7ª Maratona de Programação da SDC

17 de Outubro de 2019

Caderno de Problemas

Informações Gerais

Este caderno contém 8 problemas; as páginas estão numeradas de 1 a 12, não contando com esta página. Verifique se o caderno está completo.

- A prova pode ser resolvida por equipes de até três integrantes;
- A prova tem duração de três horas;
- Não é permitido a consulta de qualquer material online;
- É permitido o uso de material impresso;
- O nome do arquivo de cada solução deve ser *codigo_do_problema,* onde *codigo_do_problema* é a letra maiúscula que identifica o problema. Por exemplo: Problema A, o nome do programa deve ser A.(c | cpp | py | java).

Problema Extra: Soma

Timelimit: 1 segundo

Some três inteiros.

Entrada:

Três números inteiros.

Saída:

A saída consiste na soma de três variáveis.

Exemplos de Entrada	Exemplos de Saída
3 2 1	6

Solução em C++

```
#include <iostream>

int main()
{
   int a, b, c, soma;

   std::cin >> a >> b >> c;

   soma = a + b + c;

   std::cout << soma << '\n';

   return 0;
}
```

Solução em Python

```
a = int(input())
b = int(input())
c = int(input())

soma = a + b + c

print("{}".format(soma))
```

Solução em Java

```
import java.util.Scanner;

public class Soma {
	public static void main(String[] args) {
		Scanner in = new Scanner(System.in)
		int a, b, c, soma;

		a = in.nextInt();
		b = in.nextInt();
		c = in.nextInt();

		soma = a + b + c;
		System.out.println(soma);

	}
```

Problema A: Mochilas do CCHLA

TImelimit: 2 segundos

Bola Fernandes precisa da sua ajuda para realizar uma das mais deliciosas atividades da Semana da Computação (SDC), o Coffee Break. Ele tem uma lista de alimentos para comprar, cada alimento possui um peso e um valor referente ao nível de satisfação do público da SDC, por exemplo, a galera gosta mais de coxinha do que bolo.

O problema é que Bola Fernandes costuma comprar as suas mochilas nos bazares do CCHLA enquanto está “refletindo” um pouco lá na Praça da Alegria, essas mochilas duvidosas têm vontade própria e não aceitam que alguns alimentos sejam armazenados no seu interior. A sua tarefa é ajudar Bola Fernandes a escolher qual das suas mochilas armazenará os itens que resultam no maior valor de satisfação.

Entrada:

A entrada possui um único caso de teste. A primeira linha da entrada tem dois inteiros **N** (5 ≤ **N** ≤ 10000) e **M** (1 ≤ **M** ≤ 100), separados por espaços, indicando o número de alimentos e a quantidade de mochilas, respectivamente. Seguem **N** linhas, cada uma com dois inteiros **V** (0 ≤ **V** ≤ 10000) e **W** (1 ≤ **W** ≤ 10000), separados por espaços, indicando o valor de satisfação e o peso de cada alimento, respectivamente. Nas linhas seguintes teremos **M** mochilas, cada uma representada por dois inteiros **E** (0 ≤ **E** ≤ **N**) e **C** (1 ≤ **C** ≤ 100000), separados por espaços, indicando a quantidade de alimentos que não podem ser armazenados e a capacidade da mochila, na linha que segue a definição de cada mochila teremos **E** inteiros **I** (0 ≤ **I** ≤ **N**) ordenados, onde i-ésimo inteiro representa o índice do alimento que não pode ser armazenado.

Saída:

Seu programa deve produzir uma única linha indicando qual a melhor mochila e o valor de satisfação obtido em sua utilização.

Exemplos de Entrada	Exemplos de Saída
5 5 100 8 60 3 70 6 15 4 15 2 2 10 0 3 1 8	1 137.5

4 3 11 0 1 2 2 9 0 3 1 4 4	

5 5 60 3 15 4 70 6 15 2 100 8 2 10 0 3 1 8 4 3 11 0 1 2 2 9 0 3 1 4 4	1 123.333

Problema B: Crescente e Decrescente

Timelimit = 1 segundo

Em um planeta chamado Galeticos, todos os números precisam estar ordenados tanto de forma crescente como de forma decrescente. Porém, devido a uma pane no sistema do planeta todos os números ficaram desordenados e, consequentemente os habitantes do planeta estão adoecendo.

