Aulas 04 - Estrutura linear

Estrutura lógica linear

A estrutura lógica linear é a base de toda a programação. É dessa forma (linear) que funcionam todos os programas mais simples. Ela não é suficiente para compor o mais complexo dos algoritmos, mas nós veremos nos capítulos seguintes, novas estruturas que irão complementá-la.

O que a estrutura linear representa, na verdade, é a sequência fundamental da computação ENTRADA/PROCESSAMENTO/SAÍDA de dados, responsável, inclusive, pela composição básica do hardware da máquina.

Como um exemplo desta primeira estrutura lógica, podemos usar o algoritmo para cálculo da idade:

- Leia da entrada o Ano de Nascimento.
- Leia da entrada o Ano Atual.
- Faça idade receber o resultado de “Ano Atual - Ano Nascimento”.
- Escreva a idade na saída.

Entrada/saída padrão

Função printf()

A função **printf()** tem a seguinte forma geral:

```
printf (string_de_controle,lista_de_argumentos);
```

Teremos, na string de controle, uma descrição de tudo que a função vai colocar na tela. A string de controle mostra não apenas os caracteres que devem ser colocados na tela, mas também quais as variáveis e suas respectivas posições. Isto é feito usando-se os códigos de controle, que usam a notação **%**. Na string de controle indicamos quais, de qual tipo e em que posição estão as variáveis a serem apresentadas. É muito importante que, para cada código de controle, tenhamos um argumento na lista de argumentos. Apresentamos agora alguns dos códigos **%**:

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

Vamos ver alguns exemplos de **printf()** e o que eles exibem:

`printf ("Teste %% %%")` exibe "Teste % %"

`printf ("%f",40.345)` exibe "40.345"

`printf ("Um caractere %c e um inteiro %d",'D',120)` exibe "Um caractere D e um inteiro 120"

`printf ("%s e um exemplo","Este")` exibe "Este e um exemplo"

`printf ("%s%d%%","Juros de ",10)` exibe "Juros de 10%"

Maiores detalhes sobre a função **printf()** (incluindo outros códigos de controle) serão vistos posteriormente.

Para exercitar:

Escreva um programa que declare seis variáveis inteiras e atribua os valores 10, 20, 30, ..., 60 a elas. Declare 6 variáveis caracteres e atribua a elas as letras c, o, e, l, h, o. Finalmente, o programa deverá imprimir, usando todas as variáveis declaradas:

As variáveis inteiras contêm os números: 10, 20, 30, 40, 50, 60.

O animal contido nas variáveis caracteres é o coelho.

Funções scanf() e gets()

O formato geral da função **scanf()** é:

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

Usando a função **scanf()** podemos pedir dados ao usuário. Um exemplo de uso, pode ser visto acima. Mais uma vez, devemos ficar atentos a fim de colocar o mesmo número de argumentos que o de códigos de controle na string de controle. Outra coisa importante é lembrarmo-nos de colocar o **&** antes das variáveis da lista de argumentos. É impossível justificar isto agora, mas veremos depois a razão para este procedimento. Exemplos do uso da função scanf() com a função printf():

```
printf("Que ano voce nasceu?\n");
scanf("%f",&ano_nasc);

printf("Em que ano estamos?\n");
scanf("%f",&ano_atual);
```

O formato geral da função **gets()** é:

```
gets (nome_da_string);
```

Se quisermos ler uma string (palavra ou frase) fornecida pelo usuário podemos usar a função **gets()** (gets = get string). Um exemplo do uso desta função é apresentado abaixo. A função **gets()** coloca o terminador nulo na string, quando você aperta a tecla "Enter" ocupando com o “Enter” a primeira posição.

```
#include <stdio.h>
int main ()
{
    char nome[100]; //aceita frase de até 100 caracteres
    printf ("Digite uma palavra ou frase: ");
    gets (nome);
    printf ("\n\nVoce digitou %s\n\n",nome);
    system("pause");
    return(0);
}
```

Neste programa, o tamanho máximo da string que você pode entrar é uma string de 99 caracteres. Se você entrar com uma string de comprimento maior, o programa irá aceitar, mas os resultados poderão conter erros.

