

Programação em Octave

Carlos Campani

27 de setembro de 2010

1. Programa “Oi mundo”

```
printf("Oi mundo! Eu sou o Octave!\n")
```

printf saída formatada

\n nova linha

2. Usando a estrutura condicional para encontrar o maior número

```
a=input("a=");  
b=input("b=");  
if a>b  
    disp(a)  
else  
    disp(b)  
endif
```

input lê alguma coisa da entrada padrão imprimindo uma mensagem

disp mostra alguma informação na saída padrão

= atribuição

if ... endif estrutura condicional

else alternativa executada caso o teste seja falso

; para não ecoar os valores parciais calculados

3. Lendo valores e calculando expressões com operadores e funções

```
printf("Operadores e funções\n");
x=input("x=");
y=input("y=");
printf("%f + %f = %f\n",x,y,x+y);
printf("%f * %f = %f\n",x,y,x*y);
printf("%f / %f = %f\n",x,y,x/y);
printf("%f ** %f = %f\n",x,y,x**y);
printf("log %f = %f\n",x,log(x));
printf("sin %f = %f\n",x,sin(x));
```

%f especifica um número real em ponto flutuante

****** operador potência

log função logaritmo natural

sin função seno

4. Calculando a equação do 2º grau

```
printf("ax2 + bx + c\n");
a=input("a=");
b=input("b=");
c=input("c=");
if (a==0)
    printf("não é equação do segundo grau\n");
else
    delta=b*b-4*a*c;
    if (delta>=0)
        printf("x1=%f\n",(-b+sqrt(delta))/(2*a));
        printf("x2=%f\n",(-b-sqrt(delta))/(2*a));
    else
        printf("raizes complexas");
    endif
endif
```

== igualdade

sqrt função raiz quadrada

5. Calculando o número mínimo de notas para dar o troco

```
v=input("valor=");
printf("%d notas de R$ 100,00\n",floor(v/100))
v=rem(v,100);
printf("%d notas de R$ 50,00\n",floor(v/50))
v=rem(v,50);
printf("%d notas de R$ 20,00\n",floor(v/20))
v=rem(v,20);
printf("%d notas de R$ 10,00\n",floor(v/10))
v=rem(v,10);
printf("%d notas de R$ 5,00\n",floor(v/5))
v=rem(v,5);
printf("%d notas de R$ 2,00\n",floor(v/2))
printf("%d notas de R$ 1,00\n",rem(v,2))
```

v variável que armazena o valor que falta a ser pago

%d especifica um valor inteiro

floor(x) função que arredonda o número x para baixo

rem(x,y) função que devolve o resto da divisão inteira de x por y

6. Soluciona sistema de equações de duas variáveis por substituição

```
a=input("a="); b=input("b=");
c=input("c="); d=input("d=");
e=input("e="); f=input("f=");
denom=a*e-b*d;
if denom!=0
    x=(c*e-b*f)/denom;
    y=(a*f-c*d)/denom;
    printf("x=%4.2f y=%4.2f",x,y)
else
    printf("sistema sem solução")
endif
```

if denom!=0 evita a divisão por zero

%4.2f formata número real em ponto flutuante em um campo de tamanho 4 com 2 casas depois da vírgula (o ponto decimal ocupa uma posição dentro do campo)

7. Usando a estrutura de repetição para contar

```
i=1;
while i<11
    disp(i);
    i+=1;
endwhile
```

i=1 inicialização da variável i

while ... endwhile estrutura de repetição

i<11 teste que, se verdadeiro, executa os comandos da estrutura

i+=1 equivale a $i=i+1$ (chamado de *incremento*)

8. Outra forma de contar

```
i=0;
while i<10
    i+=1;
    disp(i);
endwhile
```

9. Contando de forma regressiva

```
i=10;
while i>0
    disp(i);
    i-=1;
endwhile
```

i=10 inicialização agora é com o maior valor (contagem decrescente)

i-=1 equivale a $i=i-1$ (chamado de *decremento*)

10. Fatorial

```
printf("Fatorial\n");
valor=input("n=");
n=valor;
fat=1;
while (n!=0 & n!=1)
    fat=fat*n;
    n=n-1;
endwhile
printf("Fatorial de %d é %d",valor,fat);
```

!= testa se é diferente

& conjunção (a disjunção é **|** e a negação é **~**)

%d especifica um valor inteiro

11. Obtendo os 10 primeiros termos da série de Fibonacci

```
fib=ones(1,10);
i=3;
while (i<=10)
    fib(i)=fib(i-1)+fib(i-2);
    i++;
endwhile
disp(fib)
```

ones função que devolve uma matriz $n \times m$, caso seja chamada como `ones(n,m)`, onde n é o número de linhas e m o de colunas

fib(i) acesso indexado ao vetor `fib`, devolvendo o i -ésimo elemento do vetor (primeiro elemento tem índice 1)

12. Usando for

```
x=zeros(1,10);  
x(1)=1; x(10)=1;  
for i=(1:10)  
    x(i)=x(i)+i;  
end  
disp(x)
```

zeros(n,m) função que devolve uma matriz $n \times m$ de zeros

for i=vetor ... end laço de repetição for – vetor contém os valores que serão assumidos pela variável i a cada volta do laço

(1:10) notação de dois pontos usada para criar um vetor de valores de 1 até 10 – de forma geral, $(a:b:c)$ produz um vetor de valores x tais que $a \leq x \leq c$ com passo b , por exemplo $(1:0.5:2.5)=[1,1.5,2,2.5]$

x(1), x(10), x(i) acesso indexado aos elementos do vetor x

13. Calculando a média aritmética de um vetor de valores numéricos

```
x=[3,4,2,6,7];  
media=0;  
for i=x;  
    media=media+i;  
endfor;  
media=media/length(x);  
disp(media)
```

[3,4,2,6,7] constante vetor

for i=x varia i com os elementos do vetor x

endfor finaliza o for

length(x) devolve o tamanho do vetor x

14. Usando matrizes

```
x=rand(3,2);
y=rand(2,4);
printf("x:\n")
disp(x)
printf("y:\n")
disp(y)
z=x*y;
printf("x * y:\n")
disp(z)
z=z';
printf("(x * Y)'\n")
disp(z)
```

rand(n,m) função que retorna uma matriz $n \times m$ de valores aleatórios entre 0 e 1

* operação de produto de matrizes

' transposta

15. Definindo uma função para calcular seno em graus

```
function y=f(x)
y=sin(x*pi/180);
endfunction
disp(f(45))
disp(f(90))
```

function y=f(x) cabeçalho da definição da função

pi número pi

endfunction finaliza função

f(45) chamada da função

16. Plotando funções

```
x=linspace(0,2*pi,100);  
y=sin(x);  
plot(x,y)
```

`linspace(a,b,c)` cria um vetor com espaçamento linear – a é o primeiro elemento, b o último e c o incremento

`plot` plota a função