**Otávio Vinícius Guimarães Silveira Rocha**

Engenharia de Software – 2º Período

Lista 5 – Laboratórios de Computação II

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 – Faça uma pesquisa sobre formas de calcular tempo em Java (pesquisa por stopwatch,duration, etc)

[...]

2 – Implemente 2 funções para calcular o fatorial de um número inteiro: uma iterativa e outra recursiva (com impressão dos valores e sem impressão dos valores). Qual o maior valor para o qual você conseguiu calcular o fatorial em cada versão? Quanto tempo levou para calcular o maior fatorial?

**public** **static** **long** fatorialIteSemImpressao(**int** num) {

**long** fat = 1;

**for** (**int** i = 2; i <= num; i++) {

fat \*= i;

}

**return** fat;

}

**private** **static** **long** calcTempoIteSemImpressao(**int** n) {

**long** start = System.*nanoTime*();

*fatorialIteSemImpressao*(n);

**long** finish = System.*nanoTime*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**public** **static** **long** fatorialIteComImpressao(**int** num) {

**long** fat = 1;

**for** (**int** i = 2; i <= num; i++) {

fat \*= i;

}

**return** fat;

}

**private** **static** **long** calcTempoIteComImpressao(**int** n) {

**long** start = System.*nanoTime*();

System.***out***.print("\nFatorial de " + n + " é " + *fatorialIteComImpressao*(n));

**long** finish = System.*nanoTime*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**public** **static** **long** fatorialRecSemImpressao(**int** num) {

**long** fat = 0;

**if** (num <= 1) {

**return** fat = 1;

} **else** {

**return** fat = (num - 1) \* num;

}

}

**private** **static** **long** calcTempoRecSemImpressao(**int** n) {

**long** start = System.*nanoTime*();

*fatorialRecSemImpressao*(n);

**long** finish = System.*nanoTime*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**private** **static** **long** calcTempoRecComImpressao(**int** n) {

**long** start = System.*nanoTime*();

System.***out***.print("\nFatorial de " + n + " é " + *fatorialRecComImpressao*(n));

**long** finish = System.*nanoTime*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**private** **static** **long** fatorialRecComImpressao (**long** n) {

**if** (n==0) {

**return** 1;

} **else** {

**return** n \* *fatorialRecComImpressao* (n-1);

}

}

**public** **static** **void** main(String[] args) {

**for** (**int** i = 0; i <= 39; i++) {

System.***out***.println("========================================================");

System.***out***.println("Iterativo Sem Impressao: (Tempo de execução : " + *calcTempoIteSemImpressao*(i) + " nano segundos)");

System.***out***.println("\nIterativo Com Impressao: (Tempo de execução : " + *calcTempoIteComImpressao*(i) + " nano segundos)");

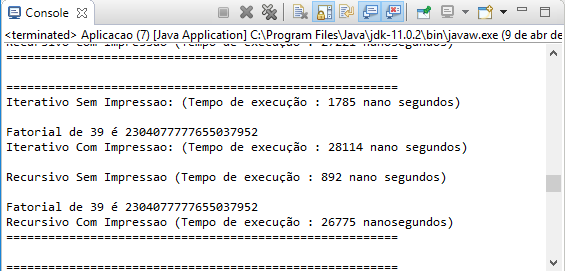
System.***out***.println("\nRecursivo Sem Impressao (Tempo de execução : " + *calcTempoRecSemImpressao*(i) + " nano segundos)");

System.***out***.println("\nRecursivo Com Impressao (Tempo de execução : " + *calcTempoRecComImpressao*(i) + " nanosegundos)");

System.***out***.println("========================================================");

System.***out***.println();

}

}

**Qual o maior valor para o qual você conseguiu calcular o fatorial em cada versão?** R: Fatorial do inteiro 39

**Quanto tempo levou para calcular o maior fatorial?**

* Interativo sem impressão: 1785 nano segundos;
* Interativo com impressão: 28114 nano segundos;
* Recursivo sem Impressão: 892 nano segundos;
* Recursivo com impressão: 26775 nano segundos;

3–Implemente 2 funções para calcular o enésimo termo da série de Fibonacci: uma iterativa e outra recursiva (com impressão dos valores e sem impressão dos valores). Qual foi o maior termo calculado? Aconteceu algum erro? Quanto tempo levou para calcular o maior termo da série?

**public** **static** **long** fibonacciIterativo(**int** n) {

**int** F = 0, ant = 0;

**for** (**int** i = 1; i <= n; i++) {

**if** (i == 1) {

F = 1;

ant = 0;

} **else** {

F += ant;

ant = F - ant;

}

}

**return** F;

}

**private** **static** **long** calcTempoIteSemImpressao(**int** n) {

**long** start = System.*currentTimeMillis*();

*fibonacciIterativo*(n);

**long** finish = System.*currentTimeMillis*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**private** **static** **long** calcTempoIteComImpressao(**int** n) {

**long** start = System.*currentTimeMillis*();

System.***out***.print("Fibonacci (" + n + "): " + *fibonacciIterativo*(n));

**long** finish = System.*currentTimeMillis*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**public** **static** **long** fibonacciRecursivo(**int** n) {

**if** (n < 2) {

**return** n;

} **else** {

**return** *fibonacciRecursivo*(n - 1) + *fibonacciRecursivo*(n - 2);

}

}

**private** **static** **long** calcTempoRecSemImpressao(**int** n) {

**long** start = System.*currentTimeMillis*();

*fibonacciRecursivo*(n);

**long** finish = System.*currentTimeMillis*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**private** **static** **long** calcTempoRecComImpressao(**int** n) {

