



Departamento de Informática

Introdução à Programação

Concurso de Arqueologia

Enunciado do Primeiro Trabalho



Designed by macrovector / Freepik

Ano letivo 2021/22

1 Introdução

O trabalho é **individual** e é entregue no Mooshak, que aceita submissões até às **17h00** do dia **21 de novembro de 2021**. Leia este enunciado com a máxima atenção, para perceber muito bem o problema e todos os detalhes sobre o concurso do Mooshak e os critérios de avaliação do trabalho.

2 Conceitos e Objetivo do Trabalho

Uma *start-up* que se quer lançar no mercado das atividades ao ar livre está a preparar um *Concurso de Arqueologia*. O concurso real decorrerá num grande terreno e terá muitos *arqueólogos* (que são os concorrentes). Mas a versão experimental desenrola-se numa faixa estreita de terra, que se encontra dividida em *talhões* contíguos, como se ilustra na [Figura 1](#).

Os talhões são identificados pelo seu número de ordem (primeiro, segundo, etc.), que é atribuído de Oeste para Este (ou seja, o talhão da esquerda, mais a Oeste, é o 1º).

Quando o concurso começa, há *tesouros* enterrados nos talhões, que os arqueólogos vão tentar adquirir. Cada talhão tem um ou nenhum tesouro enterrado e cada tesouro tem um *valor*. A *riqueza enterrada* é a soma dos valores dos tesouros enterrados. Um arqueólogo adquire um tesouro ao escavar, pela primeira vez, o talhão em que o tesouro se encontra. Se um arqueólogo escavar um talhão que já foi escavado (por si ou por um adversário), já não pode encontrar um tesouro e sofrerá uma penalização. O *mérito* de um arqueólogo corresponde à soma dos valores dos tesouros que adquiriu subtraída das penalizações que sofreu. Quando um arqueólogo tenta escavar um talhão que não faz parte do terreno onde o concurso decorre, é automaticamente *desclassificado*, perdendo a *licença para escavar* mais talhões.

Para exemplificar, suponhamos que o concurso decorre num terreno com cinco talhões, iniciando-se com tesouros enterrados nos dois primeiros e no último (com valores 10, 5 e 20, respetivamente) e sem tesouro no terceiro e no quarto talhões (c.f. [Figura 2](#)). Inicialmente, a riqueza enterrada é 35 (porque $35 = 10 + 5 + 20$). Representando um talhão com tesouro por '*' e um talhão sem tesouro por '-', o *estado do terreno* é codificado por *--*--*.

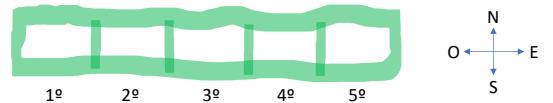


Figura 1: Terreno com 5 talhões

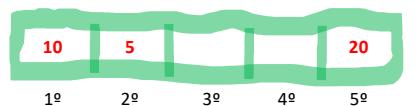


Figura 2: Tesouros enterrados

- Se a primeira escavação ocorrer no segundo talhão, o mérito do arqueólogo que o escavar passa a ser 5, a riqueza enterrada já é só 30 e o estado do terreno é *----*.
- Se a segunda escavação ocorrer no quarto talhão, o mérito do arqueólogo que o escavar, a riqueza enterrada e o estado do terreno não sofrem qualquer alteração. O mérito não sobe, porque o talhão não tinha tesouro, nem desce, porque o talhão ainda não tinha sido escavado.
- Se a terceira escavação ocorrer no segundo ou no quarto talhão, o mérito do arqueólogo que o escavar diminui, porque o talhão já tinha sido escavado. A riqueza enterrada e o estado do terreno permanecem inalterados.

O objetivo deste trabalho é programar em Java uma versão muito simplificada do Concurso de Arqueologia, na qual há apenas dois arqueólogos. Com a informação sobre o terreno, os tesouros enterrados, os arqueólogos e a sequência de escavações dos arqueólogos, o programa deve ser capaz de apresentar a riqueza enterrada, o estado do terreno e o mérito dos arqueólogos com licença para escavar, em qualquer momento do concurso (mesmo após o concurso ter acabado).

3 Regras do Concurso

No início do concurso, os dois arqueólogos encontram-se à esquerda do primeiro talhão, têm licença para escavar e mérito zero.

Um arqueólogo com licença para escavar pode sempre pedir para efetuar uma nova escavação, indicando o talhão que pretende escavar pelo “salto” que quer dar. O *salto* é a diferença entre a posição para onde quer ir e a posição onde se encontra. O salto pode ser positivo ou negativo (mas não pode ser nulo).

