MC-102 — Aula 06
Comandos Repetitivos

Eduardo C. Xavier

Instituto de Computação – Unicamp

5 de Outubro de 2020

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 Python.
- 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 $(n-1)$ é divisor de n.
- Lembre-se que o operador % retorna o resto da divisão.
- Portanto b é divisor n se e somente se $(n\%b) == 0$.

Exemplo: Números Primos

```
Leia um número e salve em n
div = 2
indicadora = True #assumimos que n é primo
Enquanto div <= (n-1) faça
    Se (n%div) == 0 então
        indicadora = False // descobrimos que n não é primo
    div = div + 1
Se indicadora == True então
    o número é primo
```

Exemplo: Números Primos

Em Python:

```
n = int(input("Digite um número:"))
div = 2
eprimo = True
while div <= n-1 :
  if n%div == 0:
    eprimo=False
  div = div + 1

if eprimo :
  print("É primo!!");
else:
  print("Não é primo!!");
```

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

Exemplo: Números Primos

Com término antecipado do laço:

```
n = int(input("Digite um número:"))
div = 2
eprimo = True
while div <= n-1 and eprimo : #se eprimo==False podemos sair já do laço
    if n%div == 0:
      eprimo=False
    div = div + 1

if eprimo :
    print("É primo!!");
else:
    print("Não é primo!!");
```

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 = True #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 = False
    anterior = atual
```

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 = True #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 = False
    anterior = atual
```

Exemplo: Números em Ordem

Em Python:

```
n = int(input("Digite um número:"))
anterior = int(input())
i = 1 #leu um número
ordenado = True

while i < n  and ordenado :
  atual = int(input())
  i = i + 1 #leu mais um número
  if atual < anterior:
    ordenado = False
  anterior = atual

if ordenado:
  print("Sequência está ordenada")
else:
  print("Sequência não está ordenada")
```

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 $(n-1)$ for divisor de n.
- Podemos usar uma variável que conta o número de divisores de n.
- Se o número de divisores for 0, então n é primo.

```
Leia um número e salve em n
num_divisores = 0 #ninguém divide n ainda
Para div de 2 até (n-1) faça
    Se (n%div) == 0
        num_divisores = num_divisores + 1

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

Exemplo: Números Primos

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

```
Leia um número e salve em n
num_divisores = 0 #ninguém divide n ainda
Para div  de 2 até (n-1) faça
    Se (n%div) == 0
        num_divisores = num_divisores + 1

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

Exemplo: Números Primos

```
n = int(input("Digite um número:"))

num_divisores = 0;

for div in range(2,n):
    if n % div == 0:
        num_divisores = num_divisores + 1

if num_divisores == 0:
    print("É primo!!")
else:
    print("Não é primo!!")
```

Exemplo: Números Primos

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

```
n = int(input("Digite um número:"))

num_divisores = 0;

for div in range(2,n):
    if n % div == 0:
        num_divisores = num_divisores + 1
        break

if num_divisores == 0:
    print("É primo!!")
else:
    print("Não é primo!!")
```

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 associe com var. n
Leia um número e associe com var. maior
Repita n-1 vezes
    Leia um número e associe com var. aux
    Se aux > maior então
        maior = aux
```

Maior Número

```
n = int(input("Digite um número:"))

maior = int(input())

for i in range(1,n): #lê mais (n-1) números
    aux = int(input())
    if aux > maior:
        maior = aux

print("O maior número é: ", 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 associe com var. n

f1 = 0
f2 = 1
Repita n vezes:
        Imprima f2
        aux = f2
        f2 = f2 + f1
        f1 = aux
```

Números de Fibonacci

```
n = int(input("Digite um número:"))

f1 = 0
f2 = 1
for i in range(n):
    print(f2,', ',end='')
    aux = f2
    f2 = f2 + f1
    f1 = aux
print()
```

Exercício

- Refaça o programa para imprimir os n primeiros números da série de Fibonacci, mas sem o uso de uma variável auxiliar como **aux**.

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 $m \geq n$

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

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