

Universidade Federal de Goiás
Instituto de Informática
Introdução à Programação
Lista - L1- b

Prof. Msc. Elias Batista Ferreira
Prof. Dr. Gustavo Teodoro Laureano
Profa. Dra. Luciana Berretta
Prof. Dr. Thierson Rosa Couto

Instruções para a Resolução dos Problemas

Os problemas devem ser submetidos ao sistema Sharif da sua turma. A pontuação de cada problema é definida de acordo com o grau de dificuldade do problema, conforme a tabela abaixo:

Grau de Dificuldade	Pontos
+	1
++	2
+++	3
++++	4
+++++	5

A lista Lista L1 completa vale 100 pontos (que correspondem a 10 em termos de nota da lista). No Sharif a Lista L1 aparece segmentada em três listas. Este texto corresponde à lista L1- b. Para obter os cem pontos o aluno deve conseguir resolver um número de exercícios de graus de dificuldade +, ++ e +++ que somados formem 90 pontos. Pontos excedentes obtidos com exercícios desses graus de dificuldade serão descartados. Os 10 pontos restantes devem ser obtidos resolvendo-se problemas com graus de dificuldade ++++ ou +++++.

Sumário

1	Turma de Introdução à Programação	3
2	Ultrapassagem populacional	5
3	Valor de y dado x	6
4	Arredondamento	7
5	Cálculo da Área de um Triângulo	8
6	Custo Final de um Carro	9
7	Decolagem	10
8	Distância entre dois pontos	12
9	Fatorial	13
10	Gerador de tabuada	14
11	José	15
12	Maior segmento crescente de uma sequência	16
13	Maior segmento igual de uma sequência	17
14	Número de finais	18
15	Número primo	19
16	Número primo	20
17	Ordena 3 números	21
18	Quatro Algarismos	22
19	Raízes de equações de grau 2	23
20	Salário	25
21	Sequência ordenada	26
22	Soma dos 3 menores	27
23	Transcrição de datas	28

1 Turma de Introdução à Programação



(+)

A disciplina de Introdução à Programação possui oito provas, cinco listas de exercícios e uma nota de trabalho final. Para que um aluno seja aprovado por nota na disciplina, ele deve obter uma nota final maior ou igual a seis. A nota final é computada pela seguinte fórmula:

$$NF = 0.7 \cdot MP + 0.15 \cdot ML + 0.15 \cdot NT \quad (1)$$

onde MP é a média aritmética das notas de prova, ML é a média aritmética das notas das cinco listas e NT é a nota do trabalho final.

Para ser aprovado na disciplina o aluno deve ter presença igual a ou superior a 75% da carga horária da disciplina que no caso de Introdução à Programação é 128 horas.

Escreva um programa para ler as notas de cada aluno de uma turma, computar a nota final do aluno e imprimir a nota final e uma indicação da situação final do aluno. Essa indicação pode ser uma das seguintes alternativas:

- Aprovado - se o aluno teve $NF \geq 6$ e presença superior à quantidade de horas mínima.
- Reprovado por nota - se o aluno teve a presença mínima, mas sua nota NF não é suficiente para ser aprovado.
- Reprovado por frequência insuficiente - o aluno obteve nota NF superior ou igual a seis mas sua presença às aulas não foi suficiente para ser aprovado.
- Reprovado por frequência e por nota - o aluno não alcançou o valor mínimo de NF e também não tem frequência mínima para aprovação.

Entrada

A entrada contém várias linhas, cada uma contendo os dados de um aluno separados entre si por um espaço. O primeiro valor em uma linha corresponde à matrícula do aluno (um valor inteiro sem sinal). Os próximos oito valores seguintes correspondem às notas das oito provas. Os seguintes cinco valores correspondem às notas obtidas nas listas de exercícios. O penúltimo valor corresponde a nota do trabalho final e o último valor em uma linha corresponde à presença do aluno. A última linha da entrada contém todos os valores iguais a -1 essa linha serve apenas para indicar o fim da entrada e não deve ser processada.

