Curso de Linguagem C (Automação e Controle)

Aula 1

Alexandre Wolf
as_wolf@terra.com.br

Sobre a linguagem C

- Pode-se construir programas organizados e concisos ocupando pouco espaço de memória com alta velocidade de execução (como o Assembly);
- O C foi desenvolvido a partir da necessidade de se escrever programas que utilizassem recursos próprios da linguagem de máquina de uma forma mais simples e portável que o Assembly;

Características

- Portabilidade entre máquinas e sistemas operacionais;
- Dados compostos em forma estruturada.
- Programas Estruturados;
- Total interação com o Sistema Operacional;
- Código compacto (42 comandos ANSI) e rápido.

Nomes Reservados (42 comandos)

auto	double	if	static
break	else	int	struct
case	entry	long	switch
char	extern	register	typedef
continue	float	return	union
default	for	sizeof	unsigned
do	goto	short	while

Requisitos para Programação

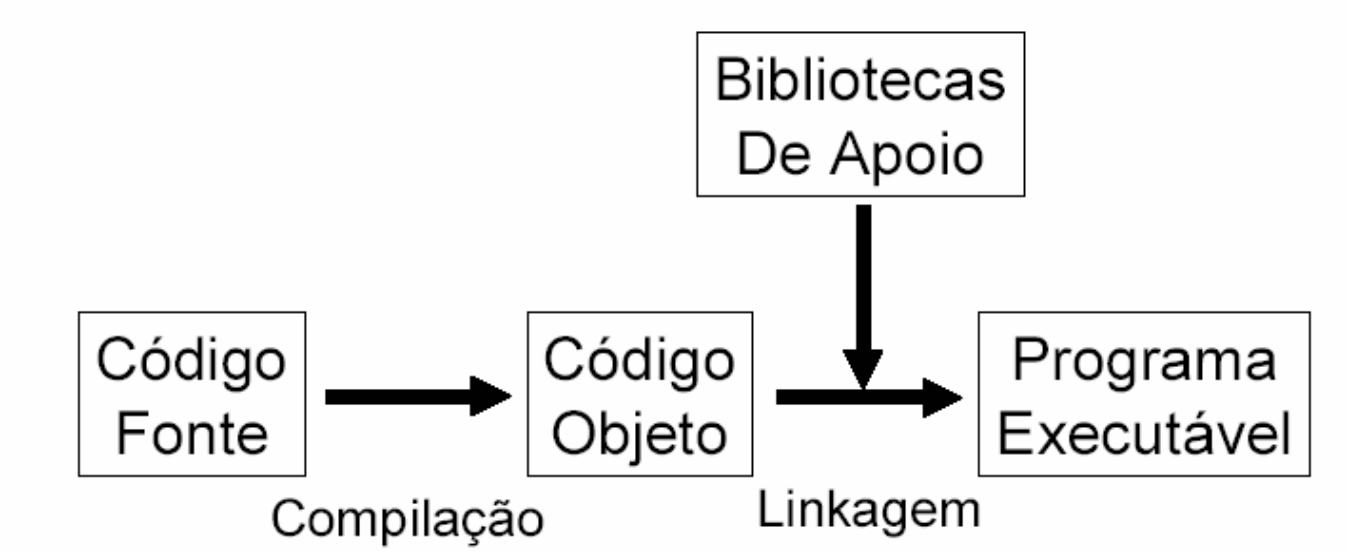
- Algum compilador de linguagem C (todos abaixo são gratuitos e muito bons)
  - GCC;
  - Turbo C;
    - (ótima sugestão, possui ambiente próprio para desenvolvimento, com facilidades para depuração)
  - Dev C++;
  - TCLite;
  - LCC;
  - DJGPP.

Definição de Alguns Padrões

- Para melhor compreensão, abaixo será apresentado o padrão utilizado para especificar a parametrização dos comandos:
  - <> Substituição obrigatória pelo que se pede;
  - [] Uso opcional;
  - Sem símbolos: uso obrigatório;

- Ex:
  - Eu, <Meu Nome> sou legal [e muito <Outro Adjetivo>].

Compilação??? Linkedição ???



- O código fonte .C (source code) é o programa propriamente dito, quando compilado, se não apresentar erros é gerado um arquivo .OBJ, após esse processo inicia-se o processo de Linkedição, ou seja, juntar o código sem erros de sintaxe com as bibliotecas de apoio, formando assim o programa executável .EXE.

