

MC-102 — Aula 06

Comandos Repetitivos

Instituto de Computação – Unicamp

13 de Março de 2014

Roteiro

1 Revisão

2 Variável Indicadora

3 Variável Contadora

4 Outros Exemplos

5 Exercícios

Introdução

- Vimos quais são os comandos de repetição em C.
- Veremos mais alguns exemplos de sua utilização.

Comando while

- Estrutura:

```
while ( condicao )  
    comando;
```

- Ou:

```
while ( condicao ){  
    comandos;  
}
```

- Enquanto a condição for verdadeira ($\neq 0$), ele executa o(s) comando(s);

Imprimindo os 100 primeiros números inteiros

```
int i=1;
while (i<=100)
{
    printf("%d ",i);
    i++;
}
```

Comando do-while

- Estrutura:

```
do  
    comando;  
while ( condicao );
```

- Ou:

```
do{  
    comandos;  
}while ( condicao );
```

- Diferença do while: Sempre executa comandos na primeira vez. Teste condicional é feito por último.

Imprimindo os 100 primeiros números inteiros

```
int i;  
i=1;  
do{  
    printf("\n %d",i);  
    i = i+1;  
}while(i<= 100);
```

O comando for

- Estrutura:

```
for (inicio ; condicao ; passo)
    comando ;
```

- Ou:

```
for (inicio ; condicao ; passo) {
    comandos;
}
```

- Início: Uma ou mais atribuições, separadas por “,”
- Condição: Executa enquanto a condição for verdadeira.
- Passo: Um ou mais comandos, separados por “,”

Imprimindo os 100 primeiros números inteiros

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

Variável Indicadora

- Um outro uso comum de laços é para a verificação se um determinado objeto, ou conjunto de objetos, satisfazem uma propriedade ou não.
- Um padrão que pode ser útil na resolução deste tipo de problema é o uso de uma **variável indicadora**.
 - ▶ Assumimos que o objeto satisfaz a propriedade (indicadora = Verdade).
 - ▶ Com um laço verificamos se o objeto realmente satisfaz a propriedade. Se em alguma iteração descobrirmos que o objeto não satisfaz a propriedade, então fazemos (indicadora = Falso).

Exemplo: Números Primos

- Um número é primo se seus únicos divisores são 1 e ele mesmo.
- Dado um número n como detectar se este é ou não primo??
 - 1 Lê um número n .
 - 2 Testa se nenhum dos números entre 2 e $(n - 1)$ divide n .
- Lembre-se que o operador $\%$ retorna o resto da divisão.
- Portanto $(a\%b)$ é zero se e somente se b divide a .

Exemplo: Números Primos

```
Ler um número n
div = 2
indicadora = 1 //assumimos que n é primo
Enquanto div <= (n-1) faça
    Se (n%div) for igual a zero Então
        indicadora = 0 // descobrimos que n não é primo
    div = div +1
Se indicadora == 1 então o número é primo
```

Exemplo: Números Primos

Em C:

```
int main(){
    int div, n, eprimo;

    printf("Digite um numero:");
    scanf("%d",&n);

    div = 2;
    eprimo=1;
    while( (div<=n-1) && (eprimo) ){
        if(n%div == 0)
            eprimo=0;
        div++;
    }
    if(eprimo)
        printf("\nE primo!!\n");
    else
        printf("\nNao e primo!!\n");
}
```

Exemplo: Números Primos

Com o uso de **break**.

```
int main(){
    int div, n, eprimo;

    printf("\n Digite um numero:");
    scanf("%d",&n);
    div = 2;
    eprimo=1;
    while(div<=n-1){
        if(n%div == 0){
            eprimo=0;
            break;
        }
        div++;
    }
    if(eprimo)
        printf("\nE primo!!\n");
    else
        printf("\nNao e primo!!\n");
}
```

Exemplo: Números em Ordem

- Vamos fazer um programa que lê n números inteiros do teclado, e no final informa se os números lidos estão ou não em ordem crescente.
- Usaremos uma variável indicadora na resolução deste problema.

Exemplo: Números em Ordem

- Um laço principal será responsável pela leitura dos números.
- Vamos usar duas variáveis, uma que guarda o número lido na iteração atual, e uma que guarda o número lido na iteração anterior.
- Os números estarão ordenados se a condição ($\text{anterior} \leq \text{atual}$) for válida durante a leitura de todos os números.

Leia n

ordenado = 1

Leia número em anterior

Repita (n-1) vezes

 Leia número em atual

 Se $\text{atual} < \text{anterior}$

 ordenado = 0

 anterior = atual

Exemplo: Números em Ordem

Em C:

```
printf("Digite o valor de n:");
scanf("%d", &n);

scanf("%d", &anterior);
i = 1;//leu um número

ordenado = 1;
while( (i < n)  && ordenado){
    scanf("%d", &atual);
    i++;
    if(atual < anterior)
        ordenado = 0;
    anterior = atual;
}
```

Exemplo: Números em Ordem

```
#include <stdio.h>

int main(){
    int i, n, atual, anterior, ordenado;

    printf("Digite o valor de n:");
    scanf("%d", &n);

    scanf("%d", &anterior);
    i = 1;//leu um número

    ordenado = 1;
    while( (i < n) && ordenado){
        scanf("%d", &atual);
        i++;
        if(atual < anterior)
            ordenado = 0;
        anterior = atual;
    }
    if(ordenado)
        printf("Sequencia ordenada!\n");
    else
        printf("Sequencia não ordenada!\n");
}
```

Variável Contadora

- Considere ainda o uso de laços para a verificação se um determinado objeto, ou conjunto de objetos, satisfazem uma propriedade ou não.
- Um outro padrão que pode ser útil é o uso de uma **variável contadora**.
 - ▶ Esperamos que um objeto satisfaça x sub-propriedades. Usamos um laço e uma variável que **conta** o número de vezes que o objeto tem a sub-propriedade satisfeita.
 - ▶ Ao terminar o laço, se contadora for igual à x então o objeto satisfaz a propriedade.