Saída

O programa deve gerar uma linha para cada aluno contendo a seguinte frase: “Matricula: m , Nota Final: n , Situação Final: s ”. O valor de m corresponde à matrícula de um aluno, o valor de n corresponde ao valor da nota final (NF) do aluno e s é uma das seguintes frases correspondendo à situação final do aluno:

- APROVADO
- REPROVADO POR FREQUENCIA
- REPROVADO POR NOTA
- REPROVADO POR NOTA E POR FREQUENCIA

Exemplo

Entrada																			
4448901	2.3	3.2	4.3	5.0	6.5	7.2	7.3	8.4	9.2	8.3	9.5	7.6	10.0	9.0	118				
4448902	4.5	8.2	4.4	7.0	7.5	9.2	8.3	9.5	10.0	9.2	8.3	9.5	7.6	10.0	80				
4448903	4.1	3.2	4.4	5.0	6.5	6.2	5.5	6.5	8.4	7.2	6.3	9.5	5.6	4.0	110				
4448903	4.1	3.2	4.4	5.0	6.5	6.2	5.5	6.5	8.4	7.2	6.3	9.5	5.6	4.0	80				
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1				
Saída																			
Matricula:	4448901,	Nota Final:	6.56,	Situacao Final:	APROVADO														
Matricula:	4448902,	Nota Final:	7.97,	Situacao Final:	REPROVADO POR FREQUENCIA														
Matricula:	4448903,	Nota Final:	5.33,	Situacao Final:	REPROVADO POR NOTA														
Matricula:	4448903,	Nota Final:	5.33,	Situacao Final:	REPROVADO POR NOTA E POR FREQUENCIA														

2 Ultrapassagem populacional



(+)

Supondo que a população de um país A seja de a habitantes com uma taxa anual de crescimento de 3% e que a população de um país B seja de b habitantes, com uma taxa anual de crescimento de 1,5%, fazer um algoritmo que calcule e escreva o número de anos necessários para que a população do país A ultrapasse ou iguale a população do país B , mantidas essas taxas de crescimento.

Entrada

O programa deverá ler duas linhas de entrada, cada uma contendo um número inteiro positivo representando a população de um país. O valor na primeira linha corresponde ao número de habitantes do país A e será sempre menor que o valor na segunda linha, o qual corresponde ao número de habitantes do país B .

Saída

A saída deve conter, numa linha com a frase $\text{ANOS} = x$, onde x é um valor em anos e deve ser seguido por um caractere de quebra de linha: `'\n'`.

Exemplo

A seguir são mostrados dois casos distintos de entrada, somente para efeito de ilustração, porém, esse problema contém apenas um caso de teste na entrada, formado pelas duas linhas de entrada descritas acima.

Entrada
900000000
2000000000
Saída
ANOS = 55

3 Valor de y dado x



(+)

Desenvolver um algoritmo para ler um número x , calcular e imprimir o valor de y de acordo com as condições abaixo:

$$y = \begin{cases} x, & \text{se } x < 1; \\ 0, & \text{se } x = 1; \\ x^2, & \text{se } x > 1; \end{cases}$$

Entrada

O programa deve ler uma linha contendo um único número inteiro correspondendo ao valor de x .

Saída

O programa deve imprimir $Y = y$, onde y é o valor computado de y dado x . Após o valor de y , o programa deve imprimir um caractere de quebra de linha: ‘\n’.

Exemplo

Entrada
3
Saída
Y = 9

4 Arredondamento



(++)

Escreva um algoritmo que leia um número real e realize o arredondamento deste número usando 1, 2 e 3 casas decimais. A apresentação do número deve conter, obrigatoriamente 6 casas decimais. As casas decimais posteriores ao dígito arredondado devem conter o valor 0.

Considerações

O arredondamento de um número é uma operação que elimina algarismos de menor significância. A regra de arredondamento aplica-se nos algarismos situados após a posição da quantidade de casas decimais desejada. Ou seja, o processo de arredondamento do número 12.318215 considerando 1 casa decimal deve avaliar os números 18215. Para 2 casas decimais deve-se avaliar os números 8215, e assim por diante.

- Se o algarismo seguinte for menor que 5, então o anterior não se modifica
- Se o algarismo seguinte for maior ou igual a 5, então o anterior é incrementado

Entrada

O programa deve ler 1 valor real.