Outra forma bastante utilizada de se ler uma string com espaços, é através do próprio **scanf()**. O formato desta leitura é da seguinte forma:

```
scanf("%50[^\n]s", nome_da_string);
```

Essa forma é a mais recomendada para ler uma string com espaços. Após o %, pode-se especificar o tamanho máximo que será lido. Uma maneira mais fácil de ler uma string com espaços é tirando o tamanho após o % e também a letra s após os colchetes.

```
scanf("%[^\n]", nome_da_string);
getchar();
```

Mas tenha em mente, toda vez que utilizar o formato acima, logo abaixo e em TODAS as letituras que virão posteriormente, você deverá utilizar a função **getchar()** após o **scanf()** para não pular as outras leituras. Nunca se esqueça disso.

Para ler uma string sem espaços, isto é, até que seja digitado um espaço ou um Enter, basta fazer o seguinte:

```
scanf("%s", nome_da_string);
```

Perceba que o & **não** é necessário quando se lê strings (sendo com espaços ou não).

Constantes de barra invertida

O C utiliza, para nos facilitar a tarefa de programar, vários códigos chamados códigos de barra invertida. Estes são caracteres que podem ser usados como qualquer outro. Uma lista com alguns dos códigos de barra invertida é dada a seguir:

Código	Significado
\b	Retrocesso ("back")
\f	Alimentação de formulário ("form feed")
\n	Nova linha ("new line")
\t	Tabulação horizontal ("tab")
\"	Aspas
\'	Apóstrofo
\0	Nulo (0 em decimal)
****	Barra invertida
\v	Tabulação vertical
\a	Sinal sonoro ("beep")
\N	Constante octal (N é o valor da constante)
\xN	Constante hexadecimal (N é o valor da constante)

Reedite o mesmo programa criado anteriormente com as novas funções:

Escreva um programa que declare seis variáveis inteiras e atribua a cada variável um valor digitado pelo usuário. Declare 6 variáveis caracteres e atribua a elas uma letra digitada pelo usuário. Finalmente, o programa deverá imprimir, usando todas as variáveis declaradas. Utilize constantes de barra invertida para melhorar a disposição dos dados na tela.

Código fonte comentado com as funções estudadas:

```c
#include <stdio.h>

 //programa que converte dias em anos
 int main ()
{
 int dias;            /* Declaracao de Variaveis */
 float anos;

 printf ("Entre com o numero de dias: "); /* Entrada de Dados */
 scanf ("%d",&dias);

 anos = dias/365.25;       /* Conversao Dias->Anos */

 printf ("\n\n%d dias equivalem a %f anos.\n",dias,anos);
 system("pause");
 return(0);
}
```

Vamos entender como o programa acima funciona. São declaradas duas variáveis chamadas **dias** e **anos**. A primeira é um **int** (inteiro) e a segunda um **float** (ponto flutuante). As variáveis declaradas como ponto flutuante existem para armazenar números que possuem casas decimais, como 5.1497.

É feita então uma chamada à função **printf()**, que coloca uma mensagem na tela **"Entre com o numero de dias:**".

Queremos agora ler um dado que será fornecido pelo usuário e colocá-lo na variável inteira **dias**. Para tanto usamos a função **scanf()**. O **"%d"** diz à função que iremos ler um inteiro. O segundo parâmetro passado à função diz que o dado lido deverá ser armazenado na variável **dias**. É importante ressaltar a necessidade de se colocar um **&** antes do nome da variável a ser lida quando se usa a função **scanf()**. O motivo disto só ficará claro mais tarde. Observe que, no C, quando temos mais de um parâmetro para uma função, eles sempre serão separados por vírgula.

Temos então uma expressão matemática simples que atribui a **anos** o valor de **dias** dividido por 365.25 (365.25 é uma constante ponto flutuante 365,25). Como **anos** é uma variável **float** o compilador fará uma conversão automática entre os tipos das variáveis.