Suponhamos que o terreno tem cinco talhões e que um arqueólogo acabou de escavar o terceiro talhão. Se o pedido de escavação tiver salto 2, o arqueólogo quer (e vai) escavar o quinto talhão. Se, depois, pretender efetuar uma nova escavação com salto -1, o arqueólogo quer (e vai) escavar o quarto talhão.

Note que, quando o concurso começa, os arqueólogos não se encontram num talhão: estão à esquerda do primeiro talhão. Neste caso: se o salto for 1, pretendem escavar o primeiro talhão; se o salto for 2, pretendem escavar o segundo talhão; e assim sucessivamente. Se o salto for negativo ou superior ao número de talhões do terreno, o pedido de escavação é para um talhão que não existe.

Se um arqueólogo com licença para escavar quiser escavar um talhão que não existe, é desclassificado, abandonando imediatamente o recinto do concurso. Consequentemente, perde a licença para escavar e deixa de ter posição e mérito.

Enquanto um arqueólogo com licença para escavar quiser escavar um talhão que existe, escava-o e a sua posição passa a ser esse talhão, ocorrendo uma das três seguintes situações:

- Se houver um tesouro enterrado no talhão, o arqueólogo retira-o (e o talhão deixa de ter tesouro). O mérito do arqueólogo aumenta tantas unidades quanto o valor do tesouro.
- Se não houver um tesouro enterrado no talhão e o talhão ainda não tiver sido escavado, o talhão não sofre qualquer alteração (continua sem ter tesouro). O mérito do arqueólogo mantém-se.
- Se não houver um tesouro enterrado no talhão e o talhão já tiver sido escavado, o talhão não sofre qualquer alteração (continua sem ter tesouro). O mérito do arqueólogo diminui 10 unidades por cada escavação anterior realizada no talhão (por exemplo, se o talhão já tinha sido escavado uma vez, o mérito diminui 10 unidades; se o talhão já tinha sido escavado duas vezes, o mérito diminui 20 unidades; se o talhão já tinha sido escavado cinco vezes, o mérito diminui 50 unidades).

O concurso termina quando nenhum arqueólogo tiver licença para escavar ou nenhum arqueólogo com licença para escavar quiser efetuar mais escavações.

4 Especificação do Sistema

Pretende-se que a interface da aplicação seja simples, para poder ser utilizada em ambientes diversos e permitir automatizar o processo de teste. Por estes motivos, a entrada e a saída deverão respeitar o formato preciso especificado nesta secção. Em particular, poderá admitir que o input obedece às restrições de valor e de formato indicadas, ou seja, que o utilizador não comete erros, para além dos previstos neste enunciado.

O programa lê linhas da entrada padrão (`System.in`), escreve linhas na saída padrão (`System.out`) e distingue maiúsculas de minúsculas (por exemplo, as palavras “sair” e “Sair” são diferentes).

Forma do Input

A entrada tem a seguinte estrutura (onde o símbolo \leftrightarrow representa uma mudança de linha):

```

nomeArqueólogo1 ←
nomeArqueólogo2 ←
T ←
talhão1 talhão2 ... talhãoT ←
comando ←
comando ←
.....
comando ←
sair ←

```

onde:

- **nomeArqueólogo**₁ e **nomeArqueólogo**₂ são duas cadeias de caracteres diferentes, de comprimento entre 1 e 40 (possivelmente constituídas por várias palavras, como “Fulano Jr.”);
- **T** é um número inteiro entre 1 e 100;
- **talhão**₁, ..., **talhão**_T são **T** números inteiros entre 0 e 50 000, separados por um espaço;
- **comando** é um de quatro comandos (chamados *comando-riqueza*, *comando-terreno*, *comando-mérito* e *comando-escavação*) ou um comando inválido, explicados a seguir.

As duas primeiras linhas da entrada têm os nomes dos arqueólogos. A terceira linha tem o número de talhões (**T**) do terreno. A quarta linha indica que talhões têm tesouro e quais os seus valores: se **talhão**_i for um número positivo, há um tesouro enterrado no *i*-ésimo talhão cujo valor é **talhão**_i; se **talhão**_i for zero, o *i*-ésimo talhão não tem tesouro.

Segue-se um número arbitrário de comandos. A última linha tem um comando especial, o *comando-sair*, que só pode ocorrer na última linha porque faz terminar a execução do programa.