Saída

O programa deve imprimir a primeira linha contendo o número arredondado com 1 casa decimal, a segunda com 2 casas decimais e a terceira com 3 casas decimais.

Exemplo

Entrada
3.1752
Saída
3.200000
3.180000
3.175000
Entrada
0.1825
Saída
0.200000
0.180000
0.183000

5 Cálculo da Área de um Triângulo



(++)

Desenvolver um algoritmo para ler os comprimentos dos três lados de um triângulo (L_1 , L_2 e L_3) e calcular a área do triângulo.

Considerações

A área de um triângulo pode ser computada pela fórmula:

$$A = \sqrt{T(T - L_1)(T - L_2)(T - L_3)}$$

onde

$$T = \frac{L_1 + L_2 + L_3}{2}$$

A função `sqrt()` computa a raiz quadrada de uma expressão. Para usar essa função você deve incluir o arquivo de cabeçalho `math.h`, inserindo a seguinte diretiva de pré-processamento logo no início do seu arquivo com o programa em C: **#include** <math.h>

Entrada

O programa deve ler três valores reais na entrada, cada um correspondendo ao comprimento de um lado do triângulo. Cada valor ocorre em uma linha diferente na entrada.

Saída

O programa deve imprimir uma linha contendo a frase: A AREA DO TRIANGULO E = X , onde X é o valor da área do triângulo e deve conter no máximo 2 casas decimais. Após o valor da área do triângulo, o programa deve imprimir um caractere de quebra de linha: `"\n"`.

Exemplo

Entrada
4
5
6
Saída
AREA DO TRIANGULO E = 9.92

6 Custo Final de um Carro



(++)

O custo ao consumidor de um carro novo é a soma do custo de fábrica com a porcentagem do distribuidor e dos impostos (aplicados ao custo de fábrica). Supondo que a porcentagem do distribuidor seja de $x\%$ do preço de fábrica e os impostos de $y\%$ do preço de fábrica, fazer um programa para ler o custo de fábrica de um carro, a porcentagem do distribuidor e o percentual de impostos, calcular e imprimir o custo final do carro ao consumidor.

Entrada

O programa deve ler três valores na entrada: o preço de fábrica do carro, o percentual do distribuidor e o percentual de impostos. Cada valor aparece em uma linha de entrada. Todos os valores são do tipo float.

Saída

O programa deve imprimir uma linha, contendo a frase O VALOR DO CARRO E = Z, onde Z é o valor do preço final do carro ao consumidor. O valor de Z deve ter duas casas decimais. Após imprimir o valor do preço final, o programa deve imprimir o caractere de quebra de linha '\n'.

Observações

Exemplo

Entrada
25000
12
30
Saída
O VALOR DO CARRO E = 35500.00

7 Decolagem



(++)

Escrever um algoritmo que leia a massa (em toneladas) de um avião, sua aceleração (m/s^2) e o tempo (s) que levou do repouso até a decolagem. O programa deve calcular e escrever a velocidade atingida (Km/h), o comprimento da pista (m) e o trabalho mecânico realizado (J) no momento da decolagem.

Dicas

- | | |
|---|--|
| <ul style="list-style-type: none">• v = velocidade; a = aceleração; t = tempo;• m = massa;• s = espaço percorrido;• W = trabalho mecânico realizado;• Um double deve ser lido com "%lf" | <ul style="list-style-type: none">• $1 \text{ m/s} = 3,6 \text{ Km/h}$;• $v = a * t$;• $s = \frac{at^2}{2}$;• $W = \frac{mv^2}{2}$;• A massa utilizada no trabalho é em Kg |
|---|--|

Entrada

O programa deve ler três linhas de entrada. A primeira linha contém um valor do tipo *double* representando a massa do avião em toneladas. A segunda linha, contém um valor do tipo *double* correspondente à aceleração de avião. A terceira, linha contém um valor do tipo *double* correspondente ao tempo em segundos gasto na decolagem.

