|  |  |
| --- | --- |
| Logotipo, nome da empresa  Descrição gerada automaticamente | Campus de Cascavel  Centro de Ciências Exatas e Tecnológicas - CCET  Curso de Ciência da Computação  Disciplina: Estruturas de Dados  Professor: Josué Castro |

**1ª Avaliação (28 de 04 de 2023)**

**Questão 1 ( Linguagem à escolha (C++ ou Go), 10 pts.):**

Vó Vitória mantém, desde o nascimento dos netos Joãozinho e Zezinho, um ritual que faz a alegria dos meninos. Ela guarda todas as moedas recebidas como troco em dois pequenos cofrinhos, um para cada neto. Quando um dos cofrinhos fica cheio, ela chama os dois netos para um alegre almoço, ao final do qual entrega aos garotos as moedas guardadas nos cofrinhos de cada um.

Ela sempre foi muito zelosa quanto à distribuição igualitária do troco arrecadado. Quando, por força do valor das moedas, ela não consegue depositar a mesma quantia nos dois cofrinhos, ela memoriza a diferença de forma a compensá-la no próximo depósito.

**Tarefa:**

Vó Vitória está ficando velha e tem medo que deslizes de memória a façam cometer injustiças com os netos, deixando de compensar as diferenças entre os cofrinhos. Sua tarefa é ajudar Vó Vitória, escrevendo um programa de computador que indique as diferenças entre os depósitos, de forma que ela não tenha que preocupar-se em memorizá-las.

**Entradas:**

A entrada é composta de vários conjuntos de teste. A primeira linha de um conjunto de teste contém um número inteiro N, que indica o número de depósitos nos cofrinhos. As N linhas seguintes descrevem cada uma um depósito nos cofrinhos; o depósito é indicado por dois valores inteiros J e Z, separados por um espaço em branco, representando respectivamente os valores, em centavos, depositados nos cofres de Joãozinho e Zezinho. O final da entrada é indicado por N = 0.

**Saídas:**

Para cada conjunto de teste da entrada seu programa deve produzir um conjunto de linhas na saída. A primeira linha deve conter um identificador do conjunto de teste, no formato “Teste n”, onde n é numerado sequencialmente a partir de 1. A seguir seu programa deve escrever uma linha para cada depósito do conjunto de testes. Cada linha deve conter um inteiro que representa a diferença (em centavos) entre o valor depositado nos cofrinhos do Joãozinho e do Zezinho. Deixe uma linha em branco ao final de cada conjunto de teste. A grafia mostrada no Exemplo de Saída, abaixo, deve ser seguida rigorosamente.

**Exemplo:**

|  |  |
| --- | --- |
| **Entrada (cofre.in)** | **Saída (cofre.out)** |
| 3  20 25  10 5  10 10  4  0 5  12 0  0 20  17 1  0 | Teste 1  -5  0  0  Teste 2  -5  7  -13  3 |

**Restrições:**

0 ≤ N ≤ 100 (N = 0 apenas para indicar o fim da entrada)

0 ≤ J ≤ 100 (valor de cada depósito no cofre de Joãozinho)

0 ≤ Z ≤ 100 (valor de cada depósito no cofre de Zezinho)

**Questão 2 (Linguagem C++, 20 pts.):**

Você conseguiu um estágio para trabalhar como programador na secretaria da sua escola. Como primeira tarefa, Dona Vilma, a coordenadora, solicitou que você aprimore um programa que foi desenvolvido pelo estagiário anterior. Esse programa tem como entrada uma lista de nomes e de médias finais dos alunos de uma turma, e determina o aluno com a maior média na turma. Dona Vilma pretende utilizar o programa para premiar o melhor aluno de cada turma da escola. O programa desenvolvido pelo estagiário anterior encontra-se a seguir:

|  |
| --- |
| **Linguagem: C++** |
|  |

Como você pode verificar, o programa na forma atual tem uma imperfeição: no caso de haver alunos empatados com a melhor média na turma, ele imprime apenas o primeiro aluno que aparece na lista.

**Tarefa:**

Dona Vilma deseja que você altere o programa para que ele produza uma lista com todos os alunos da turma que obtiveram a maior média, e não apenas um deles. Você consegue ajudá-la nesta tarefa?

**Entradas:**

A entrada é constituída de vários conjuntos de teste, representando várias turmas. A primeira linha de um conjunto de testes contém um número inteiro N (1 ≤ N ≤ 1000) que indica o total de alunos na turma. As N linhas seguintes contêm, cada uma, um par de números inteiros C (1 ≤ C ≤ 20000) e M (0 ≤ M ≤ 100), indicando respectivamente o código e a média de um aluno. O final da entrada é indicado por uma turma com N = 0.