A única forma de salvar a população de Galeticos é ordenando os números. Você poderia ajudar? Tudo o que você precisa fazer é receber uma série de números, inserir os mesmos em um vetor e ordenar.

Entrada:

Leia um inteiro **N,** que representa o tamanho do vetor. As **N** linhas seguintes são valores inteiros que serão inseridos no vetor.

Saída:

A saída consiste de duas linhas. A primeira linha deve apresentar “Crescente = “ seguido do vetor ordenado de forma crescente. Já a segunda linha apresenta “Decrescente = “ seguido do vetor ordenado de forma decrescente.

Exemplos de Entrada	Exemplos de Saída
3 110 2 300	Crescente = 2 110 300 Decrescente = 300 110 2
10 2 0 4 6 9 4 10 30 250 50	Crescente = 0 2 4 4 6 9 10 30 50 250 Decrescente = 250 50 30 10 9 6 4 4 2 0

Problema C: Carro no Planeta G7787878

Timelimit: 2 segundos

Criado por: Thiago Gouveia da Silva
Revisado por: Wesley

Há um pequeno planeta rochoso localizado nos ermos da Via Láctea. Seu nome: GRandInt. Aconteceu que no momento da concepção deste problema, seu nome foi G7787878.

O chefe do departamento de trânsito deste planeta, o bom e velho senhor 4569778 está planejando o novo sistema de placas para todos os veículos de G7787878. Antes de propor seu novo sistema, ele deseja saber quantos veículos o sistema atual consegue identificar. O sistema atual utiliza quatro grupos distintos de caracteres:

- **L:** composto pelas 26 letras minúsculas do alfabeto;
- **N:** composto pelos 10 dígitos indo-arábicos;
- **1**: composto pelos 3 símbolos '$', '@' e '&';
- **2**: composto pelos 4 símbolos '!', '%', '#', e '*'.

Um sistema de placas consiste em combinar tais grupos em uma sequência imutável predefinida, tal qual **LNN**, capaz de identificar 2600 veículos, ou **NNN12**, que identifica 12000 veículos. Você poderia calcular quantos veículos um sistema é capaz de identificar?

Entrada:

A entrada possui vários casos de teste. A primeira linha da entrada tem um inteiro **T** (1 ≤ **T** ≤ 20), indicando o número de casos de teste. Seguem **T** linhas, cada uma com uma palavra (com no máximo cinco letras) representando um sistema de placas.

Saída:

Para cada caso de teste, seu programa deve imprimir o maior número de veículos que o sistema é capaz de identificar.

Exemplos de Entrada	Exemplos de Saída
4 1 11 2 22	3 9 4 16
4 2NL1N 1LN1N 2N12L 11LL1	31200 23400 12480 18252

Problema D: Ignore os carros do Planeta G155487 versão 2

Timelimit: 2 segundos

Criado por: Thiago Gouveia da Silva

Revisado por: Wesley

Há um pequeno planeta rochoso localizado nos ermos da Via Láctea. Seu nome: GRandInt versão 2. Aconteceu que no momento da concepção deste problema, seu nome foi G155487 versão 2.

O chefe do departamento de trânsito deste planeta, o bom e velho senhor 4569778 está planejando o novo sistema de placas para todo os veículos de G155487 versão 2. O que ele deseja é que o sistema seja capaz de prover uma placa para cada veículo do planeta, mas que sobre o menor número de placas possível para novos veículos. O senhor 4569778 optou por utilizar quatro grupos distintos de caracteres:

- **L:** composto pelas 26 letras minúsculas do alfabeto;
- **N:** composto pelos 10 dígitos indo-arábicos;
- **1**: composto pelos 3 símbolos ‘$’, ‘@’ e ‘&’;
- **2**: composto pelos 4 símbolos ‘!’, ‘%’, ‘#’, e ‘*’.

Um sistema de placas consiste em combinar tais grupos em uma sequência imutável predefinida, tal qual **LNN**, capaz de identificar 2600 veículos, ou **NNN12**, que identifica 12000 veículos. Você poderia calcular o melhor sistema de placas para G155487 versão 2, segundo a definição do senhor 4569778?

Entrada:

A entrada possui vários casos de teste. A primeira linha da entrada tem um inteiro **T** (1 ≤ **T** ≤ 10), indicando o número de casos de teste. Seguem **T** linhas, cada uma com um número inteiro v (1 ≤ **v** ≤ 108), representando um caso de teste, que é o número de veículos para o qual se deseja calcular o melhor sistema de placas.

