UNIPAC

Universidade Presidente Antônio Carlos

Bacharelado em Ciência da Computação

Introdução a Programação

Material de Apoio

Parte III

Prof. Nairon Neri Silva

naironsilva@unipac.br

1º sem / 2020

Material cedido pela professora Lívia

Linguagem de Programação C

- A linguagem de programação C foi originalmente projetada para ser implementada no sistema operacional UNIX em um DEC PDP-11.
- C é o resultado de um processo de desenvolvimento que começou com uma linguagem mais antiga, chamada BCPL.
- A linguagem BCPL influenciou uma linguagem chamada B, inventada por Ken Thompson.
- Na década de 70, B levou ao desenvolvimento da linguagem C.

Linguagem de Programação C

- O padrão C foi a versão fornecida com o sistema operacional UNIX versão 5.
- Com a popularidade dos microcomputadores, um grande número de implementações de C foi criada.
- Para remediar a falta de padrão da linguagem C, o ANSI (American National Standards Institute) estabeleceu, em 1983, um comitê para criar um padrão definitivo da linguagem C.

Características da Linguagem C

- Linguagem dita de médio nível;
- Linguagem Estrutural;
- Fácil portabilidade entre hardwares e sistemas operacionais;
- Alta interatividade com o sistema operacional;
- Código compacto e rápido.

Características da Linguagem C

- Ao contrário do que possa parecer, nem todas as linguagens foram feitas para programadores. C é virtualmente única, porque ela foi criada, influenciada e testada em campo por programadores.
- Oferece ao programador exatamente o que ele quer: poucas restrições e queixas, código rápido e eficiência. Por isso ela é a linguagem mais popular entre os programadores profissionais altamente qualificados.

Características da Linguagem C

- A linguagem C é sensível ao caso (*case sensitive*), isto quer significa que letras maiúsculas e minúsculas são tratadas como caracteres separados.

Abc *é diferente de* ***abc***

Interpretação versus Compilação

- Um interpretador lê linha a linha do código-fonte, o examina sintaticamente e o executa.
- Um compilador lê todo programa e o converte em código objeto (código de máquina), a partir daí pode ser executado.
- A linguagem C é compilada.

Estrutura básica de um programa em C

- A diretiva `#include` instrui o compilador a ler outro arquivo-fonte adicionado àquele que contém a diretiva `#include`. O nome do arquivo adicional deve estar entre aspas ou símbolos de maior e menor.

  Sintaxe:

```
#include <nome_da_biblioteca>
#include “nome_da_biblioteca”
```

Estrutura básica de um programa em C

- Comentários: são delimitados entre os sinais /* e */. Podem vir em qualquer posição do programa e não apenas em linhas separadas. Eles também podem começar em uma linha e terminar em outra.

  Sintaxe:

```
/* Escreva aqui seu comentário */
```

Estrutura básica de um programa em C

- Ponto e vírgula: é um terminador de comandos, por isso, todos os comandos devem ser terminados por um.

- Desta forma, podemos ter vários comandos numa mesma linha sendo cada um terminado com um ponto e vírgula.

Estrutura básica de um programa em C

- Por ser uma linguagem estruturada, a linguagem C permite a criação de blocos de código. Um bloco de código é um grupo de comandos de programa conectados logicamente que o computador trata como uma unidade.
- Para criar um bloco de código, coloque uma sequência de comandos entre chaves.

```
{

}
```

Palavras Reservadas

- Uma palavra reservada é essencialmente um comando, e na maioria das vezes, as palavras reservadas de uma linguagem definem o que pode ser feito e como será feito.
- As palavras reservadas não podem ser usadas a não ser nos seus propósitos originais, isto é, não podemos declarar funções ou variáveis com os mesmos nomes.

Palavras Reservadas

- Palavras reservadas do ANSI C:

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

Tipos de Dados

- Assume-se que cada constante, variável, expressão ou função é um certo tipo de dados. Esse tipo refere-se essencialmente ao conjunto de valores que uma constante variável, etc. pode assumir.
- Tipo primitivos de dados: essencialmente são os números, valores lógicos, caracteres, etc que são identificados na maioria das linguagens.
- A escolha da representação dos dados é determinada, entre outras, pelas operações a serem realizadas sobre os dados.

Tipos de Dados em C

Tipo	Num de bits	Intervalo	
		Inicio	Fim
char	8	-128	127
unsigned char	8	0	255
signed char	8	-128	127
int	16	-32.768	32.767
unsigned int	16	0	65.535
signed int	16	-32.768	32.767
short int	16	-32.768	32.767
unsigned short int	16	0	65.535
signed short int	16	-32.768	32.767
long int	32	-2.147.483.648	2.147.483.647
signed long int	32	-2.147.483.648	2.147.483.647
unsigned long int	32	0	4.294.967.295
float	32	3,4E-38	3.4E+38
double	64	1,7E-308	1,7E+308
long double	80	3,4E-4932	3,4E+4932

O tipo void declara explicitamente uma função que não retorna valor algum.

Correspondências: Portugol e C

Portugol	C
inteiro	int
real	float
caracter	char
logico	-

Declaração e Inicialização de variáveis

- Variáveis: identificadores que representam valores de dados num programa, capaz de receber valores de acordo com o seu tipo.