**Saídas:**

Para cada turma da entrada seu programa deve produzir três linhas na saída. A primeira linha deve conter um identificador do conjunto de teste, no formato “Turma n”, onde n é numerado a partir de 1. A segunda linha deve conter os códigos dos alunos que obtiveram a maior média da turma. Os códigos dos alunos devem aparecer na mesma ordem da entrada, e cada um deve ser seguido de um espaço em branco. A terceira linha deve ser deixada em branco. O formato mostrado no exemplo de saída abaixo deve ser seguido rigorosamente.

**Exemplo:**

|  |  |
| --- | --- |
| **Entrada (estagio.in)** | **Saída (estagio.out)** |
| 3  1 85  2 91  3 73  5  12300 81  12601 99  15023 76  10111 99  212 99  0 | Turma 1  2  Turma 2  12601 10111 212 |

**Restrições:**

0 ≤ N ≤ 1000 (N = 0 apenas para indicar o fim da entrada)

1 ≤ C ≤ 20000

0 ≤ M ≤ 100

**Questão 3 (Linguagem C++, 20 pts..):**

O quebra-cabeças Torres de Hanoi é muito antigo e conhecido, sendo constituído de um conjunto

de N discos de tamanhos diferentes e três pinos verticais, nos quais os discos podem ser encaixados.

Diagrama

Descrição gerada automaticamente

Cada pino pode conter uma pilha com qualquer número de discos, desde que cada disco não seja colocado acima de outro disco de menor tamanho. A configuração inicial consiste de todos os discos no pino 1. O objetivo do quebra-cabeças é mover todos os discos para um dos outros pinos, sempre obedecendo à restrição de não colocar um disco sobre outro menor. Um algoritmo para resolver este problema é o seguinte:

|  |
| --- |
| procedimento Hanoi(N, Orig, Dest, Temp)  se N = 1 então  mover o menor disco do pino Orig para o pino Dest;  senão  Hanoi(N-1, Orig, Temp, Dest);  mover o N-ésimo menor disco do pino Orig para o pino Dest;  Hanoi(N-1, Temp, Dest, Orig);  fim-se  fim |

**Tarefa:**

Sua tarefa é escrever um programa que determine quantos movimentos de trocar um disco de um pino para outro serão executados pelo algoritmo acima para resolver o quebra-cabeça.

**Entradas:**

A entrada possui vários conjuntos de teste. Cada conjunto de teste é composto por uma única linha, que contém um único número inteiro N (0 ≤ N ≤ 30), indicando o número de discos. O final da entrada é indicado por N = 0.

**Saídas:**

Para cada conjunto de teste, o seu programa deve escrever três linhas na saída. A primeira linha deve conter um identificador do conjunto de teste, no formato “Teste n”, onde n é numerado sequencialmente a partir de 1. A segunda linha deve conter o número de movimentos que são executados pelo algoritmo dado para resolver o problema das Torres de Hanói com N discos. A terceira linha deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser seguida rigorosamente.

**Exemplo:**

|  |  |
| --- | --- |
| **Entrada (hanoi.in)** | **Saída (hanoi.out)** |
| 1  2  0 | Teste 1  1  Teste 2  3 |

**Restrições:**

0 ≤ N ≤ 30 (N = 0 apenas para indicar o final da entrada)

**Questão 4 (Linguagem Go, 20 pts.):**

Muitas crianças gostam de decidir todas as disputas através do famoso jogo de Par ou Ímpar. Nesse jogo, um dos participantes escolhe Par e o outro, ímpar. Após a escolha, os dois jogadores mostram, simultaneamente, uma certa quantidade de dedos de uma das mãos. Se a soma dos dedos das mãos dos dois jogadores for par, vence o jogador que escolheu Par inicialmente, caso contrário vence o que escolheu Ímpar.

**Tarefa:**

Dada uma sequência de informações sobre partidas de Par ou Ímpar (nomes dos jogadores e números que os jogadores escolheram), você deve escrever um programa para indicar o vencedor de cada uma das partidas.

**Entrada:**

A entrada é composta de vários conjuntos de testes. A primeira linha de um conjunto de testes contém um inteiro N, que indica o número de partidas de Par ou Ímpar que aconteceram. As duas linhas seguintes contêm cada uma um nome de jogador. Um nome de jogador é uma cadeia de no mínimo um e no máximo dez letras (maiúsculas e minúsculas), sem espaços em branco. As N linhas seguintes contêm cada uma dois inteiros A e B que representam o número de dedos que cada jogador mostrou em cada partida (0 ≤ A ≤ 5 e 0 ≤ B ≤ 5). Em todas as partidas, o primeiro jogador sempre escolhe Par. O final da entrada é indicado por N = 0.