Saída

O programa deve imprimir três linhas. A primeira, contém a frase: VELOCIDADE = x , onde x é o valor da velocidade do avião em Km/h. A segunda, contém a frase: ESPACO PERCORRIDO = y , onde y corresponde ao espaço em metros percorrido pelo avião durante a decolagem. A terceira linha contém a frase: TRABALHO REALIZADO = z , onde z corresponde ao valor do trabalho em Joules, realizado pelo avião durante a decolagem. Os valores de x , y e z devem ser do tipo *double* e devem conter duas casas decimais e após esses valores deve vir o caractere de quebra de linha `\n`.

Exemplos

Entrada
10
5
90
Saída
VELOCIDADE = 1620.00
ESPACO PERCORRIDO = 20250.00
TRABALHO REALIZADO = 1012500000.00

Entrada
3
30
25
Saída
VELOCIDADE = 2700.00
ESPACO PERCORRIDO = 9375.00
TRABALHO REALIZADO = 843750000.00

8 Distância entre dois pontos



(++)

Dados dois pontos A e B , cujas coordenadas $A(x_1, y_1)$ e $B(x_2, y_2)$ serão informadas via teclado, desenvolver um programa que calcule a distância entre A e B .

Entrada

O programa deve ler os quatro valores reais correspondendo às coordenadas dos dois pontos : x_1, y_1, x_2, y_2 , nessa ordem, e um valor por linha.

Saída

O programa deve imprimir uma linha contendo a frase: A DISTANCIA ENTRE A e B = X , onde X é o valor da distância entre os dois pontos e deve conter no máximo 2 casas decimais. Após o valor da distância, o programa deve imprimir um caractere de quebra de linha: ‘\n’.

Observações

A distância entre dois pontos é computada pela fórmula:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Você pode usar a função `sqrt()` para calcular a raiz quadrada na fórmula da distância. Para computar o quadrado de um valor x você pode usar a função `pow(x,2)`. Para usar essas funções, você precisa colocar `#include <math.h>` no início do texto do programa.

Exemplo

Entrada
3
4
5
6
Saída
A DISTANCIA ENTRE A e B = 2.83

9 Fatorial



(++)

Dado um número inteiro n , calcule seu fatorial $n!$. O fatorial de um número é dado pela equação: $n! = n(n-1)(n-2) \dots 1$. Por definição, $0! = 1$.

Entrada

O programa deve ler um número inteiro n .

Saída

O programa deve apresentar uma linha com a mensagem: " $n! = f$ ", onde n é o número lido e f o seu fatorial.

Observações

O fatorial de um número é resultado de uma operação de produtório que pode levar a valores incrivelmente grandes. Lembre-se de usar tipos de dados apropriados ao problema proposto.

Exemplo

Entrada
2
Saída
2! = 2

Entrada
4
Saída
4! = 24

10 Gerador de tabuada



(++)

Escreva um programa em linguagem C que leia um número qualquer n de 0 a 9 e imprima na tela a tabuada de soma, subtração, multiplicação e divisão desse número para K valores, iniciando em i em incrementos de s .

Entrada

O programa deve ler quatro números quaisquer n , i , K e s .

Saída

O programa deve apresentar, em sequência, a tabuada de soma, subtração, multiplicação e divisão, com o texto: "Tabuada de soma:", "Tabuada de subtracao:", "Tabuada de multiplicacao:" e "Tabuada de divisao:" antes de cada tabuada. Cada linha da tabuada segue o formato: " n op $B = R$ ", onde n é o número lido, B é o segundo termo da tabuada, op é o operador da tabuada e R o resultado da operação. Os números devem ser apresentados com 2 casas decimais.

Exemplo

Entrada
3 1 2 0.1
Saída
Tabuada de soma: 3.00 + 1.00 = 4.00 3.00 + 1.10 = 4.10 Tabuada de subtracao: 3.00 - 1.00 = 2.00 3.00 - 1.10 = 1.90 Tabuada de multiplicacao: 3.00 x 1.00 = 3.00 3.00 x 1.10 = 3.30 Tabuada de divisao: 3.00 / 1.00 = 3.00 3.00 / 1.10 = 2.73

11 José



(+++)

João tem um irmão mais novo, José, que começou a ir à escola e já está tendo problemas com números. Para ajudá-lo a pegar o jeito com a escala numérica, sua professora escreve dois números de três dígitos e pede a José para comparar esses números. Mas em vez de interpretá-los com o dígito mais significativo à esquerda, ele deve interpretá-lo com o dígito mais significativo à direita. Ele tem que dizer à professora qual o maior dos dois números. Escreva um programa que irá verificar as respostas de José.

