Modalidade Programação

Folha de Informações

1 Instruções Gerais

- A prova deve ser feita individualmente, na escola, sob supervisão.
- 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.
- É proibido utilizar ferramentas de Inteligência Artificial para resolver as tarefas.
- 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.
- Procure resolver as tarefas de maneira eficiente. Na correção, eficiência também será levada em conta. As soluções serão testadas com muitas outras entradas além das apresentadas como exemplo nas tarefas.

2 Ambiente de Prova

O Ambiente de Prova da OBI2025 (chamado CMS) é o mesmo utilizado na Olimpíada Internacional de Informática.

Consulte o professor coordenador local da OBI na sua escola para saber o endereço do Ambiente de Prova e a sua senha, que é diferente da senha de acesso a sua página pessoal na OBI2025 e foi enviada por email. A senha da prova somente o professor coordenador da OBI2025 na sua escola conhece.

No lado esquerdo da tela do Ambiente de Prova, para cada tarefa, você terá dois links: *Enunciado* e *Submissões*. Clique no link Enunciado para ver o enunciado da tarefa, clique no link Submissões para submeter uma solução para a tarefa.

Você pode submeter até 50 soluções para cada tarefa.

O tempo de prova é controlado pelo Ambiente de Prova e começa a contar a partir do momento que você clica o botão *Iniciar*.

3 Subtarefas

Na OBI uma subtarefa é um grupo de casos de testes com determinadas restrições de entrada. Cada subtarefa vale um conjunto de pontos; a soma dos pontos das subtarefas de uma dada tarefa é sempre 100.

Basicamente, você pode pensar que cada subtarefa é a divisão da tarefa em um problema "menor", potencialmente mais fácil de ser resolvido. Note que você pode escrever e submeter programas diferentes para resolver subtarefas diferentes, mas em geral é possível também escrever um único programa que resolve todas as subtarefas de uma dada tarefa.

Para cada subtarefa, sua solução é avaliada com múltiplos casos de teste, diferentes dos mostrados nos enunciados da tarefa, e para cada caso de teste sua solução recebe um número de pontos que depende do resultado da execução de seu programa com o caso de teste.

1. Para cada submissão a pontuação de cada subtarefa é o menor número de pontos obtido entre todos os casos de teste da subtarefa.
2. A pontuação final de cada subtarefa é a maior pontuação da subtarefa, entre todas as submissões.
3. A pontuação final de cada tarefa é a soma das pontuações finais de suas subtarefas.

Por exemplo, considere uma competidora que fez duas submissões para uma tarefa que contém duas subtarefas. Considere ainda que a primeira solução submetida recebeu 30 pontos para a primeira subtarefa e 10 pontos para a segunda subtarefa, e a segunda solução recebeu 0 pontos para a primeira subtarefa e 40 pontos para a segunda subtarefa. Então a pontuação final dessa competidora para essa tarefa é 70 pontos, ou seja, o máximo entre (30, 0), correspondente à primeira subtarefa, somado ao máximo entre (10, 40), correspondente à segunda subtarefa.

4 Resultado da correção

O ambiente de prova executa sua solução usando vários casos de testes, diferentes dos mostrados como exemplos nos enunciados das tarefas. Cada execução usa um caso de teste diferente e executa por um limite máximo de tempo e um limite máximo de uso de memória. Os limites de tempo e memória são estabelecidos para cada tarefa e cada linguaguem de programação.

Para resolver corretamente um caso de teste, sua solução deverá:

- Terminar a execução dentro do tempo limite estabelecido.
- Não exceder o limite de uso de memória estabelecido.
- Imprimir a resposta correta, **exatamente** no mesmo formato descrito na seção "Saída" do enunciado, e nada mais. Por exemplo, se a saída esperada para um caso de teste é "`3`", então o código precisa imprimir somente "`3`" para ser considerado correto (imprimir "`resposta = 3`", por exemplo, seria avaliado como resposta errada).

O resultado da correção de um caso de teste pode ser:

- *Correto*: se a solução produziu o resultado correto dentro dos limites de tempo e memória estabelecidos para a tarefa.
- *Incorreto*: se a solução

- produziu um resultado incorreto; ou
- foi interrompida por exceder o tempo limite ou o limite de memória estabelecido para a tarefa; ou
- foi interrompida por algum erro durante a execução (acesso inválido à memória, por exemplo).

Se a execução foi interrompida por algum erro durante a execução, nem sempre é possível para o ambiente de prova determinar o tipo de erro que causou a interrupção da execução. Quando o ambiente de prova não consegue determinar a causa do erro, ele emite uma mensagem de erro genérica: "A execução falhou pois o código de retorno foi diferente de zero".

5 Exemplos de soluções para diferentes linguagens

Nesta seção você encontra exemplos de soluções, para ilustrar como a entrada e saída para uma tarefa deve ser feita nas diferentes linguagens.