A segunda chamada à função **printf()** tem três argumentos. A string **"\n\n%d dias equivalem a %f anos.\n"** diz à função para pular duas linhas, colocar um inteiro na tela, colocar a mensagem **" dias equivalem a "**, colocar um valor **float** na tela, colocar a mensagem **" anos."** e pular outra linha. Os outros parâmetros são as variáveis, **dias** e **anos**, das quais devem ser lidos os valores do **inteiro** e do **float**, **respectivamente**.

O **system("pause");** é utilizado para pausar a execução do programa em uma dada posição. Ele não é estritamente necessário, e em muitos casos, seu uso não é recomendado.

Exercícios resolvidos:

1) Suponha que desejamos um programa que imprima na tela a soma de três números lidos. Este é um problema bastante simples. Teríamos que executar a seguinte sequência:

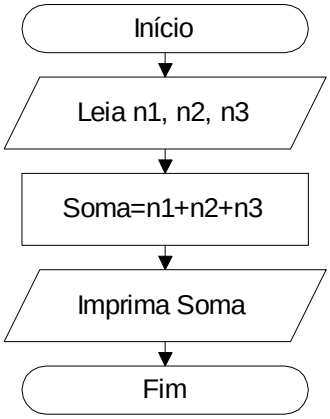
Algoritmo:

1- ler os três números

2- somá-los

3- imprimir o resultado.

A solução gráfica é apresentada a seguir:



Codificação:

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

int main ()
{
    int a,b,c, soma;

    printf("Digite o primeiro numero: \n");
    scanf("%d",&a);

    printf("Digite o segundo numero: \n");
    scanf("%d",&b);

    printf("Digite o terceiro numero: \n");
    scanf("%d",&c);

    soma = a+b+c;

    printf("\t\tSOMA DOS 3 NUMEROS: %d\n", soma);
    system("pause");
    return 0;
}
```

2) O problema consiste em determinar o volume de um sólido. Para que isto aconteça, preciso ter em mãos a altura, o comprimento e a largura deste sólido.

Algoritmo:

1- Ler a altura do sólido.

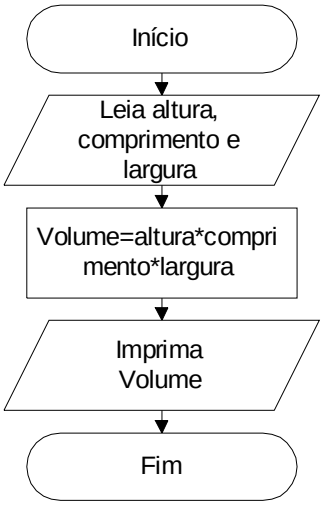
2- Ler o comprimento do sólido.

3- Ler a largura do sólido.

4- Determinar Volume=altura*comprimento*largura

5- Imprimir Volume

A solução gráfica é apresentada a seguir:



Codificação:

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

int main ()
{
    float altura,comprimento,largura,volume;

    printf("\n\nDigite o comprimento do solido: ");
    scanf("%f",&comprimento);

    printf("Digite a altura do solido: ");
    scanf("%f",&altura);

    printf("Digite a largura do solido: ");
    scanf("%f",&largura);

    volume=comprimento*altura*largura;

    printf("O volume do solido e: %.1f m3\n",volume);
    system("pause");
    return 0;
}
```

Documentação interna ao programa (comentários)

Como todo bom programador deve saber, nunca teremos bons programas se eles não possuírem uma documentação bem feita, que esclareça principalmente as estruturas utilizadas no programa e que seja capaz de explicar como elas funcionam. A partir disso, para cada um de nossos programas desenvolvidos, normalmente fazemos dois tipos de documentação:

- O primeiro, uma documentação em papel, que será anexada às listagens do programa, podendo conter uma série de exemplos ou até mesmo uma descrição, na forma de um manual, de como utilizá-lo de forma rápida e correta.
- E outro tipo, uma documentação interna ao programa, composta por observações ou comentários, que deverá acompanhar as diversas instruções e trechos do programa, esclarecendo-os.