Entrada

A entrada conterá um inteiro T , o número de casos de testes, e, para cada caso de teste, uma única linha com dois números de três dígitos, A e B , os quais não serão iguais e não conterão zeros.

Saída

A saída deve conter, numa linha para cada caso de teste, com o maior dos números na entrada, comparados como descrito no enunciado da tarefa. O número deve ser escrito invertido, para mostrar a José como ele deve lê-lo.

Exemplo

Entrada
3
734 893
221 231
839 237
Saída
437
132
938

12 Maior segmento crescente de uma sequência



(++) (POLI 89) Dados n e uma sequência de n números inteiros, determinar o comprimento de um segmento crescente de comprimento máximo.

Entrada

O programa deve ler um número inteiro maior que zero n e uma sequência de n números inteiros em qualquer ordem.

Saída

O programa deve apresentar a mensagem "O comprimento do segmento crescente maximo e: k ", onde k é o tamanho do maior segmento crescente encontrado.

Exemplo

Entrada
9
5 10 3 2 4 7 9 8 5
Saída
O comprimento do segmento crescente maximo e: 4
Entrada
5
10 8 7 5 2
Saída
O comprimento do segmento crescente maximo e: 1

13 Maior segmento igual de uma sequência



(++) (POLI 87) Dados n e uma sequência de n números inteiros, determinar quantos segmentos de números iguais consecutivos compõem essa sequência.

Entrada

O programa deve ler um número inteiro maior que zero n e uma sequência de n números inteiros em qualquer ordem.

Saída

O programa deve apresentar a mensagem "O comprimento do segmento de numeros iguais e: k ", onde k é o tamanho do maior segmento crescente encontrado.

Exemplo

Entrada
2
1 1
Saída
O comprimento do segmento de numeros iguais e: 2
Entrada
7
0 0 2 5 5 5 6
Saída
O comprimento do segmento de numeros iguais e: 3

14 Número de finais



(++)

Em um campeonato de futebol os times são nomeados como Time1, Time2, ..., TimeN. A organização do campeonato deseja saber quais são as finais possíveis dado a quantidade N de times. Para resolver esse problema, você foi contratado para fazer um programa de computador que, dada a quantidade N de times, imprima todas as configurações possíveis de finais.

Entrada

O programa deve ler um número N , inteiro e positivo, referente à quantidade de times do campeonato.

Saída

O programa deve apresentar na tela a sequência de finais com cada linha no formato: Final k : Time i X Time j , onde k é um contador de finais, i e j são as denominações de cada time. Caso o número de times informado for menor que 2, então o programa deve imprimir a mensagem: "Campeonato invalido!".

Exemplo

Entrada
3
Saída
Final 1: Time1 X Time2
Final 2: Time1 X Time3
Final 3: Time2 X Time3

Entrada
1
Saída
Campeonato invalido!

15 Número primo



(+)

Faça um programa que leia um número N e informa se o número é primo ou não.

Entrada

O programa deverá ler um número inteiro N positivo.

Saída

O programa deverá apresentar a mensagem "PRIMO" caso N seja primo e "NAO PRIMO" caso contrário. Caso o valor de N não seja um número inteiro positivo, o programa deve apresentar a mensagem "Numero invalido!".

Exemplo

Entrada
7
Saída
PRIMO

Entrada
9
Saída
NAO PRIMO

16 Número primo



(+)

Faça um programa que leia um número inteiro N e informe se o número é primo ou não.

Entrada

O programa deverá ler um número inteiro N .

Saída

O programa deverá apresentar a mensagem "PRIMO" caso N seja primo e "NAO PRIMO" caso contrário. Caso o valor de N não seja um número positivo, o programa deve apresentar a mensagem "NUMERO INVALIDO"

Exemplo

Entrada
7
Saída
PRIMO

Entrada
9
Saída
NAO PRIMO

17 Ordena 3 números



(++)

Escreva um algoritmo que leia 3 números reais em qualquer ordem e os apresente de forma ordenada na tela.

Entrada

O programa deve ler 3 valores reais.

