Competidor(a): _____

Número de inscrição: _____–_____ *(opcional)*

*Este Caderno de Tarefas não pode ser levado para casa após a prova. Após a prova entregue este Caderno de Tarefas para seu professor guardar. Os professores poderão devolver os Cadernos de Tarefas aos competidores após o término do período de aplicação das provas (12 a 14 de Junho de 2024 ).*


## Olimpíada Brasileira de Informática

## OBI2023

# Caderno de Tarefas

## Modalidade Programação • Nível Júnior • Fase 1

12 a 14 de Junho de 2024

A PROVA TEM DURAÇÃO DE 2 horas

**Promoção:**


Sociedade Brasileira de Computação

**Apoio:**


**Coordenação:**

# Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 7 páginas (não contando a folha de rosto), numeradas de 1 a 7. Verifique se o caderno está completo.
- A prova deve ser feita individualmente.
- É proibido consultar a Internet, livros, anotações ou qualquer outro material durante a prova. É permitida a consulta ao *help* do ambiente de programação se este estiver disponível.
- As tarefas têm o mesmo valor na correção.
- A correção é automatizada, portanto siga atentamente as exigências da tarefa quanto ao formato da entrada e saída de seu programa; em particular, seu programa não deve escrever frases como "Digite o dado de entrada:" ou similares.
- Não implemente nenhum recurso gráfico nas suas soluções (janelas, menus, etc.), nem utilize qualquer rotina para limpar a tela ou posicionar o cursor.
- As tarefas **não** estão necessariamente ordenadas, neste caderno, por ordem de dificuldade; procure resolver primeiro as questões mais fáceis.
- Preste muita atenção no nome dos arquivos fonte indicados nas tarefas. Soluções na linguagem C devem ser arquivos com sufixo *.c*; soluções na linguagem C++ devem ser arquivos com sufixo *.cc* ou *.cpp*; soluções na linguagem Java devem ser arquivos com sufixo *.java* e a classe principal deve ter o mesmo nome do arquivo fonte; soluções na linguagem Python 3 devem ser arquivos com sufixo *.py*; e soluções na linguagem Javascript devem ter arquivos com sufixo *.js*.
- Na linguagem Java, **não** use o comando *package*, e note que o nome de sua classe principal deve usar somente letras minúsculas (o mesmo nome do arquivo indicado nas tarefas).
- Você pode submeter até 50 soluções para cada tarefa. A pontuação total de cada tarefa é a melhor pontuação entre todas as submissões. Se a tarefa tem sub-tarefas, para cada sub-tarefa é considerada a melhor pontuação entre todas as submissões.
- Não utilize arquivos para entrada ou saída. Todos os dados devem ser lidos da entrada padrão (normalmente é o teclado) e escritos na saída padrão (normalmente é a tela). Utilize as funções padrão para entrada e saída de dados:
  - em C: *scanf, getchar, printf, putchar*;
  - em C++: as mesmas de C ou os objetos *cout* e *cin*.
  - em Java: qualquer classe ou função padrão, como por exemplo *Scanner, BufferedReader, BufferedWriter* e *System.out.println*
  - em Python: *read,readline,readlines,input,print,write*
  - em Javascript: *scanf, printf*
- Procure resolver a tarefa de maneira eficiente. Na correção, eficiência também será levada em conta. As soluções serão testadas com outras entradas além das apresentadas como exemplo nas tarefas.

# Ogro

*Nome do arquivo:* `ogro.c`, `ogro.cpp`, `ogro.java`, `ogro.js` *ou* `ogro.py`

Ogro e Bicho-Papão têm fama de malvados, mas na verdade são amáveis, honestos e trabalhadores, além de vizinhos e amigos. O Bicho-Papão tem dificuldades em aprender aritmética e por isso o Ogro inventou uma brincadeira simples para auxiliar seu amigo: o Ogro inicia mostrando um certo número de dedos na sua mão esquerda (vamos chamar esse valor de $E$) e um número de dedos diferente na mão direita (vamos chamar esse valor de $D$). Então, Bicho-Papão deve falar o *resultado* da brincadeira, definido assim:

- se o número de dedos na mão esquerda é maior do que o número de dedos na mão direita (ou seja $E > D$) então o resultado é a soma dos dois números (ou seja $E + D$);
- caso contrário, o resultado é o dobro da diferença entre o número de dedos na mão direita e o número de dedos na mão esquerda (ou seja, $2 \times (D - E)$).