Cada linguagem de alto nível dispõe de uma notação ou de instruções específicas para colocar estes comentários no meio do programa, sem que estes alterem o seu funcionamento normal, é claro.

O compilador C desconsidera qualquer coisa que esteja começando com **/ /**, outra forma de fazer um comentário é utilizando no início **/ *** e terminando com ***/**. Um comentário pode, inclusive, ter mais de uma linha.

Verifique como poderia ser documentado o nosso programa anterior:

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

//Autor: João da Silva
//Data:  01/01/10      }
/*  Este programa foi elaborado para o calculo  do
    volume de um solido qualquer, no formato de um
    paralelepipedo, e que possui tres dimensoes:
    altura, comprimento e largura. Seu volume  é
    calculado  simplesmente  pela  multiplicacao
    destes tres valores de entrada.*/

int main ()
{
    float altura,comprimento,largura,volume;
//leitura das 3 dimensoes
    printf("\n\nDigite o comprimento do solido: ");
    scanf("%f",&comprimento);

    printf("Digite a altura do solido: ");
    scanf("%f",&altura);

    printf("Digite a largura do solido: ");
    scanf("%f",&largura);
//calculo do volume
    volume=comprimento*altura*largura;
//impressao final do resultado
    printf("O volume do solido e: %.1f m3\n",volume);
    system("pause");
    return 0;
}
```

Exercício resolvido passo a passo - estrutura linear

Para se obter uma indicação da taxa de inflação sobre uma mercadoria, deve-se comparar o preço pago hoje pelo produto e o seu preço pago há algum tempo. Com o propósito de uniformidade, vamos supor que compremos regularmente uma vez por mês esta mercadoria.

Sendo fornecidos esses valores, escrever um algoritmo que seja capaz de calcular duas estatísticas indicadoras de nossa inflação. A primeira delas será a diferença algébrica entre os preços e a segunda a diferença percentual no mês. A diferença percentual é dada por

(PA - PV) / PV * 100, onde PA é o preço atual da mercadoria e PV o preço velho.

PASSO 1:

- Leia cuidadosamente a especificação do problema até o final, quantas vezes forem necessárias até entender bem o que o problema está exigindo. Enquanto não entender bem o enunciado do problema, não passe para o passo 2.

PASSO 2:

- Levantar e analisar todos os dados iniciais exigidos na especificação do problema.

As entradas exigidas nesse problema são:

o preço atual da mercadoria, que poderá ficar em PA

o preço velho da mercadoria, que poderá ficar em PV

PASSO 3:

- Levantar e analisar todos os resultados necessários e exigidos na especificação do problema.
- As saídas exigidas nesse problema são:
  a diferença algébrica entre os dois preços, que pode ficar em uma variável chamada DA.
  a diferença percentual no mês, na variável DP.

PASSO 4:

- Processamentos necessários:
  DA será calculada como PA - PV
  DP será calculada pela fórmula do enunciado, (PA - PV) / PV * 100

PASSO 5:

- Procurar escrever a solução lógica para esse problema, utilizando uma das técnicas apresentadas, como Fluxograma, ou outra qualquer que preferir.

Vamos lá:

Fluxograma

PASSO 6:

- Pronta a solução lógica, passamos à fase de testes, para saber se ela está correta ou se precisa de reparos antes da codificação. É importante salientar que o teste de mesa deve ser aplicado antes da codificação, porque em caso de se encontrar erros, o algoritmo precisará ser modificado e a codificação estaria perdida se já tivesse sido realizada.
- Teste de mesa:

Linhas do algoritmo:	PA	PV	DA	DP	SAÍDA
1	49.80				
2		46.20			
3			3.60		
4				7.79	
5					3.60 e 7.79 %

Observe pelo teste de mesa, que um produto com o preço atual de 49.80 e que custava 46.20 permitiu um índice de aumento de 3.60 (em valor de moeda) e de 7.79 % em termos percentuais. O valor da moeda pode ser considerado como real, ou qualquer outra moeda.

PASSO 7:

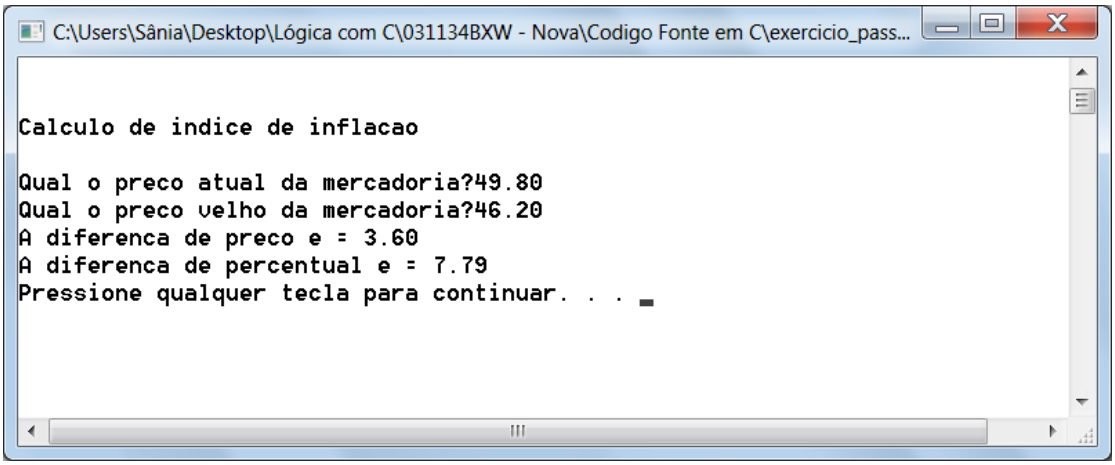
- Proceder à codificação na linguagem C:

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

int main ()
{
//declaracao das variaveis
    float da,dp,pa,pv;

    printf("\n\nCalculo de indice de inflacao\n\n");

    printf("Qual o preco atual da mercadoria?");
    scanf("%f",&pa);

    printf("Qual o preco velho da mercadoria?");
    scanf("%f",&pv);
//calculo do indice
    da = pa-pv;
    dp =(pa-pv)/pv*100;
//impressao final dos resultados
//2.f siginifica 2 casas decimais apos a virgula
    printf("A diferenca de preco e = %.2f\n",da);
    printf("A diferenca de percentual e = %.2f\n",dp);

    system("pause");
    return 0;
}
```