Comando-Riqueza

O comando-riqueza indica que se pretende saber a riqueza enterrada no momento corrente do concurso. Este comando não altera o estado do concurso. As linhas com comandos-riqueza têm:

```
riqueza←
```

Se *riquezaEnterrada* representar a riqueza enterrada no momento corrente do concurso, o programa deve escrever uma linha na consola com:

```
Riqueza enterrada: riquezaEnterrada←
```

Comando-Terreno

O comando-terreno indica que se pretende visualizar o estado do terreno no momento corrente do concurso. Este comando não altera o estado do concurso. As linhas com comandos-terreno têm:

```
terreno←
```

O estado do terreno é uma sequência de caracteres com comprimento igual ao número de talhões do terreno. O *i*-ésimo carácter da sequência é '*' (asterisco), se o *i*-ésimo talhão tiver um tesouro enterrado; e é '-' (sinal menos), no caso contrário. Se *estadoTerreno* denotar o estado do terreno no momento corrente do concurso, o programa deve escrever uma linha na consola com:

```
estadoTerreno←
```

Comando-Mérito

O comando-mérito indica que se pretende saber o mérito do arqueólogo com o nome dado, no momento corrente do concurso. Este comando não altera o estado do concurso. As linhas com comandos-mérito têm a forma (com um espaço a separar as duas componentes):

```
merito nomeArqueólogo←
```

onde:

- *nomeArqueólogo* é uma cadeia de caracteres de comprimento entre 1 e 40.

O programa escreve uma linha na consola, distinguindo três casos:

- Se *nomeArqueólogo* não for o nome de um dos arqueólogos, a linha tem:

```
Arqueólogo inexistente←
```

- Se *nomeArqueólogo* for o nome de um arqueólogo desclassificado, a linha tem:

```
Arqueólogo desclassificado←
```

- Nos restantes casos, a linha tem a seguinte forma, onde *méritoArqueólogo* representa o mérito do arqueólogo referido no comando (que pode ser positivo, zero ou negativo):

```
Merito de nomeArqueólogo: méritoArqueólogo←
```

Comando-Escavação

O comando-escavação indica que, no momento corrente do concurso, o arqueólogo com o nome dado quer escavar o talhão especificado pelo salto dado. As linhas com comandos-escavação têm a seguinte forma (com um espaço a separar componentes consecutivas):

```
escavacao salto nomeArqueólogo←
```

onde:

- *salto* é um número inteiro;
- *nomeArqueólogo* é uma cadeia de caracteres de comprimento entre 1 e 40.

O programa deve usar esta informação para atualizar o estado do concurso, exceto nos três casos seguintes, nos quais o estado do concurso não muda e o programa escreve uma linha:

- Se *salto* for o número zero, a linha tem:

```
Salto invalido←
```

- Se *salto* não for zero e *nomeArqueólogo* não for o nome de um dos arqueólogos, a linha tem:

```
Arqueólogo inexistente←
```

- Se *salto* não for zero e *nomeArqueólogo* for o nome de um arqueólogo desclassificado, a linha tem:

```
Arqueólogo desclassificado←
```

Nas restantes situações, o programa não deve escrever qualquer resultado, exceto se o arqueólogo for desclassificado em consequência deste pedido de escavação. Neste último caso, o programa deve escrever uma linha com:

```
nomeArqueólogo perdeu a licenca de escavacao←
```

Comando-Sair

O comando-sair indica que se pretende terminar a execução do programa. A linha com o comando-sair tem:

sair←

O programa termina, escrevendo uma linha na consola. Distinguem-se três casos:

- Se nenhum arqueólogo tem licença para escavar, a linha tem:

Correu mal! Foram ambos desclassificados.←

- Se algum arqueólogo tem licença para escavar e ainda há algum tesouro enterrado, a linha tem:

Ainda havia tesouros por descobrir...←

- Se algum arqueólogo tem licença para escavar e não há qualquer tesouro enterrado, a linha tem:

Todos os tesouros foram descobertos!←

Comandos Inválidos

Sempre que o utilizador escrever uma linha que não comece com as palavras “riqueza”, “terreno”, “merito”, “escavacao” ou “sair”, o estado do concurso não deve ser alterado e o programa deve escrever uma linha com:

Comando invalido←

5 Exemplos

Apresentam-se alguns exemplos. A coluna da esquerda ilustra a interação: o input está escrito a azul e o output a preto. Todas as linhas do input e do output terminam com o símbolo de mudança de linha, que se omitiu para aumentar a legibilidade. A coluna da direita tem informação para o leitor do enunciado, servindo apenas para relembrar as regras descritas anteriormente. Nessa coluna, “T” abrevia “talhão” e “M” abrevia “o seu mérito”.

