Programando em Java
(Ciclos e Recursividade)

Mestrado Integrado em Engenharia Informática FCT UNL
http://ctp.di.fct.unl.pt/miei/ip/

Corpo Docente 2020/2021
António Ravara, Artur Miguel Dias, Bernardo Toninho,
Ema Vieira, Inês Fernandes, Margarida Mamede,
Miguel Monteiro, Rui Nóbrega

Problema do Somatório dos Quadrados

Dado um número inteiro positivo, **n**, calcular o somatório:

$$\sum_{i=1}^{n} i^2$$

- (Input) Uma linha com um número inteiro positivo (**n**).
- (Output) O programa escreve uma linha com o resultado.
- (Exemplo)

Input	Output
5	55

Tarefas:

1. Resolva o problema iterativamente.
2. Resolva o problema recursivamente.

Problema do Somatório dos Quadrados

Cálculo do somatório (versão iterativa):

$$S(n) = \sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + n^2$$

Como calcular a soma? Há muitas formas (dos extremos para o centro, do fim para o príncipio, …).

A forma "mais natural" (indicada no somatório), é começar com o i a 1 e ir somando até n. Precisamos de ir acumulando as somas até obter o resultado final.

Exemplo (n = 5):

```
soma = 0;
soma = soma + 1²;   // soma tem (0) + 1²
soma = soma + 2²;   // soma tem (1²) + 2²
soma = soma + 3²;   // soma tem (1²+2²) + 3²
soma = soma + 4²;   // soma tem (1²+2²+3²) + 4²
soma = soma + 5²;   // soma tem (1²+2²+3²+4²) + 5²
```

Problema do Somatório dos Quadrados

Cálculo do somatório (versão iterativa):

$$S(n) = \sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + n^2$$

Exemplo (n = 5):

```
soma = 0;
soma = soma + 1²;  // soma tem (0) + 1²
soma = soma + 2²;  // soma tem (1²) + 2²
soma = soma + 3²;  // soma tem (1²+2²) + 3²
soma = soma + 4²;  // soma tem (1²+2²+3²) + 4²
soma = soma + 5²;  // soma tem (1²+2²+3²+4²) + 5²
```

Estratégia:

- começar com o i a 1 e ir somando até i ser n;
- precisamos de ir *acumulando* as somas até obter o resultado final.

Ou seja, para i de 1 até n, soma-se i^2 ao acumulador.

Problema do Somatório dos Quadrados

Cálculo do somatório (ideia geral da versão recursiva):

$$S(n) = \sum_{i=1}^{n} i^2 = \left(\sum_{i=1}^{n-1} i^2\right) + n^2 = S(n-1) + n^2$$

Definição matemática (versão recursiva):

$$S(n) = \begin{cases} 1, & \text{se } n = 1; \\ S(n-1) + n^2, & \text{se } n \geq 2. \end{cases}$$

Problema do Somatório dos Quadrados

Cálculo do somatório (versão recursiva):

$$S(n) = \begin{cases} 1, & \text{se } n = 1; \\ S(n-1) + n^2, & \text{se } n \geq 2. \end{cases}$$

Exemplo (n = 5):

S(5) = S(4) + 5^2;
S(4) = S(3) + 4^2;
S(3) = S(2) + 3^2;
S(2) = S(1) + 2^2;
S(1) = 1;

então: S(5) é 30+25 = 55
então: S(4) é 14+16 = 30
então: S(3) é 5+9 = 14
então: S(2) é 1+4 = 5

Problema do Máximo Divisor Comum

Dados dois números inteiros positivos, **a** e **b**, calcular:

mdc(**a**, **b**)

- (Input) Uma linha com dois números inteiros positivos (**a** e **b**).
- (Output) O programa escreve uma linha com o resultado.
- (Exemplo)

Input	Output
`12 90`	`6`

Tarefas:

1. Recorde a resolução recursiva do problema.
2. Resolva o problema iterativamente.

Problema do Máximo Divisor Comum

Algoritmo de Euclides (recursivo) para calcular mdc(**a**, **b**):

$$\mathrm{mdc}(a,b) = \begin{cases} a, & \text{se } b = 0; \\ \mathrm{mdc}(b, a\%b), & \text{se } b > 0. \end{cases}$$

```java
// Greatest common divisor by Euclid's algorithm
// Pre: a >= 0 && b >= 0 && ( a!=0 || b!=0 )
public static int gcdRec( int a, int b ) {
   if ( b == 0 )
      return a;
   else
      return gcdRec(b, a%b);
}
```

Problema do Máximo Divisor Comum

Algoritmo de Euclides (recursivo) para calcular mdc(**a**, **b**):

$$\text{mdc}(a,b) = \begin{cases} a, & \text{se } b = 0; \\ \text{mdc}(b, a\%b), & \text{se } b > 0. \end{cases}$$

Exemplo (**a** = 12 e **b** = 90):

```
mdc(12,90)  =  mdc(90,12%90)
mdc(90,12)  =  mdc(12,90%12)
mdc(12,6)   =  mdc(6,0)
mdc(6,0)    =  6
```

Nota: `12%90 = 12`
Nota: `90%12 = 6`
Nota: `12%6 = 0`

Algoritmo de Euclides (iterativo) para calcular mdc(**a**, **b**):

Enquanto **b** > 0, os argumentos mudam (simultaneamente):

(novo **a**) ⬅️ (anterior **b**)

(novo **b**) ⬅️ resto da divisão de (anterior **a**) por (anterior **b**)

(Quando **b** = 0) o resultado é **a**

Problema CSI Portugal (versão original)

Escreva um programa que analise vários cenários.

- (Input) A primeira linha tem um inteiro, **C**, que representa o número de cenários. Por cada cenário, há duas linhas consecutivas. (...)
- (Output) Por cada cenário, há uma linha com: "`Com alibi`", se a atividade do suspeito decorreu durante todo o intervalo estimado para o crime; "`Sem alibi`", nos restantes casos.
- (Exemplo)

Input

```
2
14 15
9 16
14 17
9 16
```

Output

```
Com alibi
Sem alibi
```

Tarefa: Resolva a versão original do problema CSI Portugal.

Classe Main para o Exercício CSI

```
import java.util.Scanner;
public class Main {
    public static void main( String[] args ) {
        Scanner input = new Scanner(System.in);
        int start = input.nextInt();
        int end = input.nextInt();
        input.nextLine();
        Interval crime = new Interval(start, end);
        start = input.nextInt();
        end = input.nextInt();
        input.nextLine();
        Interval activity = new Interval(start, end);
        input.close();
        if ( activity.contains(crime) )
            System.out.println("Com alibi");
        else
            System.out.println("Sem alibi");
    }
}
```

Classe disponibilizada para o exercício da 2ª aula prática.

Estará bem feita?

Se não estiver, altere-a antes de resolver a versão original do CSI.