

Trabalho de recursividade

Depuração do código recursivo

Nesse documento iremos abordar dois fluxos de depuração do nosso código, em que no primeiro deles iremos utilizar somente a opção de depuração “Step over” e no segundo fluxo iremos utilizar somente a opção “Step into”, exceto em métodos de classes nativas do Java, como o método “parseInt” da classe Integer.

Com a finalidade de otimização, primeiro iremos descrever o fluxo de depuração utilizando STEP OVER, pois uma boa parte do fluxo vai se repetir, ao encontrarmos fluxos que só serão executados utilizando a opção STEP INTO, iremos fazer uma referência ao tópico referente ao fluxo encontrado.

Fluxo de depuração padrão - STEP OVER

Linha 1

Início do método principal do programa

Linhas 2 e 4

Print no console das mensagens que indicam, respectivamente, o início do programa, e a solicitação para o usuário digitar o tamanho da lista que ele deseja preencher.

Linha 5

Nessa linha a execução do programa para pois está aguardando uma entrada de dados pelo usuário no terminal, ao usuário realizar a entrada de dados, salvamos em uma variável com o nome prompt.

Linha 7

Abertura do bloco try para tratarmos posteriormente a exceção de entrada de dados inválida.

Linha 8

Utilizamos a classe Integer para parsear a entrada do usuário de String para o tipo inteiro, nessa linha é possível ser lançada a exceção “NumberFormatException”, caso a String que armazenamos contenha caracteres não numéricos, com exceção do sinal de “-” no início.

Linha 9

Verificamos se o número que foi parseado é negativo, pois não é possível criar um array de tamanho negativo. Caso seja verdadeiro, lançamos a exceção “NumberFormatException” para ambas receberem a mesma tratativa posteriormente.

Linha 11

Declaramos o array de será preenchido pelo usuário, com o tamanho informado.

Linha 12

Nessa linha é chamado o método recursivo para fazer o preenchimento do array, onde informamos o array que será preenchido e o índice da primeira inserção.

Fluxo com STEP OVER - [Linha 24](#).

Fluxo de depuração STEP INTO

Da linha 1 até a linha 12 do main, o comportamento é idêntico ao fluxo descrito no STEP OVER

A partir da linha 12, passamos a utilizar STEP INTO para acompanhar as chamadas recursivas. que chama `insertUserInputOnArray(array, 0);`

Linha 24

Entramos no método com `index = 0`

Linha 25

Verifica `index == array.length` → falso (`0 != tamanho do array`)

Linha 27–30

Entra no try, exibe a mensagem pedindo o valor da posição 1, lê a entrada do usuário, faz o `parseInt` e armazena em `array[0]`

Linha 35

Chama-se recursivamente `insertUserInputOnArray(array, 1)` e usamos novamente STEP INTO.

Para `index = 1` e depois `index = 2`, a sequência é a mesma

Linha 25

irá dar falso, depois lê o valor, faz o parse e armazena em `array[index]`, e depois chama recursivamente com `index + 1`

Quando a chamada chega com `index = 3` (tamanho do array)

Linha 25

A condição `index == array.length` é verdadeira, o método executa o `return`; e não entra no try

Depois disso tudo a recursão vai desempilhando a pilha voltando para o index = 2 depois index = 1 e depois index = 0 , finalizando na main.