O problema é que o Ogro também não é lá muito bom em aritmética, e pediu sua ajuda para conferir se o Bicho-Papão falou a resposta correta.

Dados o número de dedos mostrados na mão esquerda ($E$) e o número de dedos mostrados na mão direita ($D$), escreva um programa para determinar a resposta da brincadeira.

## Entrada

A entrada é composta por duas linhas. A primeira linha contém um inteiro $E$, o número de dedos mostrados na mão esquerda. A segunda linha contém um inteiro $D$, o número de dedos mostrados na mão direita.

## Saída

Seu programa deve produzir uma única linha na saída, contendo um único número inteiro, o resultado da brincadeira.

## Restrições

- $0 \leq E \leq 5$
- $0 \leq D \leq 5$
- $E \neq D$

## Informações sobre a pontuação

A tarefa vale 100 pontos. Estes pontos estão distribuídos em subtarefas, cada uma com suas **restrições adicionais** às definidas acima.

- **Subtarefa 1 (0 pontos)**: Esta subtarefa é composta apenas pelos exemplos mostrados abaixo. Ela não vale pontos, serve apenas para que você verifique se o seu programa imprime o resultado correto para os exemplos.
- **Subtarefa 2 (30 pontos):** $E > D$.
- **Subtarefa 3 (70 pontos):** Sem restrições adicionais.

Seu programa pode resolver corretamente todas ou algumas das subtarefas acima *(elas não precisam ser resolvidas em ordem)*. Sua pontuação final na tarefa é a soma dos pontos de todas as subtarefas resolvidas corretamente por qualquer uma das suas submissões.

## Exemplos

| **Exemplo de entrada 1** | **Exemplo de saída 1** |
|---|---|
| 1<br>0 | 1 |

| **Exemplo de entrada 2** | **Exemplo de saída 2** |
|---|---|
| 2<br>5 | 6 |

# Concurso

*Nome do arquivo:* `consurso.c`, `consurso.cpp`, `consurso.java`, `consurso.js` *ou* `consurso.py`

Cláudia trabalha na OBI (Organização dos Bons Informáticos), que recentemente realizou um concurso para contratar novos funcionários. Agora, Cláudia tem a tarefa de determinar a *nota de corte* para o concurso. Chamamos de nota de corte a nota mínima necessária para ser aprovado no concurso. Ou seja, se a nota de corte do concurso for $C$, então todos os participantes com uma nota maior ou igual a $C$ serão aprovados no concurso e todos com nota menor que $C$ serão reprovados.

Seu chefe pediu para que Cláudia aprove no mínimo $K$ candidatos do concurso para a próxima fase, mas ela também não quer que a nota de corte seja muito baixa. Por isso, Cláudia decidiu que a nota de corte deverá ser a maior nota $C$ que faz com que no mínimo $K$ candidatos sejam aprovados.

Sua tarefa é: dados o número $N$ de candidatos, as notas $A_1, A_2, ..., A_N$ dos candidatos e a quantidade mínima de aprovados $K$, diga qual deve ser a maior nota de corte $C$ para que pelo menos K candidatos sejam aprovados.

## Entrada

A primeira linha da entrada contém dois inteiros, $N$ e $K$, representando, respectivamente, o número de participantes e o número mínimo de candidatos que devem ser aprovados.

A segunda linha da entrada contém $N$ inteiros $A_i$, representando as notas dos participantes.

## Saída

Seu programa deve imprimir uma linha contendo um único inteiro $C$, a nota de corte que deve ser escolhida por Cláudia.

## Restrições

- $1 \leq K \leq N \leq 500$
- $1 \leq A_i \leq 100$ para todo $1 \leq i \leq N$

## Informações sobre a pontuação

A tarefa vale 100 pontos. Estes pontos estão distribuídos em subtarefas, cada uma com suas **restrições adicionais** às definidas acima.

- **Subtarefa 1 (0 pontos)**: Esta subtarefa é composta apenas pelos exemplos mostrados abaixo. Ela não vale pontos, serve apenas para que você verifique se o seu programa imprime o resultado correto para os exemplos.
- **Subtarefa 2 (20 pontos):** $K = 1$.
- **Subtarefa 3 (20 pontos):** $K = 3$.
- **Subtarefa 4 (20 pontos):** $A_i \leq 2$.
- **Subtarefa 5 (40 pontos):** Sem restrições adicionais.

Seu programa pode resolver corretamente todas ou algumas das subtarefas acima *(elas não precisam ser resolvidas em ordem)*. Sua pontuação final na tarefa é a soma dos pontos de todas as subtarefas resolvidas corretamente por qualquer uma das suas submissões.