Saída

O programa deve imprimir uma linha contendo a lista ordenada de números separados por vírgula e espaço, cada número com 2 casas decimais.

Exemplo

Entrada
3.0 1 3.1
Saída
1.00, 3.00, 3.10

18 Quatro Algarismos



(++)

Dado um número inteiro de três algarismos, construir outro número inteiro de quatro algarismos de acordo com a seguinte regra: os três primeiros algarismos, contados da esquerda para a direita são iguais ao número dado. O quarto algarismo é um dígito de controle calculado da seguinte forma: primeiro algarismo + segundo algarismo $\times 3$ + terceiro algarismo $\times 5$. O dígito de controle é igual ao resto da divisão dessa soma por 7.

Entrada

O programa deve ler uma linha de dados contendo apenas um número com três algarismos.

Saída

O programa deve imprimir uma linha contendo a frase: O NOVO NUMERO E = X , onde X é o novo número inteiro com quatro algarismos, seguido por um caractere de quebra de linha: ‘\n’.

Observações

Exemplo

Entrada
123
Saída
O NOVO NUMERO E = 1231

19 Raízes de equações de grau 2



(++)

Desenvolver um programa que leia os coeficientes (a , b e c) de uma equação de segundo grau e calcule as raízes da equação. O programa deve mostrar a classificação das raízes, e, quando possível, o valor das raízes calculadas.

Entrada

O programa deve ler três valores reais na entrada. O primeiro valor corresponde ao valor do coeficiente a , o segundo, do coeficiente b e o terceiro, do coeficiente c , de uma equação de segundo grau. Os três valores ocorrem em uma única linha na entrada, separados entre si por um espaço.

Saída

O programa deve imprimir uma linha contendo uma das seguintes frases, conforme for o resultado do cálculo das raízes da equação: RAIZES DISTINTAS, ou RAIZ UNICA, ou RAIZES IMAGINARIAS. No primeiro caso o programa deve imprimir uma outra linha contendo a frase $X1 = x_1$, onde x_1 é o valor da menor raiz encontrada para a equação. Ainda no primeiro caso, o programa deve imprimir uma terceira linha com a frase $X2 = x_2$, onde x_2 corresponde ao valor da segunda raiz. No segundo caso, o programa deve imprimir uma frase $X1 = x_1$, onde x_1 é o valor da única raiz da equação. O terceiro caso não há o que imprimir pois as raízes são imaginárias.

Observações

Dada uma equação do segundo grau do tipo $ax^2 + bx + c$, Δ (delta) $= b^2 - 4ac$. Se $\Delta = 0$, a raiz da equação é ÚNICA. Se $\Delta < 0$. As raízes da equação são IMAGINÁRIAS. Se $\Delta > 0$, então há duas RAÍZES DISTINTAS para a equação. A fórmula geral para computar as raízes de uma equação do segundo grau é a fórmula de Báskara, dada por:

$$x = \frac{-b \pm \sqrt{\Delta}}{2a}$$

Exemplo

A seguir são mostrados três exemplos distintos de entrada, e suas correspondentes saídas, entretanto, existe apenas uma linha de entrada para esse problema.

Entrada
2 12 10
Saída
RAIZES DISTINTAS
X1 = -1.00
X2 = -5.00

Entrada
2 12 18
Saída
RAIZ UNICA
X1 = -3.00

Entrada
15 17 89
Saída
RAIZES IMAGINARIAS

20 Salário



(++)

Escreva um programa que leia várias linhas contendo cada uma a matrícula de um funcionário (um valor inteiro), seu número de horas trabalhadas, o valor que recebe por hora e calcule o salário desse funcionário. A seguir, mostre o número e o salário do funcionário, com duas casas decimais.

Entrada

A entrada é formada por várias linhas, cada uma contendo três valores decimais separados um do outro por um espaço em branco. O último valor na linha é seguido pelo caractere de quebra de linha (`\n`). A última linha contém uma matrícula igual a zero e não deve ser processada. Ela serve apenas para indicar ao programa o término da entrada de dados.

