Processamento da Informação

Estruturas de repetição: “while”

Prof. Jesús P. Mena-Chalco
CMCC/UFABC

Q2/2018

Soma de inteiros em um intervalo

Soma de inteiros em um intervalo: $[n_1, n_2]$

```java
import java.util.Scanner;
class Soma
{

  public static void main(String[] args) {
    Scanner entrada = new Scanner(System.in);
    int n1 = entrada.nextInt();
    int n2 = entrada.nextInt();

    int soma = 0;
    int k    = n1;

    while (k <= n2) {
      soma = soma+k;
      k    = k+1;
    }

    System.out.printf("Soma entre %d e %d: %d\n", n1, n2, soma);
  }

}
```

Permite a leitura de 2 números inteiros.

```
Interactions | Console | Compiler Output
Welcome to DrJava.  Working directory is /home/jmenac/PI-laboratorio-04
> run Soma
  44
  11
Soma entre 44 e 11: 0
>
```

Soma de inteiros em um intervalo: $[n_1, n_2]$

```java
import java.util.Scanner;
class Soma
{

  static int FS(int a, int b) {
    int soma = 0;
    int k    = a;

    while (k <= b) {
      soma = soma+k;
      k    = k+1;
    }

    return soma;
  }

  public static void main(String[] args) {
    Scanner entrada = new Scanner(System.in);
    int n1 = entrada.nextInt();
    int n2 = entrada.nextInt();

    System.out.printf("Soma entre %d e %d: %d\n", n1, n2, FS(n1, n2) );
  }

}
```

Chamado à função FS

```
Interactions | Console | Compiler Output
Welcome to DrJava.  Working directory is /home/jmenac/PI-laboratorio-04
> run Soma
10
13
Soma entre 10 e 13: 46
>
```

(outra) aproximação de Pi

Procure na internet por “Theoretical Computer Science Cheat Sheet”

Pág. 6.

Aproximação de Pi

Existem várias formas para obter uma aproximaçao de PI, por exemplo:

Schneider $\pi \approx \sqrt{7 + \sqrt{6 + \sqrt{5}}}$ 3 digitos

Irvine $\pi \approx \sqrt{\sqrt{3^4 + \frac{19^2}{78 - 56}}}$ 8 digitos

Euler

$$\frac{\pi^2}{6} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \cdots$$

$$\frac{\pi^2}{8} = \frac{1}{1^2} + \frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \frac{1}{9^2} + \cdots$$

$$\frac{\pi^2}{12} = \frac{1}{1^2} - \frac{1}{2^2} + \frac{1}{3^2} - \frac{1}{4^2} + \frac{1}{5^2} - \cdots$$

Aproximação de Pi

$$\frac{\pi^2}{6} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \cdots$$

```java
import java.util.Scanner;
import java.math.*;

class PiEuler
{

  static int Pi(int termos) {
    double soma = 0;
    int i       = 1;

    while (i <= termos) {
      soma = soma + 1.0/i*i;
      i    = i+1;
    }

    return Math.sqrt(soma*6);
  }

  public static void main(String[] args) {
    Scanner entrada = new Scanner(System.in);
    int termos = entrada.nextInt();

    System.out.printf("Pi = %.20f\n", Pi(termos) );
  }

}
```

Interactions | Console | Compiler Output

```
Welcome to DrJava.  Working directory is /home/jmenac/PI-laboratorio-04
> run PiEuler
1000
Pi = 3.14063805620599460000
>
```

Quais são os erros nesta solução?

Aproximação de Pi

$$\frac{\pi^2}{6} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \cdots$$

```
 1 import java.util.Scanner;
 2 import java.math.*;
 3
 4 class PiEuler
 5 {
 6
 7   static double Pi(int termos) {
 8     double soma = 0;
 9     int i       = 1;
10
11     while (i <= termos) {
12       soma = soma + 1.0/(i*i);
13       i    = i+1;
14     }
15
16     return Math.sqrt(soma*6);
17   }
18
19   public static void main(String[] args) {
20     Scanner entrada = new Scanner(System.in);
21     int termos = entrada.nextInt();
22
23     System.out.printf("Pi = %.20f\n", Pi(termos) );
24   }
25
26 }
27
28
29
```

Interactions | Console | Compiler Output

```
Welcome to DrJava.  Working directory is /home/jmenac/PI-laboratorio-04
> run PiEuler
1000
Pi = 3.14063805620599460000
>
```

Outra aproximação de Pi

Uma outra variação da serie e Euler para aproximar a Pi.

$$\frac{\pi^2}{12} = \frac{1}{1^2} - \frac{1}{2^2} + \frac{1}{3^2} - \frac{1}{4^2} + \frac{1}{5^2} - \cdots$$

Essa nova somatória permite obter uma melhor aproximação de Pi.

Implemente essa nova somatoria.