**Saída:**

Para cada conjunto de teste da entrada, seu programa deve produzir a saída da seguinte forma. A primeira linha deve conter um identificador do conjunto de teste, no formato “Teste n”, onde n é numerado sequencialmente a partir de 1. As próximas N linhas devem indicar o nome do vencedor de cada partida. A próxima linha deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser seguida rigorosamente.

**Exemplo:**

|  |  |
| --- | --- |
| **Entrada (par.in)** | **Saída (par.out)** |
| 3  Pedro  Paulo  2 4  3 5  1 0  2  Claudio  Carlos  1 5  2 3  00 | Teste 1  Pedro  Pedro  Paulo  Teste 2  Claudio  Carlos |

**Restrições:**

0 ≤ N ≤ 1000 (N = 0 apenas para indicar o fim da entrada)

0 ≤ A ≤ 5

0 ≤ B ≤ 5

1 ≤ comprimento do nome de jogador ≤ 10

**Questão 5 (Linguagem Go, 20 pts..):**

A vovó tem um televisor muito antigo, que ultimamente está exibindo um defeito incômodo: a imagem aparece ‘deslocada’ (para cima ou para baixo, para o lado direito ou para o lado esquerdo). Quando a imagem está deslocada para cima, a parte da imagem que deixa de ser vista na parte superior reaparece na parte de baixo da tela. Da mesma forma, quando a imagem está deslocada a direita, a parte da imagem que deixa de ser vista à direita reaparece na tela do lado esquerdo.

Diagrama

Descrição gerada automaticamente com confiança média

A imagem do televisor pode ser vista como uma matriz de pontos organizados em linhas e colunas. Para consertar o televisor da vovó, você pode ajustar a imagem introduzindo uma série de ‘comandos de correção’ em um painel de ajuste. Cada comando de correção desloca a imagem de um certo número de linhas (para cima ou para baixo) e um certo número de colunas (para a direita ou para a esquerda).

**Tarefa:**

Dada uma matriz que representa uma imagem defeituosa e uma série de comandos de correção, seu programa deve calcular a matriz que representa a imagem resultante após todos os comandos terem sido aplicados sequencialmente.

**Entrada:**

A entrada possui vários conjuntos de teste. Cada conjunto de teste inicia com a descrição da matriz que representa a imagem do televisor. A primeira linha contém dois inteiros M e N representando o número de linhas e o número de colunas da matriz (1 ≤ M ≤ 1000 e 1 ≤ N ≤ 1000). As M linhas seguintes da entrada contém cada uma N inteiros, descrevendo o valor de cada ponto da imagem. Após a descrição da imagem, segue-se a descrição dos comandos de correção. Cada comando de correção é descrito em uma linha contendo dois inteiros X e Y. O valor de X representa o deslocamento na direção horizontal (valor positivo representa deslocamento para a direita, valor negativo para a esquerda), e o valor de Y representa o deslocamento da direção vertical (valor positivo para cima, valor negativo para baixo). O final da lista de comandos é indicado por X = Y = 0, e o final da entrada é indicado por M = N = 0.

**Saída:**

Para cada conjunto de teste, o seu programa deve produzir uma imagem na saída. A primeira linha da saída deve conter um identificador do conjunto de teste, no formato “Teste n”, onde n é numerado sequencialmente a partir de 1. A seguir deve aparecer a matriz que representa a imagem resultante, no mesmo formato da imagem de entrada. Ou seja, as N linhas seguintes devem conter cada uma M inteiros que representam os pixels da imagem. Após a imagem deixe uma linha em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser seguida rigorosamente.

**Exemplo:**

|  |  |
| --- | --- |
| **Entrada (tv.in)** | **Saída (tv.out)** |
| 3 3  1 2 3  4 5 6  7 8 9  1 0  1 -1  0 0  3 4  6 7 8 5  10 11 12 9  2 3 4 1  -3 2  0 0  0 0 | Teste 1  8 9 7  2 3 1  5 6 4  Teste 2  1 2 3 4  5 6 7 8  9 10 11 12 |

**Restrições:**

0 ≤ N ≤ 1000 (N = 0 apenas para indicar o final da entrada)

0 ≤ M ≤ 1000 (M = 0 apenas para indicar o final da entrada)

0 ≤ X ≤ 1000

0 ≤ Y ≤ 1000

0 ≤ número de comandos de correção em cada conjunto de teste ≤ 1000

**Boa Prova!**