

Regras Básicas

- Usar SEMPRE comentários em seu código fonte;
- Usar INDENTAÇÃO de seus fontes. (padrão min 3 espaços);
- O padrão de indentação deve ser o mesmo em todo o programa e deve ser inserido na digitação do programa e não após ele já estar pronto;
- Nunca tentar redefinir palavras reservadas de uma linguagem;

Regras Básicas

- Devemos sempre que possível aproveitar ao máximo as linhas de código e as variáveis já existentes;
- Uma linha de comando termina sempre com um ponto-e-vírgula (;)
- Podemos ter dois comandos em uma mesma linha, desde que os mesmos estejam separados por ponto-e-vírgula (;)

Regras Básicas

- Recomenda-se quando na codificação dos algoritmos no computador o uso de letras minúsculas (facilidade de leitura), reservando as maiúsculas para algum tipo de destaque ou para constantes.
- As regras para nomes de identificadores (nomes de variáveis, nomes de funções) válidos não devem possuir espaços em branco.
- Recomenda-se o uso de letras ao início dos identificadores e não números;

Regras Básicas

- Os nomes de quaisquer identificadores (variáveis, nome de funções) não podem em hipótese nenhuma ser repetidos.
- Após o término de cada área do algoritmo ou após cada função é aconselhável deixarmos uma linha em branco para uma melhor organização.
- **O computador detecta somente erros de sintaxe, nunca erros de lógica.**

Regras Básicas

- Nomes de variáveis:
 - Inicia-se com letra minúscula e se a palavra for composta, o demais termos da composição devem ser iniciados por maiúsculas.
 - Deve-se sempre buscar nomes intuitivos, de acordo com a variável representa na lógica do programa. Isso facilita a leitura do código
 - Ex.:
 - Ao invés de *variavelC* para representar um resultado de uma soma use, *resultadoDaSoma* ou *resultadoSoma*

Regras Básicas

- Nome de funções/procedimentos:
 - Procure nomes intuitivos, de acordo com o significado do que a função realiza.
 - Inicie preferencialmente com um verbo.
 - Para nomes compostos siga a regra de iniciar com letras minúsculas e as demais iniciais de palavras maiúsculas.
- Ex.:
 - Função para calcular uma soma:
 - calculaSoma(int num1, int num2)