Warmup – Queijo Francês

2^a Seletiva Interna – 2018/2

UDESC-Joinville e IFMS-Aquidauana

Servidor BOCA (Arena Joinville):
http://200.19.107.67/boca/

Organizadores:

Claudio Cesar de Sá (coordenação geral), Peter Laureano Brendel (coordenação técnica), Daniela Marioti, Gabriel Hermann Negri , Lucas Hermann Negri (IFMS), Rogério Eduardo da Silva, Gilmário Barbosa dos Santos, Diego Buchinger, Roberto Rosso.

Patrocinador 2018: LINX

Lembretes:

- Aos *javaneiros*: **o nome da classe deve ser o mesmo nome do arquivo a ser submetido**. Ex: **classe petrus**, nome do arquivo **petrus.java**;
- Exemplo de leitura de entradas:

```
Java: Scanner in = new Scanner(new BufferedReader(new InputStreamReader(System.in)));
C: scanf("%d %d", &integer1, &integer2);
C++: int x; cin >> x;
```

  Exemplo de saída de entradas:

```
Java: System.out.println("asdf");
C: printf("%d %d", integer1, integer2);
C++: int x = 5; cout << x << endl;
```

- **É permitido consultar livros, anotações ou qualquer outro material impresso durante a prova**;
- A correção é automatizada, portanto, siga atentamente as exigências da tarefa quanto ao formato da entrada e saída de seu programa. Deve-se considerar entradas e saídas padrão;
- Todos os compiladores (Java, Python, C e C++) são padrões da distribuição Ubuntu versão 16.04 (gcc C11 habilitado);
- Procure resolver o problema de maneira eficiente. Se o tempo superar o limite pré-definido, a solução não é aceita. As soluções são testadas com outras entradas além das apresentadas como exemplo dos problemas;
- Teste seu programa antes de submetê-lo. A cada problema detectado (erro de compilação, erro em tempo de execução, solução incorreta, formatação imprecisa, tempo excedido ...), há penalização de 20 minutos. O tempo é critério de desempate entre duas ou mais equipes com a mesma quantidade de problemas resolvidos;
- Utilize o *clarification* para dúvidas da prova. Os juízes podem **opcionalmente** atendê-lo com respostas acessíveis a todos;
- Algumas interfaces estão disponíveis nas máquinas Linux, que podem ser utilizada no lugar da *Unity*. Para isto, basta dar *logout*, e selecionar a interface desejada. Usuário e senha: *udesc*;
- Ao pessoal local: **cuidado com os pés sobre as mesas para não desligarem nenhum estabilizador/computador de outras equipes**!
- O nome *Queijo Francês* é a sequência da *Pão de Queijo* de maio/2018 e uma homenagem a França, pela conquistas da Taça do Mundo. Nós ficamos para 2022!

Conteúdo

1 Problema A: *Amarelinha* para Calouros

`Arquivo: amarelinha.[c|cpp|java|py]` `Tempo limite: 2 s`

Todos já brincaram um dia de amarelinha. Aqui é um pouco diferente do usual, pois o jogo começará do fim, chamado de *céu*, e termina neste mesmo lugar, ver figura 1.

O básico é descobrir quantos saltos são necessários para o término do jogo, sendo que a pedrinha se encontre em uma célula (ou casa: 1 a 10) qualquer. Basicamente, o jogo consiste em sair do "céu", saltando de casa em casa, até onde se encontra a pedrinha, avançar a pedrinha de **uma casa** em direção ao "céu", e voltar para o "céu". Repetir este procedimento até a pedrinha alcançar o "céu". A marcação das casas do jogo de *marelinha* é a mesma do jogo tradicional. Ver figura 1:

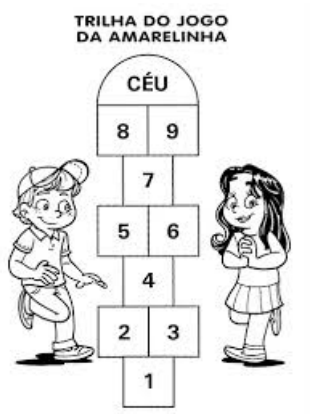

Figura 1: Clássico da Amarelinha

ou:

```
   [2]    [5]    [8]
[1]    [4]    [7]    [10]--(Céu)
   [3]    [6]    [9]
```

Quanto ao "céu" é o ponto de partida para esta "*amarelinha ao contrário*". O jogo se dá pelo avanço da pedrinha a partir de uma casa qualquer de 1 a 10, de acordo com a figura acima. Neste jogo modificado, o jogador começa e termina no "*céu*".

Se a pedrinha estiver na casa 10 basta o jogador ir até a casa 10, com um salto simples (só vale este tipo de avanço), pegar a pedrinha e arremessar para o céu e voltar! Neste caso, foram necessários 2 saltos e o jogo termina.

Caso a pedrinha esteja nas casas 8 ou 9 (o conceito é o mesmo para as casas 5 ou 6 e 2 ou 3), o jogador vai até esta casa, pega a pedrinha e avança para a casa seguinte, que no caso é a casa 10. Ou seja, a pedrinha sempre é arremessada para a casa subsequente, em direção ao "*céu*". Feito isso, o jogador deve sempre retornar ao céu e recomeçar a busca.

Assim, se a pedrinha está nas casas 8 ou 9, o jogador gastou 4 saltos para ir e voltar a partir do céu, e deixou a pedrinha na casa 10. Para o término do jogo, o jogador gastou estes 4 saltos para ir e para voltar até a casa 8 ou 9, mais os 2 saltos exigidos com a pedrinha na casa 10. Essas regras do jogo valem até a casa 1, a mais distante. Este processo se repete iterativamente até a casa 10; mas observe o tabuleiro, as células 2 e 3, 5 e 6, e 8 e 9, se encontram na mesma coluna ou equidistantes do "*céu*".

Problema

Sua tarefa é encontrar o número total de saltos dada a posição inicial X da pedrinha no tabuleiro.

Entrada

A entrada contém vários casos de teste. Um caso por linha. Em cada caso de testes é dado um valor X ($1 \leq X \leq 10$), que representa a posição inicial da pedrinha no tabuleiro. Os casos de testes se encerram com `-1`. Ou seja, `-1`, voce não lê mais nada e nem calcula.

Saída

Para cada caso de teste imprima um inteiro por linha, o qual corresponde a quantidade de saltos para finalizar o jogo.

Exemplo

Exemplo de Entrada	Exemplo de Saída
10 9 6 2 -1	2 6 20 42

2 Problema B: Bem que Combina

`Arquivo: combina.[c|cpp|java|py]` `Tempo limite: 2 s`

No mais novo jogo da Naointendo, Merio Odyssey, lançado em 2018, nosso querido carpinteiro finalmente pôde mudar a aparência clássica! Ao decorrer do jogo é possível concluir desafios e através deles liberar novas vestimentas.

Uma vestimenta consiste em um chapéu e uma camisa. Quando são liberados em um desafio, ambos são comuns a um tema.

JP é um jogador muito *hipster* e acredita que duas peças de mesmo tema não combinam e tornam Merio deselegante.

Problema

No início era fácil manter controle de quantas aparências diferentes e elegantes Merio poderia ter, mas após N desafios tornou-se inviável manter controle de todas as combinações possíveis. Para contornar este problema, Merio pediu para que você criasse um programa que mostre de quantas maneiras diferentes ele pode se manter elegante.

Entrada

A entrada consiste em um inteiro N ($1 \leq N \leq 2018$) que representa a quantidade de desafios concluídos por JP.

Saída

A saída esperada é a quantidade de combinações possíveis de vestimentas que não tornem Merio deselegante.

Exemplo de Entrada	Exemplo de Saída
1	0
2	2
5	20