5.1 Python

Para submeter suas soluções em Python, você deve seguir as seguintes instruções:

- A entrada de dados é feita pela entrada padrão, a saída deve ser feita na saída padrão.
- Ao usar a função `input()` para ler a entrada, não use o parâmetro que escreve uma cadeia de caracteres, ou seja não use por exemplo assim: `input('Entre com o valor:')`. Isso faz com que a cadeia de caracteres "Entre com o valor:" seja escrita na saída do programa, fazendo com que a saída seja diferente do que o mostrado no exemplo do enunciado da tarefa.

Um pequeno exemplo de programa em Python

O exemplo abaixo lê dois valores inteiros dados em duas linhas e imprime o maior valor.

```
1  # exemplo de solucao em Python3
2
3  # le dois valores inteiros
4  val1 = input() # NÃO use input("Entre com um valor:")
5  val2 = input()
6
7  # escreve o resultado
8  if val1 > val2:
9    print(val1)
10 else:
11   print(val2)
```

5.2 C++

Para submeter suas soluções em C++, você deve seguir as seguintes instruções:

- A entrada de dados é feita pela entrada padrão, a saída deve ser feita na saída padrão. Utilize por exemplo `cin` ou a função `scanf` para ler da entrada padrão e `cout` ou a função `printf` para excrever na saída padrão.
- A função "main" de seu programa deve retornar o valor 0 ao final da execução.

Um pequeno exemplo de programa em C++

O exemplo abaixo lê dois valores inteiros e imprime o maior desses valores.

```cpp
#include<iostream>
using namespace std;

int val1, val2;

int main(void) {
  // lê as duas vals
  cin >> val1 >> val2;

  // escreve o resultado
  if (val1 > val2)
    cout << val1 << endl;
  else
    cout << val2 << endl;

  // termina a execução
  return 0;
}
```

5.3 C

Para submeter suas soluções em C, você deve seguir as seguintes instruções:

- A entrada de dados é feita pela entrada padrão, a saída deve ser feita na saída padrão. Utilize por exemplo a função `scanf` para ler da entrada padrão e a função `printf` para escrever na saída padrão.
- A função "main" de seu programa deve obrigatoriamente retornar o valor 0 ao final da execução.

Um pequeno exemplo de programa em C

O exemplo abaixo lê dois valores inteiros e imprime o maior desses valores.

```c
#include <stdio.h>

int val1, val2;

int main(void) {
  // lê as duas variáveis
  scanf("%d%d", &val1, &val2);

  // escreve o resultado
  if (val1 > val2)
    printf("%d\n", val1);
  else
    printf("%d\n", val2);

  // termina a execução
  return 0; // importante!
}
```

5.4 Java

Para submeter suas soluções em Java, você deve seguir as seguintes instruções:

- A entrada de dados é feita pela entrada padrão, a saída deve ser feita na saída padrão. Utilize qualquer classe ou função padrão para leitura, como por exemplo `Scanner` ou `BufferedReader` e qualquer classe ou função padrão para escrita, como por exemplo `System.out.println` `BufferedWriter`.

- Não use a diretiva package.
- O nome de sua classe principal pública e do seu arquivo `.java` deve ser obrigatoriamente o identificador da tarefa com todas as letras em minúsculo, sem acento. O identificador da tarefa é o nome que aparece entre parênteses no título da tarefa no ambiente de prova. Por exemplo, se o identificador da tarefa é "festa", o arquivo deve se chamar `festa.java` e a classe principal pública deve se chamar `festa`.

Um pequeno exemplo de programa em Java

O exemplo abaixo lê dois valores inteiros e imprime o maior valor.

```
import java.util.Scanner;

public class tarefa {

    public static void main(String[] args) {
          Scanner in = new Scanner(System.in);

          // lê duas variáveis inteiras
          int a = in.nextInt();
          int b = in.nextInt();

          // escreve o resultado
    if (a > b)
        System.out.printf("%d\n", a);
    else
        System.out.printf("%d\n", b);
      }
}
```

5.5 Javascript

Para submeter suas soluções em Javascript, você deve seguir as seguintes instruções:

- A entrada de dados deve ser feita utilizando o comando `scanf`, a saída deve ser feita utilizando o comando `printf`. Esses comandos foram desenvolvidos para o ambiente Saci e são muito parecidos com os comandos de mesmo nome na linguagem C.

Peça para o professor coordenador da OBI na sua escola para baixar o ambiente Saci para Javascritp da página da OBI.

Um pequeno exemplo de programa em Javascript

O exemplo abaixo lê dois valores inteiros e imprime o maior valor.

```
// exemplo de solução

// declara duas variáveis
var val1, val2;

// lê dois valores inteiros
scanf("%d%d", "val1", "val2");

// escreve o resultado
if (val1 > val2)
  printf("%d\n", val1);
else
  printf("%d\n", val2);
```