Saída:

Para cada caso de teste, seu programa deve imprimir uma linha com os valores **v**, **b** e **s**, separados por um espaço. Onde **s** é o melhor sistema de placas possível e **b** é o número de veículos que o sistema **s** é capaz de identificar. Caso vários sistemas empatem, deve ser escolhido o que tem menos grupos. Caso o empate persista, escolha aquele que é lexicograficamente menor, considerando que **1 < 2 < L < N**, lexicograficamente falando.

Exemplos de Entrada	Exemplos de Saída
5 41314 13 750909 837 6	41314 41600 22LNN 13 16 22 750909 758160 1111112LN 837 900 11NN 6 9 11

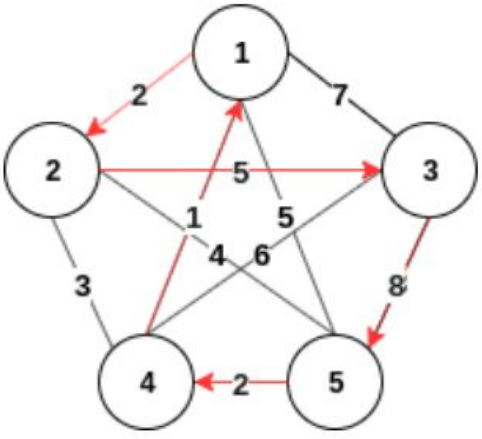
Problema E: Melhor Rota

Timelimit: 2 segundo

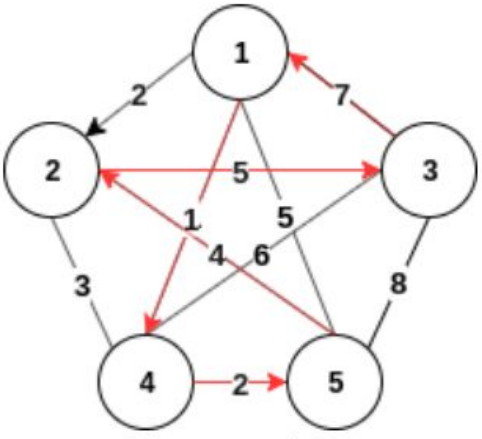
Na UFPB existem diversas lendas: gatos são alunos que passaram muito tempo e não se formaram, alunos eletrocutados nos banheiros do RU, o fluxograma de Ciência da Computação vai mudar, etc. Entre essas diversas lendas a que nos interessa é a de um suposto ônibus que faz o trajeto SEDE <- -> CI, felizmente através da influência dos alunos do Pet Computação essa lenda tornou-se verdadeira fazendo um trajeto diferente que seria deixar os alunos cadastrados em suas respectivas casas.

Visando uma maior economia de combustível a reitora Waldiscleide pediu para que Sorriso, um dos membros do Pet encontrasse a melhor rota, tal rota foi encontrada baseando-se no famoso problema do caixeiro viajante, cujo objetivo é tentar determinar a menor rota para percorrer uma série de locais (visitando uma única vez cada um deles), retornando ao local de origem.

O problema é que alguns começaram a se queixar, pois mesmo que o ônibus estivesse perto de sua casa não poderia desviar para manter a rota de custo mínimo, fazendo alguns alunos esperarem muito um tempo desnecessário para chegar em casa. Ao receber o feedback dos alunos sorriso pensou em abordar o problema de outra forma, baseando-se no problema de Mínima Latência que é uma variante do PCV no qual a meta é minimizar a soma dos tempos de chegada em todos os nós. Para entender a diferença entre o Problema do Caixeiro Viajante e o Problema de Mínima Latência temos um breve exemplo abaixo.



Caixeiro Viajante



Mínima Latência

A solução ótima para o Caixeiro Viajante é 18 (2 + 5 + 8 + 2 + 1), para a Mínima Latência a solução ótima é 42 (1 + (1 + 2) + (1 + 2 + 4) + (1 + 2 + 4 + 5) + (1 + 2 + 4 + 5 + 7)). Como podemos observar acima o local 4 na solução do caixeiro viajante mesmo estando bem próximo do ponto de partida é o penúltimo a ser visitado, já na solução da mínima latência é o segundo a ser visitado, mostrando que essa nova abordagem resolveria a principal queixa dos alunos.