## Exemplos

| **Exemplo de entrada 1** | **Exemplo de saída 1** |
|---|---|
| 3 1<br>92 83 98 | 98 |

| **Exemplo de entrada 2** | **Exemplo de saída 2** |
|---|---|
| 4 2<br>1 2 3 4 | 3 |

| **Exemplo de entrada 3** | **Exemplo de saída 3** |
|---|---|
| 5 3<br>20 20 10 20 30 | 20 |

| **Exemplo de entrada 4** | **Exemplo de saída 4** |
|---|---|
| 10 5<br>1 2 2 1 2 2 1 1 1 1 | 1 |

# Bactérias

*Nome do arquivo:* `bacterias.c`, `bacterias.cpp`, `bacterias.java`, `bacterias.js` *ou* `bacterias.py`

Tadeu foi contratado recentemente para trabalhar no mais novo laboratório de biologia de sua universidade. Sua primeira tarefa consiste em estudar a taxa de reprodução de um determinado tipo de bactérias.

Tadeu iniciou seu experimento com uma única bactéria e descobriu que as bactérias se reproduzem a cada dia que passa. Mais especificamente, a cada dia, cada bactéria no recipiente de Tadeu se transforma em $P$ bactérias, onde $P$ é um inteiro positivo que Tadeu denominou como *fator de multiplicação* das bactérias.

Tadeu gostaria de obter a maior quantidade de bactérias possível, mas ele não pode deixar que essa quantidade ultrapasse a capacidade de $N$ bactérias do recipiente, pois isso invalidaria todo o seu experimento. Observe que a quantidade de bactérias *pode ser exatamente igual a* $N$ e isso não invalida o estudo de Tadeu.

Por exemplo, em um experimento cuja bactéria tem fator de multiplicação $P = 2$ e a capacidade do recipiente é $N = 32$:

- no início do experimento há apenas 1 bactéria;
- após 1 dia haverá $1 \cdot P = 2$ bactérias;
- após 2 dias haverá $2 \cdot P = 4$ bactérias;
- após 3 dias haverá $4 \cdot P = 8$ bactérias;
- após 4 dias haverá $8 \cdot P = 16$ bactérias;
- após 5 dias haverá $16 \cdot P = 32$ bactérias, o que excede a capacidade $N = 30$ do recipiente.

Nesse caso, Tadeu pode deixar as bactérias se reproduzindo por no máximo 4 dias.

Dados o fator de multiplicação $P$ e a capacidade $N$ do recipiente, ajude Tadeu a determinar durante quantos dias ele pode deixar as bactérias se reproduzindo sem que a quantidade de bactérias exceda a capacidade do recipiente.

## Entrada

A entrada é composta de duas linhas, cada uma contendo um único inteiro. A primeira linha contém $N$, a capacidade do recipiente. A segunda linha contém $P$, o fator de multiplicação das bactérias.

## Saída

Seu programa deverá imprimir uma única linha, contendo apenas um inteiro, a quantidade máxima de dias que Tadeu pode deixar as bactérias se reproduzindo sem invalidar seu experimento.

## Restrições

- $2 \leq P \leq N \leq 30\,000$

## Informações sobre a pontuação

A tarefa vale 100 pontos. Estes pontos estão distribuídos em subtarefas, cada uma com suas **restrições adicionais** às definidas acima.

- **Subtarefa 1 (0 pontos)**: Esta subtarefa é composta apenas pelos exemplos mostrados abaixo. Ela não vale pontos, serve apenas para que você verifique se o seu programa imprime o resultado correto para os exemplos.
- **Subtarefa 2 (50 pontos):** $P = 2$ e $N \leq 100$.
- **Subtarefa 3 (50 pontos):** Sem restrições adicionais.

Seu programa pode resolver corretamente todas ou algumas das subtarefas acima *(elas não precisam ser resolvidas em ordem)*. Sua pontuação final na tarefa é a soma dos pontos de todas as subtarefas resolvidas corretamente por qualquer uma das suas submissões.

## Exemplos

| **Exemplo de entrada 1** | **Exemplo de saída 1** |
|---|---|
| 30<br>2 | 4 |

| **Exemplo de entrada 2** | **Exemplo de saída 2** |
|---|---|
| 20000<br>5 | 6 |

| **Exemplo de entrada 3** | **Exemplo de saída 3** |
|---|---|
| 49<br>7 | 2 |