Exemplo 1

Fulano	Nome de um arqueólogo; M é 0.
Beltrano	Nome do outro arqueólogo; M é 0.
5	O terreno tem 5 talhões.
10 5 0 0 20	1º, 2º e 5º talhões têm tesouro.
riqueza	
Riqueza enterrada: 35	
terreno	

escavacao 2 Beltrano	Beltrano escava 2º T; M sobe 5.
terreno	

merito Beltrano	
Merito de Beltrano: 5	
sair	
Ainda havia tesouros por descobrir...	

Exemplo 2

Fulano
Beltrano
5
10 5 0 0 20
escavacao 3 Beltrano
riqueza
Riqueza enterrada: 35
terreno

escavacao 5 Fulano
escavacao -3 Fulano
riqueza
Riqueza enterrada: 10
terreno

escavacao 2 Beltrano
merito Beltrano
Merito de Beltrano: -10
merito Fulano
Merito de Fulano: 25
escavacao -3 Beltrano
merito Beltrano
Merito de Beltrano: -20
escavacao 3 Beltrano
merito Beltrano
Merito de Beltrano: -40
escavacao -4 Beltrano
merito Beltrano
Merito de Beltrano: -30
terreno

riqueza
Riqueza enterrada: 0
escavacao 0 Sicrano
Salto invalido
escavacao 3 Sicrano
Arqueologo inexistente
escavacao 4 Fulano
Fulano perdeu a licenca de escavacao
merito Fulano
Arqueologo desclassificado
Merito Fulano
Comando invalido
escavacao -2 Fulano
Arqueologo desclassificado
merito Beltrano
Merito de Beltrano: -30
sair
Todos os tesouros foram descobertos!

Nome de um arqueólogo; **M** é 0.
Nome do outro arqueólogo; **M** é 0.
O terreno tem 5 talhões.
1º, 2º e 5º talhões têm tesouro.
Beltrano escava 3º **T**; **M** mantém-se.

Fulano escava 5º **T**; **M** sobe 20.
Fulano escava 2º **T**; **M** sobe 5.

Beltrano escava 5º **T**, já escavado 1 vez; **M** desce 10.

Beltrano escava 2º **T**, já escavado 1 vez; **M** desce 10.

Beltrano escava 5º **T**, já escavado 2 vezes; **M** desce 20.

Beltrano escava 1º **T**; **M** sobe 10.

Fulano quer escavar **T** inexistente; é desclassificado.

Exemplo 3

```
Fulano Sicrano
Beltrano Jr.
7
40 101 23 11 2 37 101
riqueza
Riqueza enterrada: 315
terreno
*****
escavacao 1 Beltrano
Arqueologo inexistente
escavacao 1 Beltrano jr.
Arqueologo inexistente
escavacam 1 Beltrano Jr.
Comando invalido
escavacao 4 Fulano Sicrano
escavacao 7 Beltrano Jr.
terreno
***-***
riqueza
Riqueza enterrada: 203
bife com batatas fritas
Comando invalido
MERITO
Comando invalido
escavacao -4 Fulano Sicrano
Fulano Sicrano perdeu a licenca de escavacao
escavacao -6 Beltrano Jr.
escavacao 1 Beltrano Jr.
merito Beltrano Jr.
Merito de Beltrano Jr.: 242
terreno
---*---*
riqueza
Riqueza enterrada: 62
escavacao 2 Beltrano Jr.
merito Beltrano Jr.
Merito de Beltrano Jr.: 232
escavacao -5 Beltrano Jr.
Beltrano Jr. perdeu a licenca de escavacao
merito Fulano Sicrano
Arqueologo desclassificado
merito Beltrano Jr.
Arqueologo desclassificado
terreno
---*---*
riqueza
Riqueza enterrada: 62
sair
Correu mal! Foram ambos desclassificados.
```

Nome de um arqueólogo; **M** é 0.
Nome do outro arqueólogo; **M** é 0.
O terreno tem 7 talhões.
Todos os talhões têm tesouro.

Fulano Sicrano escava 4º **T**; **M** sobe 11.
Beltrano Jr. escava 7º **T**; **M** sobe 101.

Fulano Sicrano quer escavar **T** inexistente;
é desclassificado.
Beltrano Jr. escava 1º **T**; **M** sobe 40.
Beltrano Jr. escava 2º **T**; **M** sobe 101.