PASSO 8:

- Efetuar os testes pelo computador, usando em uma primeira instância a própria massa de dados utilizada no teste de mesa. Produzir depois testes com outras mercadorias e outros valores.

A figura a seguir, mostra como seria apresentada na tela do computador a execução deste exemplo já compilado em disco.



Note que o resultado é mostrado em notação científica. Não se preocupe com isso por enquanto. No próximo capítulo nós trataremos deste assunto.

PASSO 9:

- Gravar e listar o programa. Anexar a solução lógica, o teste de mesa, a listagem do programa e demais anotações para a documentação do programa.

Atividades

1. Nos itens a seguir são apresentados os textos de alguns pequenos programas escritos em C. Todos eles contêm erros de sintaxe. Pedimos que você os identifique e descreva a sua forma correta:

a.

```c
#include<stdio.h>

//Este programa escreve uma mensagem 'Oi !' na terceira linha da tela
int main ()
{
    printf("\t\tPrograma teste1");

    printf("Oi");

    return 0;
}
```

Erros encontrados:

b.

```c
#include<stdio.h>

//Este programa calcula o resultado da expressão 3+4*(8-32.8)
int main ()
{
    print("\t\tPrograma teste2\n\n");

    printf("Resultado: ",3+4*(8-32,8))

    return 0;
}
```