Outra aproximação de Pi

$$\frac{\pi^2}{12} = \frac{1}{1^2} - \frac{1}{2^2} + \frac{1}{3^2} - \frac{1}{4^2} + \frac{1}{5^2} - \cdots$$

```
 1 import java.util.Scanner;
 2 import java.math.*;
 3
 4 class PiEuler
 5 {
 6
 7   static double Pi(int termos) {
 8     double soma = 0;
 9
10
11
12
13
14
15
16
17
18     return Math.sqrt(soma*12);
19   }
20
21   public static void main(String[] args) {
22     Scanner entrada = new Scanner(System.in);
23     int termos = entrada.nextInt();
24
25     System.out.printf("Pi = %.20f\n", Pi(termos) );
26   }
27
28 }
29
```

Interactions | Console | Compiler Output

```
Welcome to DrJava.  Working directory is /home/jmenac/PI-laboratorio-04
> run PiEuler
1000
Pi = 3.14159169961491600000
>
```

Outra aproximação de Pi

$$\frac{\pi^2}{12} = \frac{1}{1^2} - \frac{1}{2^2} + \frac{1}{3^2} - \frac{1}{4^2} + \frac{1}{5^2} - \cdots$$

```
 1 import java.util.Scanner;
 2 import java.math.*;
 3
 4 class PiEuler
 5 {
 6
 7   static double Pi(int termos) {
 8     double soma = 0;
 9     int i       = 1;
10     int sinal   = 1;
11
12     while (i <= termos) {
13       soma  = soma + (sinal)*1.0/(i*i);
14       i     = i+1;
15       sinal = sinal*-1;
16     }
17
18     return Math.sqrt(soma*12);
19   }
20
21   public static void main(String[] args) {
22     Scanner entrada = new Scanner(System.in);
23     int termos = entrada.nextInt();
24
25     System.out.printf("Pi = %.20f\n", Pi(termos) );
26   }
27
28 }
29
```

Interactions | Console | Compiler Output

```
Welcome to DrJava.  Working directory is /home/jmenac/PI-laboratorio-04
> run PiEuler
1000
Pi = 3.14159169961491600000
>
```

Aproximação de *e*

Procure na internet por “Theoretical Computer Science Cheat Sheet”

Pág. 3.

Aproximação de *e*

O número de Euler $$e = \sum_{n=0}^{\infty} \frac{1}{n!} = 2.718281828459045\ldots$$

$$e = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \cdots$$

Implemente essa nova somatoria mas considerando todos os termos menores ou iguais a 0.0000001

Desafio:
Quantos termos são menores ou iguais a 0.0000001?

Aproximação de e

```java
import java.util.Scanner;
import java.math.*;

class Euler
{

  static double E() {

  }

  public static void main(String[] args) {
    Scanner entrada = new Scanner(System.in);

    System.out.printf("E = %.20f\n", E() );
  }

}
```

Interactions | Console | Compiler Output

```
Welcome to DrJava.  Working directory is /home/jmenac/PI-laboratorio-04
> run Euler
E = 2.71828180114638450000
>
```

2.71828182845904523
53602874713526624 9
77572470936999595 7
49669676277240766 3
03535475945713821 7
85251664274274663 9
19320030599218174 1

Aproximação de e

```java
import java.util.Scanner;
import java.math.*;

class Euler
{

  static double E() {
    double soma = 1;
    int i       = 1;
    int d       = 1;

    while (1.0/d >= 0.0000001) {
      soma = soma + 1.0/d;
      i    = i+1;
      d    = d*i;
    }

    return soma;
  }

  public static void main(String[] args) {
    Scanner entrada = new Scanner(System.in);

    System.out.printf("E = %.20f\n", E() );
  }

}
```

Interactions | Console | Compiler Output

```
Welcome to DrJava.  Working directory is /home/jmenac/PI-laboratorio-04
> run Euler
E = 2.71828180114638450000
>
```

11 termos?

2.71828182845904523
53602874713526624
97757247093699959
57496696762772407
66303535475945713
82178525166427427
46639193200305992
181741

Prática no URI: Exercício 1059

URI – Exercício - 1059

Números Pares

Adaptado por Neilor Tonin, URI Brasil

Timelimit: 1

Faça um programa que mostre os números pares entre 1 e 100, inclusive.

Entrada

Neste problema extremamente simples de repetição não há entrada.

Saída

Imprima todos os números pares entre 1 e 100, inclusive se for o caso, um em cada linha.

Exemplo de Entrada	Exemplo de Saída
	2 4 6 ... 100

URI – Exercício - 1059

```
class Problema1059
{

  public static void main(String[] args) {
    int k = 1;

    while (k <= 100) {
      if (k%2==0) {
          System.out.printf("%d\n", k);
      }
      k = k+1;
    }

  }

}
```

Essa é a melhor solução?

URI – Exercício - 1059

```java
class Problema1059
{

  public static void main(String[] args) {
    int k = 1;

    while (k <= 100) {
      if (k%2==0) {
          System.out.printf("%d\n", k);
      }
      k = k+1;
    }

  }

}
```

```java
class Problema1059
{

  public static void main(String[] args) {
    int k = 2;

    while (k <= 100) {
      System.out.printf("%d\n", k);
      k = k+2;
    }

  }

}
```

Não precisa verifcar se o valor é par!
Veja o incremento em k (k+2)