Exemplo: Números Primos

- Um número n é primo se nenhum número de 2 até $(n - 1)$ dividí-lo.
- Podemos usar uma variável que conta quantos números dividem n .
- Se o número de divisores for 0, então n é primo.

Ler um número n

$div = 2$

$divisores = 0$ //ninguém divide n ainda

Enquanto $div \leq (n-1)$ faça

 Se $(n \% div)$ for igual a zero Então

$divisores = divisores + 1$

$div = div + 1$

Exemplo: Números Primos

```
int main(){
    int div, n, divisores;

    printf("Digite um numero:");
    scanf("%d",&n);

    div = 2;
    divisores=0;
    while(div <= n-1){
        if(n%div == 0)
            divisores++;
        div++;
    }
    if(divisores == 0)
        printf("\nE primo!!\n");
    else
        printf("\nNao e primo!!\n");
}
```

Exemplo: Números Primos

É claro que é melhor terminar o laço assim que descobrirmos algum divisor de n .

```
int main(){
    int div, n, divisores;

    printf("Digite um numero:");
    scanf("%d",&n);

    div = 2;
    divisores=0;
    while( (div <= n-1)  && (divisores == 0) ){
        if(n%div == 0)
            divisores++;
        div++;
    }
    if(divisores == 0)
        printf("\nE primo!!\n");
    else
        printf("\nNao e primo!!\n");
}
```

Outros Exemplos

- O uso de variáveis **acumuladora**, **indicadora** e **contadora** são úteis em várias situações.
- Mas não existem fórmulas para a criação de soluções para problemas.
- Em outros problemas, o uso destes padrões podem aparecer juntos, ou nem aparecer como parte da solução.

Maior Número

- Vamos fazer um programa que lê n números do teclado e informa qual foi o maior número lido.
- O programa deve ter os seguintes passos:
 - 1 Lê um número n .
 - 2 Repete n vezes a leitura de um número, determinando o maior.
- Como determinar o maior??

Maior Número

- A idéia é criarmos uma variável "maior" que sempre armazena o maior número lido até então.

Ler um número n

Ler número em maior

Repetir $n-1$ vezes

 Ler número em aux

 Se $\text{aux} > \text{maior}$ então

 Atualiza o maior

Maior Número

```
int main(){
    int cont, n, maior, aux;

    printf("\n Digite a quantidade de numeros:");
    scanf("%d",&n);

    printf("\n Digite numero:");
    scanf("%d",&maior);
    cont = 2;
    while(cont<=n){
        printf("\n Digite numero:");
        scanf("%d",&aux);
        if(aux>maior)
            maior = aux;
        cont++;
    }
    printf("\nO maior e:%d\n",maior);
}
```

Números de Fibonacci

- A série de Fibonacci é: 1, 1, 2, 3, 5, 8, 13, ...
- Ou seja o n -ésimo termo é a soma dos dois anteriores

$$F(n) = F(n - 1) + F(n - 2)$$

onde $F(1) = 1$ e $F(2) = 1$.

- Vamos fazer um programa que imprime os primeiros n números da série.

Números de Fibonacci

Ler um número inteiro n (assume que é positivo)

contador = 1

f_atual = 1, f_ant = 0

Enquanto contador $\leq n$ faça

 Imprime f_atual

 aux = f_atual

 f_atual = f_atual + f_ant

 f_ant = aux

 contador = contador + 1

Números de Fibonacci

```
int main(){
    int n, f_ant, f_atual, f_aux, cont;

    printf("\n Digite um numero:");
    scanf("%d",&n);

    cont = 1;
    f_ant=0; f_atual=1;
    while( cont<=n ){
        printf(" %d, ",f_atual);
        f_aux = f_atual;
        f_atual = f_atual + f_ant;
        f_ant = f_aux;
        cont++;
    }
    printf("\n");
}
```

Exercício

- No exemplo dos números primos não precisamos testar todos os números entre $2, \dots, (n - 1)$, para verificar se dividem ou não n . Basta testarmos até $n/2$. Por que? Qual o maior divisor possível de n ?
- Na verdade basta testarmos os números $2, \dots, \sqrt{n}$. Por que?

Exercício

- Considere o programa para determinar se uma seqüência de n números digitados pelo usuário está ordenada ou não. Refaça o programa usando uma variável contadora ao invés de indicadora.

Exercício

- Faça um programa em C que calcule o máximo divisor comum de dois números m, n . Você deve utilizar a seguinte regra do cálculo do mdc com $m \geq n$

$$\text{mdc}(m, n) = m \text{ se } n = 0$$

$$\text{mdc}(m, n) = \text{mdc}(n, m \% n) \text{ se } n > 0$$