Sorriso têm estado bastante ocupado, então ajude-o a resolver esse problema encontrando a rota da solução ótima para o problema da Mínima Latência. Lembrando que a rota sempre começa no CI e termina no CI que é o local 1.

Entrada:

A entrada possui um único caso de teste. A primeira linha da entrada tem um inteiro **N** (4 ≤ **N** ≤ 10), indicando o número de locais que serão visitados incluindo o de origem. Seguem **N** linhas, cada uma com **N** inteiros, separados por espaços. O j-ésimo inteiro da i-ésima linha representa a distância **D** (0 ≤ **D** ≤ 10000) do local i até j.

Exemplos de Entrada	Exemplos de Saída
5 0 2 7 1 5 2 0 5 3 4 7 5 0 6 8 1 3 6 0 2 5 4 8 2 0	1 4 5 2 3 1 42 19

Saída:

A saída deve conter N + 1 números inteiros, representando a sequência de locais visitados para minimizar a latência. Seque um inteiro X, representando o valor mínimo encontrado para a latência. Por fim, segue um inteiro Y, representando a distância total percorrida, considerando o trajeto que resultou na latência mínima.

Problema F: Raio do Círculo

Timelimit: 1 segundo

Pedrinho está aprendendo a calcular o raio de um círculo e sua professora passou uma tarefa de casa com 100 exercícios, valendo 3 pontos se todas as questões estiverem corretas. Como todo bom aluno, Pedrinho quer muito ganhar esses pontos, então ele pediu a sua ajuda para criar uma programa que calculasse o raio de um círculo para que ele pudesse verificar as respostas.

Entrada:

A entrada consiste de um valor de ponto flutuante (double) **N,** este valor representa o diâmetro do círculo.

Saída:

Apresenta “RAIO = “ seguido do valor do raio, com 4 casas decimais. Preste atenção aos espaços e não esqueça da quebra de linha no final.

Exemplos de Entrada	Exemplos de Saída
3.0	RAIO = 1.5000
9517.539	RAIO = 4758.7695
20.569	RAIO = 10.2845

Problema G: Hora do Campeonato de Mangabeira

Timelimt: 2 segundos

Criado por: Thiago Gouveia da Silva

O bairro de Mangabeira, em João Pessoa, na Paraíba é, segundo seus próprios habitantes, o maior bairro residencial do continente, quiçá do universo. Oriundos de Mangabeira, Pepeta e Bubu se tornaram famosos jogadores de futebol, conseguindo atuar em de grandes agremiações do Brasil, em especial no XV de Jaú, no Goytacaz-RJ, na União Barbarense e na Desportiva Ferroviária.

Tomados por esta onda de sucesso, e levados pela felicidade de se encontrarem em um jogo festivo em Lapaz, na Bolívia, Pepeta e Bubu resolveram realizar um campeonato de futebol entre os times de Mangabeira. A restrição é que o campeonato seria composto apenas por times que nunca se enfrentaram na história.

Sua tarefa é: dados os nomes de todos os times de Mangabeira e suas partidas, verificar o número máximo de times que o Campeonato de Mangabeira poderá ter.

Entrada:

A entrada contém apenas um caso de teste. A primeira linha da entrada tem dois inteiros **N** ($1 \leq N \leq 25$) e **M** ($0 \leq M \leq N^2$), indicando, respectivamente, o número de times e o número de partidas oficiais na história de Mangabeira.

Cada uma das próximas **N** linhas traz o nome de um time (200 caracteres ou menos, sem espaços).

Cada uma das **M** linhas seguintes traz uma partida no formato TimeA x Time B.

Saída:

A saída é apenas um número inteiro, o número máximo de times.

Exemplos de Entrada	Exemplos de Saída
5 6 OspinaColadadeDezembro EtoisdeMarcoaoContrario GaloFrito-RN BichodoMato VoupraCima-PE BichodoMato x OspinaColadadeDezembro BichodoMato x GaloFrito-RN GaloFrito-RN x VoupraCima-PE GaloFrito-RN x EtoisdeMarcoaoContrario VoupraCima-PE x OspinaColadadeDezembro EtoisdeMarcoaoContrario x OspinaColadadeDezembro	3