Estrutura de um Programa C (Estrutura Mínima)

```
[<Bibliotecas de Apoio>]

main()
{
    <Corpo do Programa>
}
```

Estrutura de um Programa C (Estrutura Ideal)

```
[<Bibliotecas de Apoio>]

void main(void)
{
    <Corpo do Programa>
}
```

Importante

- Todos comandos devem terminar com o ponto e vírgula ; menos nos casos onde existe a possibilidade ou se abre chaves {};
- A linguagem C é case sensitive, assim sendo, deve-se ter muito cuidado ao se digitar os comandos (diferencia letras maiúsculas de minúsculas).

Operadores Aritméticos

- () Muda a prioridade aritmética;
- * Multiplicação;
- / Divisão;
- - Subtração;
- + Soma;
- % Resto da divisão inteira;

Atenção a prioridade aritmética, grande número de erros de programação se encontra no mal uso dos seus operadores.

Importante

- Quando se deseja imprimir ou armazenar um conjunto de caracteres (palavra), ou seja, uma string, o conteúdo deve estar entre aspas duplas, Ex: “A vaca foi para o Brejo”;
- Quando se deseja armazenar ou imprimir um único caracter (letra ou único símbolo), o mesmo deve estar entre aspas simples. Ex: ‘A’;
- Se for um número, não se usa nada, somente o próprio número, Ex: 10;

Comando de Saída de Tela

- **printf(<Expr>);**
  - Somente imprime na tela o que estiver especificado entre aspas “ “;
  - Para imprimir valores (números) deve-se utilizar caracteres de controle;
  - As expressões devem ser separadas por vírgula;

Caracteres de Controle
(+ utilizados)

- %c Imprime um caracter;
- %d Imprime um decimal inteiro com seu devido sinal;
- %f Imprime um número de ponto flutuante de precisão simples (bastante grande);
- %s Imprime uma string (conjunto de caracteres);
- %% Imprime o símbolo %.

Exemplo usando printf

```
void main(void)
{
    printf(“Esse é o meu primeiro programa”);
    printf(“%c é uma letra”, ‘j’);
    printf(“%d é um número inteiro”, 30);
    printf(“%f é um número de ponto flutuante”, 12.2);
    printf(“%5.2f é um número de ponto flutuante
       formatado”, 12.2);
    printf(“%s é legal”, “linguagem C”);
    printf(“%d + %d = %d”, 10, 21, 10+21);
}
```

Tipos de dados Primitivos
(Variáveis - Básicas)

- A linguagem C possui apenas 4 tipos de dados primitivos, e mais uma grande variedade de tipos compostos (vistos mais tarde). Os tipos de dados abaixo reservam espaços na memória do computador para armazenar dados:
  - char um único caracter;
  - int um número inteiro;
  - float um número de ponto flutuante simples;
  - double um número de ponto flutuante de dupla precisão;
  - void não é um tipo de dado, mas informa sem retorno.

Usando Variáveis

- Todas variáveis devem ser declaradas no inicio do programa, para que o compilador possa reservar esses espaços de memória na memória RAM do computador.

Ex:

```
int i;
double d, dNumero, Numero;
char Letra;
```

Exemplo usando Variáveis

```
void main(void)
{
    int i, j;
    i = 2;
    j = 3;
    printf(“Esse é o meu segundo programa”);
    printf(“%d + %d = %d”, i, j, i+j);
    i = i + 1;
    printf(“%d”, i );
    i = i + j;
    printf(“%d”, i );
}
```

Comando de Entrada de Dados

- **scanf(<Expr>);**
  - Deve-se possuir variáveis para armazenar a entrada de dados;
  - Somente obtém o que estiver especificado entre aspas “ “ (parecido com o printf);
  - As expressões devem ser separadas por vírgula;
  - Os caracteres de controle são equivalentes ao do printf;
  - Quando solicitar a variável você deverá colocar o símbolo & antes da mesma (isso será visto mais além (ponteiros)).

Exemplo usando scanf

```
void main(void)
{
    int i, j;
    scanf(“%d %d”, &i, &j);
    printf(“Esse é o meu terceiro programa”);
    printf(“%d + %d = %d”, i, j, i+j);
}
```

Principais Funções do Compilador

- Ctrl + F9, compila e executa se tudo ok;
- F9, somente compila;
- Alt + F5, apresenta o resultado do programa na tela (modo DOS texto).