Exampe Especial
Algoritmos e Estruturas de Dados I

Prof.: Carlos Camarão

5 de Janeiro de 2001

Em todas as questões, você pode supor a existência de métodos estáticos de entrada e saída de valores de tipos bsicos, definidos em uma classe `EntradaPadrão`. Por exemplo, para ler inteiros, você pode usar `EntradaPadrão.lêInteiro()`.

1. Escreva um programa que leia um valor inteiro positivo n, calcule o valor dado por $\sum_{i=1}^{n}(-(i+1)/(i*i))$, e imprima esse valor. Decomponha o seu programa em três partes, cada uma delas implementada por um método: leitura, cálculo do valor desejado a partir do valor lido, e impressão do valor calculado.

   O método de leitura deve ler uma cadeia de caracteres e causar uma exceção se essa cadeia não representar um número inteiro positivo. Se a exceção ocorrer, ela deve ser tratada no programa principal. Esse tratamento deve apenas emitir uma mensagem de erro apropriada, e em seguida a execução do programa deve ser interrompida.

2. Faça um programa para ler um valor inteiro positivo n, e em seguida n seqüências de valores inteiros positivos, cada uma delas terminada por um valor inteiro negativo ou nulo, e imprimir a soma dos maiores valores de cada uma das seqüências.

   Por exemplo, se os valores digitados pelo usuário forem:

   $$\begin{array}{l} 5 \\ 2 \\ 10 \\ 0 \\ 2 \\ 18 \\ -1 \\ 3 \\ 30 \\ 5 \\ 1 \\ 0 \end{array}$$

   o programa deve imprimir como resultado o valor 76 (pois 76 = (5+2+10) + (2+18) + (3+30+5+1)).

   Em cada situação de entrada de dados, se um valor não esperado for fornecido com entrada (por exemplo uma cadeia de caracteres que não representa um número

inteiro), uma exceção deve ser causada, e o processo de entrada de dados pelo usuário do programa deve recomeçar, a partir do início (isto é, a partir da leitura de n).

Dica 1: o método *parseInt* causa uma exceção (*NumberFormatException*) se a cadeia de caracteres fornecida como argumento não representar um número inteiro.

Dica 2: Não esqueça de testar se o primeiro valor lido é um valor positivo, e causar uma exceção em caso contrário. Você pode causar e tratar apenas a exceção *Exception*, pois a exceção *NumberFormatException* é um objeto da classe *Exception*, e portanto será também tratada por um tratador que trata (recebe como argumento) uma exceção *NumberFormatException*.

3. Escreva outra versão para o seu programa acima, que difere da anterior da seguinte maneira. Se o seu programa declarou e usou um arranjo, faça uma versão que não declara e nem usa um arranjo. E vice-versa (se o seu programa não declarou e usou um arranjo, faça uma versão que declara e usa um arranjo).

4. (7 pontos) Uma operação de *contração de uma lista* é definida como sendo tal que, dados um inteiro positivo i, chamado de *índice* da contração, e uma lista de n números inteiros $x = [a_1, a_2, \ldots, a_i, \ldots, a_n]$, onde supõe-se $i < n$, retorna como resultado a lista $[a_1, \ldots, a_{i-1}, a_i - a_{i+1}, a_{i+2}, \ldots, a_n]$.

   Aplicando $n - 1$ contrações a uma lista de n inteiros quaisquer obtém-se uma lista com um único inteiro, chamado de *valor alvo*. Por exemplo, se chamarmos de *con* a operação de contração e por $con(x, i)$ a operação de aplicar *con* a uma lista x e a um índice i, temos:

$$\begin{aligned} con([12, 10, 4, 3, 5], 2) &= [12, 6, 3, 5] \\ con([12, 6, 3, 5], 3) &= [12, 6, -2] \\ con([12, 6, -2], 2) &= [12, 8] \\ con([12, 8], 1) &= [4] \end{aligned}$$

   Faça um programa que:

   - leia um valor inteiro positivo n (o número de inteiros da seqüência original), n valores inteiros de uma seqüência $a_1, \ldots, a_n$ e, em seguida, uma seqüência de $n - 1$ valores que indicam os índices de constrações,
   - realize contrações consecutivamente como no exemplo acima, e
   - imprima o valor alvo x.

   Se não existir valor alvo (i.e. se a entrada estiver de alguma forma incorreta), o programa deve emitir uma mensagem de erro apropriada.

   Dica: Use um contador para indicar o número de valores válidos para a seqüência, inicialmente igual a n (tamanho da seqüência original) e decrementado de 1 a cada contração.