Beltrano Jr. escava 4º **T**, já escavado 1 vez;
M desce 10.

Beltrano Jr. quer escavar **T** inexistente;
é desclassificado.

6 Entrega do Trabalho

O trabalho é entregue no [Mooshak](#). Deverá submeter um arquivo `.zip` ao Problema A do concurso **IP2122-P1**. Não se esqueça que:

- O arquivo deve conter apenas todos os ficheiros `.java` que tiver criado para resolver o problema.
- O arquivo tem necessariamente de conter um ficheiro `Main.java`, onde está o método `main`.
- A classe `Main` tem de estar na raiz do arquivo (tem de pertencer ao pacote principal (`default`)).
- A versão de Java instalada no Mooshak é a 8.¹

Cada aluno tem de se registar no concurso IP2122-P1. O **nome do utilizador** tem de ser o seu **número de aluno** e o **grupo** é o do seu **turno das aulas práticas**. Por exemplo, o aluno com o número 76543, inscrito no turno P3, é o utilizador com o nome 76543 e pertence ao grupo P3. Só serão considerados entregues (e avaliados) os programas dos utilizadores no concurso IP2122-P1 que respeitem estas regras. O concurso IP2122-P1 abre no dia 11 de novembro e encerra às **17h00** do dia **21 de novembro de 2021** (domingo). Pode re-submeter o trabalho as vezes que entender, até à hora limite de submissão. Apenas será avaliado o programa que se encontrar no arquivo enviado na **última submissão** que fizer.

7 Critérios de Avaliação do Trabalho

De acordo com o [Regulamento de Avaliação de Conhecimentos da FCT NOVA](#):

- Existe fraude quando:
 - (a) Se utiliza ou tenta utilizar, sob qualquer forma, num teste, exame, ou outra forma de avaliação, presencial ou a distância, informação ou equipamento não autorizado;
 - (b) Se presta ou recebe colaboração não autorizada na realização dos exames, testes, ou qualquer outra prova de avaliação de conhecimentos individuais;
 - (c) Se presta ou recebe colaboração, não permitida pelas regras aplicáveis a cada caso, na realização de trabalhos práticos, relatórios ou outros elementos de avaliação.
- Os estudantes diretamente envolvidos numa fraude são liminarmente reprovados na disciplina.

Em IP, o documento [Colaboração Permitida e Não Permitida](#), disponibilizado no Moodle, clarifica as alíneas transcritas acima. Os alunos que cometem fraude num trabalho não obterão frequência.

A avaliação do trabalho tem duas componentes independentes, cujas notas se somam para obter a nota do trabalho:

- **Funcionalidade** (correção dos resultados produzidos): **12 valores**

Um programa submetido ao concurso que obtenha a pontuação máxima (120 pontos) terá 12 valores. Um programa que não implemente corretamente todas as funcionalidades e obtenha P pontos no Mooshak, terá $P/10$ valores.²

¹Esta informação é irrelevante se só usar as instruções de Java dadas nas aulas porque, nesse caso, o programa deverá ter o mesmo comportamento na sua máquina e no Mooshak.

²O trabalho pode ser realizado incrementalmente. Por exemplo, pode começar por assumir que os nomes dos arqueólogos só têm uma palavra (a sequência de caracteres não tem espaços). É claro que, enquanto o programa não produzir os resultados corretos em todos os testes do Mooshak, não obterá 120 pontos.

- **Qualidade do código: 8 valores³**

Um código com qualidade deve ter, entre outras, as seguintes características:

- **Várias classes que caracterizem bem as diferentes entidades do problema;**
- Classes, métodos, variáveis e constantes com objetivos bem definidos e as restrições de acesso apropriadas;
- Algoritmos simples e bem estruturados, implementados com as instruções mais adequadas;
- Identificadores que expressem os conceitos que representam, escritos de acordo com as convenções ensinadas (por exemplo, o nome de uma classe deve ser um substantivo que comece com uma letra maiúscula);
- Precondições nos métodos;
- Indentação correta (lembre-se do comando CTRL-SHIFT-F do Eclipse), linhas com 100 caracteres (no máximo)⁴ e métodos com 25 linhas (no máximo);
- Um comentário no início de cada classe, que indique o que os objetos da classe representam, e um comentário antes de cada método, que explique resumidamente o que o método faz.

Bom trabalho!

³Note que a qualidade do código tem um peso muito grande na nota do trabalho.

⁴Na contagem do número de caracteres de uma linha de código, considere que um tab equivale a 4 caracteres.