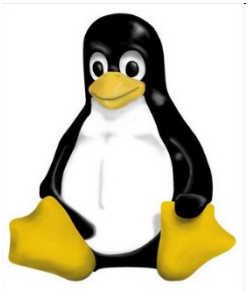
INSTITUTO FEDERAL
BAIANO

Estrutura de Dados

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org
http://softwarelivre.org/zenorato/honoratonunes

INSTITUTO FEDERAL
BAIANO



Introdução da disciplina e Revisão de Linguagem C

Prof. José Honorato Ferreira Nunes

honoratonunes@softwarelivre.org

http://softwarelivre.org/zenorato/honoratonunes

Resumo da aula

- Considerações Gerais
- Revisão da Linguagem C
  - Variáveis
  - Tipos de dados
  - Operadores e Expressões
  - Comandos de Entrada e Saída
  - Estruturas de Seleção e Repetição

Prof. José Honorato F. Nunes
honoratonunes@gmail.com

Considerações Gerais

A automatizaçäo de tarefas é um aspecto marcante da sociedade moderna e na ciência da computação houve um processo de desenvolvimento simultâneo e interativo de máquinas (hardware) e dos elementos que gerenciam a execução automática (software) de uma tarefa.

Nesta grande evolução do mundo computacional, um fator de relevante importância é a forma de armazenar as informações. Então de nada adiantaria o grande desenvolvimento do hardware e do software, se a forma de armazenamento e tratamento da informação não acompanhasse esse desenvolvimento.

Prof. José Honorato F. Nunes
honoratonunes@gmail.com

Considerações Gerais

As estruturas de dados são formas otimizadas de armazenamento e tratamento das informações eletronicamente.

As estruturas de dados, na sua maioria dos casos, foram espelhadas em formas naturais de armazenamento do nosso dia a dia, ou seja, nada mais são que a transformação de uma forma de armazenamento já conhecida e utilizada no nosso mundo para o mundo computacional. Por isso, cada tipo de estrutura de dados possui vantagens e desvantagens e cada uma delas tem sua área de atuação (massa de dados) ótima.

Prof. José Honorato F. Nunes
honoratonunes@gmail.com

Introdução a Linguagem C

A linguagem C foi criada por Dennis Ritchie, em 1972, no centro de Pesquisas da Bell Laboratories. Sua primeira utilização importante foi a reescrita do Sistema Operacional UNIX, que até então era escrito em assembly.

Em meados de 1970 o UNIX saiu do laboratório para ser liberado para as universidades. Foi o suficiente para que o sucesso da linguagem atingisse proporções tais que, por volta de 1980, já existiam várias versões de compiladores C oferecidas por várias empresas, não sendo mais restritas apenas ao ambiente UNIX, porém compatíveis com vários outros sistemas operacionais.

Prof. José Honorato F. Nunes
honoratonunes@gmail.com

Introdução a Linguagem C - sintaxe

Uma função importante em todo programa em C é a função main (cuja tradução é principal). Esta será sempre a primeira função do programa a ser executada.

```
main ( )

{

}
```

Variáveis

Como o próprio nome sugere, as variáveis, podem conter valores diferentes a cada instante de tempo, ou seja, seu conteúdo pode variar de acordo com as instruções do algoritmo.

As variáveis são referenciadas através de um nome (identificador) criado por você durante o desenvolvimento do algoritmo.

Variáveis

É um local na memória principal, isto é, um endereço que armazena um conteúdo (informação) que pode ser modificado.

Em C, não é possível ter variáveis que comecem com dígito e espaços não são permitidos.

Exemplos de nomes de variáveis indevidas: 2w, peso do aluno, sal/hora.

Observação: em C usualmente são utilizadas variáveis em minúsculo e constantes em maiúsculos.

Variáveis

EXEMPLOS DE NOMES VÁLIDOS	EXEMPLOS DE NOMES INVÁLIDOS
nome_candidato	nome candidato
endereco	endereço
RG	R.G.
mes_ferias	mês_férias
dataNasc	data-Nasc
fone1	1fone

Prof. José Honorato F. Nunes
honoratonunes@gmail.com

Tipos de dados

Quando declaramos uma variável, precisamos identificar o tipo de informação que desejamos armazenar nela. Existem diversos tipos de dados e muitos deles são comuns na grande maioria das linguagens de programação. No nosso estudo de lógica de programação, porém, utilizaremos apenas alguns dos principais.