- Sintaxe:

```
<tipo_da_variável> <lista_de_variáveis>;
```

- Exemplo:

```
char opcao;
int numero;
```

Declaração e Inicialização de variáveis

- As variáveis podem ser declaradas dentro de qualquer bloco de código, mas no início do bloco em que são definidas, e antes de serem utilizadas.

- A principal vantagem em declarar uma variável local dentro de um bloco condicional é que a memória para ela só será alocada se necessário.

Declaração e Inicialização de variáveis

- Exemplo:

```
int main ()
{
   int i;
   i = 10;
   int j;     /*Esta linha irá provocar um erro*/
}
```

Tipos de variáveis

- Existem 3 tipos de variáveis, dependendo do lugar onde as variáveis podem ser declaradas:
  - Variável global: variável declarada fora de todas as funções, incluindo a função main(), e pode ser utilizada em qualquer parte do programa.
  - Variável local: variável declarada dentro de uma função e pode ser utilizada somente pelos comandos que estiverem na mesma função.
  - Parâmetro: variável declarada como parâmetro formal de uma função, embora seja utilizada para receber os argumentos quando a função é chamada, ela pode ser utilizada como outra variável qualquer.

Comando de atribuição

- O comando de atribuição (=) é utilizado para atribuir um determinado valor a uma variável.

- Sintaxe:

```
<variável> = <valor>;
```

- Podemos inicializar uma variável no momento em que a declaramos, exemplo:

```
int x = 10;
```

Constantes

- Constantes referem-se a valores fixos que o programa não pode alterar, ou seja, não podem ser modificadas no decorrer do programa.

- Sintaxe:

```
const <tipo> <nome_constante> = <valor>;
```

- Exemplo:

```
const int a = 10; /*cria a constante a, com
                    valor inicial 10, que não
                    poderá ser modificado*/
```

Operadores aritméticos

- Operações aritméticas possíveis:

Significado	Operador	Exemplo
Adição	+	x = x + 1;
Subtração	-	x = x - 1;
Multiplicação	*	x = x * 1;
Divisão	/	x = x / 1;
Resto divisão	%	x = x % 2;

Abreviações de Expressões

- É possível abreviar expressões aritméticas:

Expressão Original	Expressão Equivalente
x = x + 1;	x + = 1;
x = x - 1;	x - = 1;
x = x * 1;	x * = 1;
x = x / 1;	x / = 1;

x = x + 1;	x++; //pos incremento
	++x; //pre incremento

x = x - 1;	x--; //pos incremento
	--x; //pre incremento

Operadores relacionais

- Operações relacionais possíveis:

Operador	Ação
>	Maior do que
>=	Maior ou igual do que
<	Menor do que
<=	Menor ou igual do que
==	Igual a
!=	Diferente de

Operadores lógicos

- Operações lógicas possíveis:

Operador	Ação
&&	E (and)
||	OU (or)
!	Não (negação)

Exemplo

```
int soma; /* Variável global */

void main()
{
   int cont = 8;      /* Variável local */
   soma = 0;      /* atribui valor a variável soma */
   soma = cont++; /* atribui uma cópia do valor de cont à
                     variável soma, e depois acrescenta uma
                     unidade à variável cont */
}
```

Entrada e Saída pelo console – printf()

- Realiza uma saída formatada, isto é, pode escrever na tela em vários formatos que estão sob seu controle.
- Está contida na biblioteca stdio.h
- Sintaxe:

```
printf(<string_de_controle>,<lista_argumentos>);
```

Entrada e Saída pelo console – printf()

- Alguns códigos de controle:

Código	Significado
%d	Inteiro
%f	Float
%c	Caractere
%s	String
%%	Coloca na tela um %

Entrada e Saída pelo console – printf()

- Exemplo:

```
#include <stdio.h>
main ()
{
   char Ch;
   Ch='D';
   printf ("%c",Ch);
}
```

Entrada e Saída pelo console – scanf()

- Rotina de entrada pelo console de uso geral.
- Sintaxe:

```
scanf(string-de-controle,lista-de-argumentos);
```

- Exemplo:

```
#include <stdio.h>
main ()
{
   int i;
   scanf("%d", &i);
   printf ("%d",i);
}
```

Entrada e Saída pelo console – scanf()

- Exemplo de entrada e saída de String

- Exemplo:

```
#include <stdio.h>
main ()
{
   char nome[20];
   scanf("%s", &nome);
   printf ("%s", nome);
}
```

Exercícios (já resolvidos em Portugol!)

24) Leia nome, idade e salário de um funcionário e exiba os mesmos dados, contudo o salário deve ser reajustado em 12%.

25) Leia a base e a altura de um triângulo. Em seguida, escreva a área do mesmo.

26) Transforme de um valor em dólar, para reais. Peça ao usuário o valor da cotação atual.

Referências Bibliográficas

- FORBELLONE, André Luiz Villar; EBERSPACHER, Henri Frederico. *Lógica de Programação*. Makron books.
- GUIMARAES, Angelo de Moura; LAGES, Newton Alberto Castilho. *Algoritmos e estruturas de dados*. LTC Editora.
- FIDALGO, Robson. *Material para aulas*. UFRPE.
- LOPES, Anita; GARCIA, Guto. *Introdução à programação – 500 algoritmos resolvidos*. Elsevier.
- http://www.ime.usp.br/~pf/algoritmos/