**long** start = System.*currentTimeMillis*();

System.***out***.print("Fibonacci: (" + n + ") " + *fibonacciRecursivo*(n));

**long** finish = System.*currentTimeMillis*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**public** **static** **void** main(String[] args) {

**for** (**int** i = 0; i <= 47; i++) {

System.***out***.println("========================================================");

System.***out***.println("[Fib Iterativo Sem Impressão] = Tempo Execução: " + *calcTempoIteSemImpressao*(i) + "ms\n");

System.***out***.print(" || Tempo Execução: " + *calcTempoIteComImpressao*(i) + "ms\n");

System.***out***.println("\n[Fib Recursivo Sem Impressão] = Tempo Execução: " + *calcTempoRecSemImpressao*(i) + "ms\n");

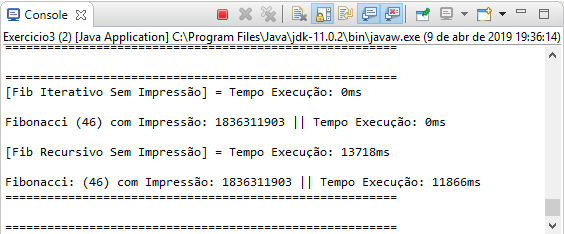
System.***out***.print(" || Tempo Execução: " + *calcTempoRecComImpressao*(i) + "ms\n");

System.***out***.println("========================================================");

System.***out***.println();

}

}



**Qual foi o maior termo calculado?** Inteiro número 46

**Aconteceu algum erro?** Não

**Quanto tempo levou para calcular o maior termo da série?**

* Fibonacci iterativo sem impressão: 0 milissegundos
* Fibonacci iterativo com impressão: 0 milissegundos
* Fibonacci recursivo sem impressão: 13718 milissegundos
* Fibonacci recursivo com impressão: 11866 milissegundos

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

4 – Implemente 2 funções para calcular a multiplicação entre dois números através de somas sucessivas: uma iterativa e outra recursiva (com impressão dos valores e sem impressão dos valores). Para testar, calcule 3 x 715.827.881 e mostre o tempo para ter o resultado

**public** **static** **long** multiplicadorIterativo(**long** a, **long** b) {

**long** total = 0;

**for** (**int** i = 0; i < b; i++) {

total += a;

}

**return** total;

}

**private** **static** **long** calcTempoIteSemImpressao(**long** a, **long** b) {

**long** start = System.*currentTimeMillis*();

*multiplicadorIterativo*(a, b);

**long** finish = System.*currentTimeMillis*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**private** **static** **long** calcTempoIteComImpressao(**long** a, **long** b) {

**long** start = System.*currentTimeMillis*();

System.***out***.println("Resultado: " + *multiplicadorIterativo*(a, b));

**long** finish = System.*currentTimeMillis*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**static** **long** multiplicadorRecursivo(**long** a, **long** b) {

**if** (b <= 0) {

b = -b;

a = -a;

**if** (b == 0) {

**return** b;

}

}

**if** (b > 1) {

a += *multiplicadorRecursivo*(a, --b);

}

**return** a;

}

**static** **long** multiplica(**long** x, **long** y) {

**return** (x > 0 && y > 1) ? x += *multiplica*(x, --y) : (y==0) ? 0 : x;

}

**private** **static** **long** calcTempoRecComImpressao(**long** a, **long** b) {

**long** start = System.*currentTimeMillis*();

System.***out***.println("Resultado: " + *multiplica*(a, b));

**long** finish = System.*currentTimeMillis*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**private** **static** **long** calcTempoRecSemImpressao(**long** a, **long** b) {

**long** start = System.*currentTimeMillis*();

*multiplica*(a, b);

**long** finish = System.*currentTimeMillis*();

**long** timeElapsed = finish - start;

**return** timeElapsed;

}

**public** **static** **void** main(String[] args) {

**long** value = 3;

**long** multiplicador = 715827881;

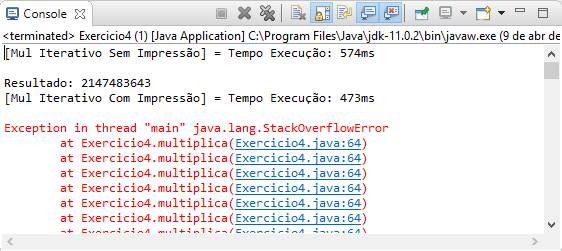
System.***out***.println("[Mul Iterativo Sem Impressão] = Tempo Execução: " + *calcTempoIteSemImpressao*(value, multiplicador) + "ms\n");

System.***out***.println("[Mul Iterativo Com Impressão] = Tempo Execução: " + *calcTempoIteComImpressao*(value, multiplicador) + "ms\n");

System.***out***.println("[Mul Recursivo Sem Impressão] = Tempo Execução: " + *calcTempoRecSemImpressao*(value, multiplicador) + "ms\n");

System.***out***.println("[Mul Recursivo Com Impressão] = Tempo Execução: " + *calcTempoRecComImpressao*(value, multiplicador) + "ms\n");

}

****

Nas multiplicações iterativas, o tempo de execução foi: 574 milissegundos para o método sem impressão e 473 milissegundos com impressão.

Não foi possível calcular a multiplicação **3 x 715.827.881** no método recursivo, pois ocorre o erro *java.lang.StackOverflowError*. Creio que houve um estouro na capacidade suportada pela pilha, por se tratar de uma multiplicação muito grande, a quantidade de processos empilhados seriam enormes.