Prof. José Honorato F. Nunes
honoratonunes@gmail.com

Tipos de dados

Os tipos de dados básicos com os quais iremos trabalhar são:

Nome	Tamanho em bits	Faixa de valores
char	8	-128 a 127
int	16	-32.768 a 32.768
float	32	10^{-38} a 10^{38}
long int	32	-2.147.483.648 a 2.147.483.648
double	64	10^{-308} a 10^{308}

Declaração e Atribuição de Variáveis

Exemplo: int numero, soma;
char nome;

Exemplos: a = 5;

Prof. José Honorato F. Nunes
honoratonunes@gmail.com

Operadores

=	atribuição
!=	diferente
<=	Menor igual
>=	Maior igual
==	igual
%	Resto da divisão entre inteiros

Operadores

Operadores Aritméticos:

Aritméticos	Tipo	Operação	Prioridade
+	Binário	Adição	5
-	Binário	Subtração	5
%	Binário	Resto da divisão	4
*	Binário	Multiplicação	3
/	Binário	Divisão	3
++	Unário	Incremento	2
--	Unário	Decremento	2
+	Unário	Manutenção do sinal	1
-	Unário	Inversão do sinal	1

Operadores

Exemplo:

```
a=a+b;
a = 4 * 2 + 3;
```

Observação:

a++ é similar a **a = a + 1;**

b-- é similar a **b = b -1;**

Operadores

Outros Operadores:

Relacionais		Lógicos	
>	Maior que	&&	And (e)
>=	Maior ou igual	||	Or (ou)
<	Menor que	!	Not (não)
<=	Menor ou igual		
==	Igual		
!=	Diferente		

Prof. José Honorato F. Nunes
honoratonunes@gmail.com

Comandos de Entrada e Saída

Comando de Impressão printf/saída (Escreva).

Através da função pré-definida printf(), cujo protótipo está contido também no arquivo stdio.h. Sua sintaxe é a seguinte:

printf(“Expressão” , lista de argumentos);

Sintaxe:

<expressão> Mensagens que serão exibidas.

<lista de argumentos> pode conter identificadores de variáveis, expressões aritméticas ou lógicas e valores constantes.

Comandos de Entrada e Saída

Código	Tipo	Elemento armazenado
%c	char	um único caractere
%d ou %i	int	um inteiro
%f	float	um número em ponto flutuante
%lf	double	ponto flutuante com dupla precisão
%e	float ou double	um número na notação científica
%s	.....	uma cadeia de caracteres

Comandos de Entrada e Saída

Primeiro Programa em C!

```
#include <stdio.h>

int main()
{
    printf("%s está a %d milhões de milhas do sol","Vênus",67);
}
```

Comandos de Entrada e Saída

Primeiro Programa em C!

```
#include <stdio.h>

int main()
{
  printf("Valor inteiro atribuído foi %d para o caracter %c e um float foi de %f ",99,"a",1.45);
}
```

Comandos de Entrada e Saída

Primeiro Programa em C!

```
#include <stdio.h>

 int  main()
 {
    printf(“Se quisesse imprimir uma string : %s“,”Minha string!”);
  }
```

Comandos de Entrada e Saída
Leitura de dados - scanf() - (Leia)

Ela é o complemento de printf() e nos permite ler dados formatados da entrada padrão (teclado). Sua sintaxe:

scanf("expressão de controle", argumentos);

Exemplo:

```
int m;
scanf (“%d”,&m);
```

- %d indicativo do tipo, neste caso do tipo inteiro.
- &m operador utilizado para obter o endereço de memória da variável.

Comandos de Entrada e Saída

```
#include <stdio.h>

int main () {
int idade;

printf(“Digite a sua idade : ”);
scanf(“%d”,&idade);
printf(“A sua idade é %d”,idade);
return 0;

}
```

Atividades

- Crie um algoritmo para calcular a média de consumo de combustível de um veículo qualquer. O usuário deverá informar: quilometragem inicial, quilometragem final e a quantidade de litros consumida durante a viagem.

Atividades

- Crie um algoritmo para armazenar o nome, sobrenome, idade e sexo de um aluno. O programa deve solicitar que o usuário informe os dados acima e depois mostrar essas informações na tela.

Estruturas de Controle: Seleção Simples

```
#include <stdio.h>

int main ()
{
    int idadeAluno;
    printf("Informe a idade do aluno: ");
    scanf("%d",&idadeAluno);

    if (idadeAluno < 18) {
        printf("O aluno é menor de idade.");
    }
    printf("A idade do aluno é:  %d",idadeAluno);
    return 0;
}
```

Estruturas de Controle: Seleção Composta

```
#include <stdio.h>
int main ()
{
    int idadeAluno;
    printf("Informe a idade do aluno: ");
    scanf("%d",&idadeAluno);

        if (idadeAluno < 18) {
            printf("O aluno é menor de idade.");
        }
        else {
            printf("O aluno é maior de idade.");
        }

    printf("A idade do aluno é:  %d",idadeAluno);
    return 0;
}
```

Atividade

Crie um algoritmo em C para calcular a média final do aluno. O sistema deve mostrar o nome do aluno e receber duas notas (trabalho e prova). Depois de calcular a média final, se o aluno tiver média >= 7 informar que ele foi aprovado, senão informar que ele foi reprovado.