Erros encontrados:

c.

```
#include<stdio.h>

int main ()
{
    int i,j,k;

    printf("\t\tPrograma teste3\n\n");

    printf("Primeiro numero: \n");
    scanf("%D",&i);

    printf("Segundo numero: \n");
    scan("%d",j);

    k = i+j

    printf("% + %d = f \n", i;j;k);
    system("pause");
    return 0;
}
```

Erros encontrados:

2. Mais alguns erros, ainda do tipo detectados durante a compilação. Identifique-os propondo soluções:

a.

```
#include<stdio.h>

int main ()
{
    int i,j,k;

    printf("\t\tPrograma teste4\n\n");

    printf("Quer saber quantas polegadas tem em um metro?");
    system("pause");

    printf("Em 1 metro temos %d polegadas.\n",100/2.54);

    return 0;
}
```

Erros encontrados:

b.

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

int main ()
{
    int cot_dolar,sal_dolar;sal_real;

    printf("\t\Programa teste5\n\n");

    printf("Qual a cotacao atual do dolar?\n");
    scanf("%f",&cot_dolar);

    printf("\n\nQual o seu salario em reais?\n");
    scanf("%d",sal_real);

    sal_real = sal_real/cot_dolar

    printf("\n\nSeu salario convertido vale %f dolares\n",sal_dola);

    system("pause");
    return 0;
}
```

Erros encontrados:

c.

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

int main ()
{
    float num1, num2;
    char soma;

    printf("\t\Programa teste6\n\n");

    soma = num 1 + num2;

    printf("\n\nA soma de %.2f com %.2f e igual a %.2d \n\n",soma,num2,num1);

    system("pause");
    return 0;
```

Erros encontrados:

d.

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

int main ()
{
    printf("\t\Programa teste7\n\n");

    soma = num1 + num2;

    printf("\n\nA soma de %.2f com %.2f e igual a %.2f \n\n",num1,num2,soma);
    system("pause");
    return 0;
}
```

Erros encontrados:

3. Comente o que aparece no programa abaixo que pode ser retirado sem prejudicar o seu funcionamento normal:

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

int main ()
{
    float num1=5.0;
    float num2=3.0;
    float soma;

    printf("\t\Programa teste8\n\n");

    soma = num1 + num2;

    printf("\n\nA soma de %.2f com %.2f e igual a %.2f \n\n",num1,num2,num1+num2);
    printf("Tiau!!\n");
    system("pause");
    return 0;
}
```

1._____
2._____
3._____
4._____
5._____

4. Digitar, compilar e executar: Sendo fornecidos na entrada o número total de funcionários de uma determinada empresa, a quantidade destes funcionários que tem menos de 18 anos e a quantidade com mais de 35 anos, calcule e imprima os seguintes dados:

- Qual a porcentagem dos funcionários da empresa com menos de 18 anos?
- Qual a porcentagem dos que se encontram entre 18 e 35 anos?
- Qual a porcentagem acima dos 35 anos de idade?

5. Elabore um programa que leia três notas de um aluno, calcule e mostre em tela a sua média final.

6. Agora, vamos melhorar um pouquinho o exercício anterior. Suponha que você fez três provas e entregou um projeto. As provas têm peso 2, 3 e 4 respectivamente e o projeto, peso 1. Calcule a média ponderada.

7. Faça um programa que calcule a média da temperatura de uma determinada semana.

---

8. O que você faria se seu professor de matemática pedisse para você elaborar um programa que determine, a partir de um raio (R) dado, o comprimento, a área e o volume de uma circunferência?

Faça este programa e teste-o para vários valores diferentes.

9. Suponha que você tenha uma casa e queira determinar a quantidade de metros quadrados de cada um dos três quartos, da sala, da cozinha e da garagem. Faça um programa capaz de fornecer todos esses dados e calcular ainda a quantidade de metros de toda a casa. Considere a medida da casa como sendo a soma das medidas dos cômodos.

10. Invente um problema qualquer.

a. Escreva o algoritmo e a solução gráfica deste problema. Codifique e execute em C.