MC-102 — Aula 07
Comandos Repetitivos

Eduardo C. Xavier

Instituto de Computação – Unicamp

28 de Março de 2017

Roteiro

1. Variável Indicadora
   - Números Primos
   - Números em Ordem
2. Variável Contadora
   - Números Primos
3. Outros Exemplos
   - Maior Número
   - Números de Fibonacci
4. Exercícios

Introdução

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

Variável Indicadora

- Um outro uso comum de laços é para verificar se um determinado objeto, ou conjunto de objetos, satisfaz 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

Problema

Determinar se um número n é primo ou não.

- 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??
  - Testar se nenhum dos números entre 2 e $(n-1)$ divide n.
- Lembre-se que o operador % retorna o resto da divisão.
- Portanto $(n\%b)$ é zero se e somente se b divide n.

Exemplo: Números Primos

```
Leia um número e salve em n
div = 2
indicadora = 1 //assumimos que n é primo
Enquanto div <= (n-1) faça
    Se (n%div) == 0 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 número:");
  scanf("%d",&n);

  div = 2;
  eprimo=1;
  while( div<=n-1 ){
    if(n%div == 0)
      eprimo=0;
    div++;
  }
  if(eprimo)
    printf("\nÉ primo!!\n");
  else
    printf("\nNão é primo!!\n");
}
```

Note que se descobrirmos que n não é primo, podemos parar o laço imediatamente.

Exemplo: Números Primos

Com término antecipado do laço:

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

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

  div = 2;
  eprimo=1;
  while( div<=n-1 && eprimo ){ //se eprimo==0 podemos sair já do laço
    if(n%div == 0)
      eprimo=0;
    div++;
  }
  if(eprimo)
    printf("\nÉ primo!!\n");
  else
    printf("\nNão é primo!!\n");
}
```

Exemplo: Números Primos

Com o uso de **break**:

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

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

Exemplo: Números em Ordem

Problema

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 (anterior <= atual) for válida durante a leitura de todos os números.

```
Leia um número e salve em n
ordenado = 1 //Assumimos que os números estão ordenados
Leia um número e salve em anterior
Repita (n-1) vezes
    Leia um número e salve em atual
    Se atual < 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

```c
#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("Sequência ordenada!\n");
  else
    printf("Sequência não ordenada!\n");
}
```

Variável Contadora

- Considere ainda o uso de laços para verificar se um determinado objeto, ou conjunto de objetos, satisfaz uma propriedade ou não.
- Um outro padrão que pode ser útil é o uso de uma **variável contadora**.
  - Esperamos que um objeto satisfaça x vezes uma sub-propriedade. 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 $(n-1)$ dividi-lo.
- Podemos usar uma variável que conta quantos números dividem n.
- Se o número de divisores for 0, então n é primo.

```
Leia um número e salve em n
div = 2
divisores = 0 //ninguém divide n ainda
Enquanto div <= (n-1) faça
    Se (n%div) == 0
        divisores = divisores + 1
    div = div + 1

Se divisores == 0 então
    Número é primo
```

Exemplo: Números Primos

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

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

   div = 2;
   divisores=0;
   while(div <= n-1){
      if(n%div == 0)
         divisores++;
      div++;
   }
   if(divisores == 0)
      printf("\nÉ primo!!\n");
   else
      printf("\nNão é 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("\nÉ primo!!\n");
  else
    printf("\nNão é 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 pode aparecer em conjunto, ou nem mesmo aparecer como parte da solução.

Maior Número

Problema

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. Leia um número e salve em n.
  2. Repita n vezes a leitura de um número determinando o maior.
- Como determinar o maior??

Maior Número

- A idéia e criar uma variável **maior** que sempre armazena o maior número lido até então.

```
Leia um número e salve em n
Leia um número e salve em maior
Repita n-1 vezes
    Leia um número e salve em aux
    Se aux > maior então
        maior = aux
```

Maior Número

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

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

   printf("\n Digite um número:");
   scanf("%d",&maior);
   cont = 1;
   while(cont<n){
      printf("\n Digite um número:");
      scanf("%d",&aux);
      if(aux>maior)
          maior = aux;
      cont++;
   }
   printf("\nO maior é:%d\n",maior);
}
```

Números de Fibonacci

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

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

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

Problema

Fazer um programa que imprime os primeiros n números da série de fibonacci.

Números de Fibonacci

```
Leia um número e salve em n
contador = 1
f_atual = 1, f_ant = 0
Enquanto contador <= n faça
        Imprima 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 número:");
   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 $2, \ldots, (n-1)$, 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 $2, \ldots, \sqrt{n}$. Por que?

Exercício

- Considere o programa para determinar se uma sequência 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 $m \geq n$

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

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