Prof. José Honorato F. Nunes
honoratonunes@gmail.com

Estruturas de Controle: Seleção Aninhada

```
#include <stdio.h>
int main ()
{
      float media;
      char nome[30];
      printf(“Informe o nome do aluno: ”);
      scanf(“%s”,&nome);
      printf(“Informe a média do aluno: ”);
      scanf(“%f”,&media);
      if (media >= 7) {
            printf("O aluno %s está %s",nome,"aprovado");
      }
      else if (media >= 3)  {
            printf("O aluno %s está %s",nome,"na final");
      }
      else {
            printf("O aluno %s está %s",nome,"reprovado");
      }
      return 0;
}
```

Atividades

Faça um algoritmo que receba um número e mostre uma mensagem caso este número seja maior que 10.

Escrever um algoritmo que leia dois valores inteiro distintos e informe qual é o maior.

Faça um algoritmo que receba um número e diga se este número está no intervalo entre 100 e 200.

Atividades

Faça um algoritmo que receba três valores inteiros e organize esses valores em ordem decrescente.

Faça um algoritmo que solicite o nome, idade e sexo do aluno e caso ele seja homem e maior de idade solicitar ao aluno o número da carteira de reservista, caso seja homem menor de idade emitir a mensagem: ”aguardando idade para o exercício militar”, caso seja mulher emitir a mensagem: ”dispensada do exercício militar”.

Atividades

Escrever um algoritmo que leia o nome e as três notas obtidas por um aluno durante o semestre. Calcular a sua média (aritmética), informar o nome e sua menção aprovado (media >= 7), Reprovado (media <= 5) e Recuperação (media entre 5.1 a 6.9).

Atividade

- Crie um algoritmo, utilizando pseudocódigo, que leia dois números informados pelo usuário e, em seguida, exiba na tela uma mensagem informando se o maior deles é o primeiro, o segundo, ou se são iguais.

- Crie um algoritmo, utilizando pseudocódigo, que:

  Leia três números informados pelo usuário;

  Multiplique o menor valor lido pelo maior valor e some o resultado com o valor do meio;

  Exiba na tela o resultado da soma.

Atividade

Crie um algoritmo em pseudocódigo para aplicar um percentual de desconto sobre o valor de uma compra informado pelo usuário. Os percentuais de desconto são:

- 15% para compras acima de R$ 500,00;
- 10% para compras entre R$ 200,00 e R$ 499,99;
- 5% para compras abaixo de R$ 200,00.

Mostre na tela uma mensagem informando: valor antes do desconto, valor do desconto e valor a ser pago.

Repetição com pré-teste: while

Sintaxe:
while(condição) comando;

Uma maneira possível de executar um laço é utilizando o comando while. Ele permite que o código fique sendo executado numa mesma parte do programa de acordo com uma determinada condição.

- o comando pode ser vazio, simples ou bloco
- ele é executado desde que a condição seja verdadeira
- testa a condição antes de executar o laço

Repetição com pré-teste: while

Ex:

```
#include <stdio.h>
main()
{
    char ch;
    while(ch!='a') ch=getchar();
}
```

Atividades: utilizar estrutura de repetição com Pré-Teste (while)

- Crie um algoritmo que leia um valor inteiro para X e escreva na tela X^3. O algoritmo deve continuar pedindo o valor de X até que o usuário informe 0 (zero), então o programa encerra.

- Desenvolva um algoritmo capaz de apresentar na tela o fatorial de um número inteiro informado pelo usuário.

Repetição com variável de controle: for

Sintaxe:
for(inicialização;condição;incremento) comando;

O comando for é de alguma maneira encontrado em todas linguagens procedurais de programação.

Em sua forma mais simples, a inicialização é um comando de atribuição que o compilador usa para estabelecer a variável de controle do loop. A condição é uma expressão de relação que testa a variável de controle do loop contra algum valor para determinar quando o loop terminará. O incremento define a maneira como a variável de controle do loop será alterada cada vez que o computador repetir o loop.

Repetição com variável de controle: for

Ex 1:

```
main()
{
    int x;
    for(x=1;x<100;x++)printf("%d\n",x);
}
```

Ex2:

```
main()
{
    int x,y;
    for (x=0,y=0;x+y<100;++x,++y)
        printf("%d ",x+y);
}
```

Repetição com variável de controle: for

Ex 3:

```
main()
{
    int linha,coluna;
    for(linha=1;linha<=24;linha++)
    {
        for(coluna=1;coluna<40;coluna++)    printf("-");
        putchar('\n');
    }
}
```

Atividades: utilizar estrutura repetição variável de controle (for)

- Crie um algoritmo que realize as seguintes atividades:

  a) Pergunte a quantidade de alunos da turma.

  b) Solicite ao usuário o nome de cada um dos X alunos.

  c) Envie cada nome lido para impressora.