Saída

A saída deve conter para cada linha de entrada a matrícula, um espaço em branco e o salário calculado com duas casas decimais

Observações

Para ler uma linha com os três valores, utilize a função **scanf**: `scanf("%d %f %f", &A, &B, &C);` e, em seguida, a função `getchar()`. A função `getchar()` é usada para consumir o caractere de quebra de linha na entrada.

Exemplo

Entrada		
2015001	16000	3.23
2015002	16010	3.0
2015003	16009	2.99
2015004	15080	3.13
0	0	0.0
Saída		
2015001	51680.00	
2015002	48030.00	
2015003	47866.91	
2015004	47200.40	

21 Sequência ordenada



(++)

Tia Zélia é uma professora muito dedicada. Ela está explicando as relações de ordem entre números para seus alunos. Na aula de hoje ela explicou a relação “menor que” representada pelo operador $<$. Ela explicou também sobre as propriedades dessa relação entre os números, incluindo a propriedade transitiva, isto é, se $x < y$ e $y < z$, então $x < z$. Tia Zélia quer fazer um treinamento com seus alunos. Ela quer propor o seguinte exercício aos seus queridos pupilos: apresentar a eles várias sequências de números e pedir que eles analisem cada sequência e que indiquem se ela está na ordem crescente. Para isso, tia Zélia quer sua ajuda. Ela vai editar um arquivo com várias sequências de números; para cada sequência haverá duas linhas no arquivo: uma com um número inteiro indicando o tamanho da sequência e a linha imediatamente seguinte contém a sequência de valores reais propriamente dita, formada por números separados por um espaço, a menos do último valor que vem seguido diretamente por um caractere de quebra de linha. Ela quer gerar sequências de números sem ter o trabalho de verificar se formam sequência ou não. Ela quer que um programa de computador faça isso para ela. Tia Zélia ficou sabendo que você é aluno de Introdução à Programação do INF-UFG e que os alunos dessa disciplina são exímios programadores! Portanto ela pede a você que faça um programa que resolva esse problema para ela.

Entrada

Para cada sequência numérica há na entrada duas linhas: uma com, apenas um valor inteiro, indica o número de valores reais que deve ocorrer na próxima linha. A linha seguinte contém tantos valores quanto indicado na linha anterior. Entre dois valores há apenas um espaço e após o último valor há um caractere de quebra de linha. A última linha da entrada contém um tamanho de sequência igual a zero e serve apenas para indicar término do processamento. Não há uma linha com sequência de valores após a ocorrência de uma linha com valor zero.

Saída

Para cada sequência da entrada o seu programa deve emitir uma das seguintes respostas: ORDENADA, se a sequência estiver em ordem crescente de valores ou DESORDENADA, em caso contrário. Após cada palavra impressa deve haver apenas um caractere de quebra de linha.

Exemplo

Entrada
10 2.98 16.42 18.0 23.67 31.99 38.50 42.30 61.782000.00 2000.10 5 4.51 4.32 4.90 56.70 150.80 6 0.00 2.56 4.00 80.4 100.98 100.97 0
Saída
ORDENADA DESORDENADA DESORDENADA

22 Soma dos 3 menores



(++)

Fazer um programa para ler quatro valores inteiros e imprimir a soma dos três menores.

Entrada

O programa deve ler quatros valores inteiros na entrada. Cada valor ocupa uma linha na entrada.

Saída

O programa deve imprimir uma linha contendo o valor da soma dos três menores números. Após o valor da soma, o programa deve imprimir um caractere de quebra de linha: ‘\n’.

Exemplo

Entrada
9
4
2
12
Saída
15

23 Transcrição de datas



(++)

Faça um algoritmo que leia uma data no formato ddmmaaaa usando um único número inteiro. Escreva a mesma data no formato dia/mês/ano, <dia> de <mês por extenso> de <ano>. O programa deve verificar se o número informado representa uma data válida. Caso não seja, imprimir na tela a mensagem "Data invalida!". Considere que o ano em questão nunca é bissexto, ou seja, fevereiro tem somente 28 dias.

Entrada

Um número inteiro positivo com 8 dígitos.

Saída

O programa deve apresentar a transcrição da data no formato "dd de mês por extenso de aa".

Exemplo

Entrada
30022001
Saída
Data invalida!
Entrada
12092017
Saída
12 de setembro de 2017