Programação Orientada a Objetos [BCC221]

Atividade Prática

Onde estão os *Panzers*?

Departamento de Computação
Universidade Federal de Ouro Preto
Prof. José Romildo Malaquias
27 de outubro de 2015

Resumo

Nesta atividade vamos implementar um programa para decodificar mensagens de guerra. Estaremos aplicando conceitos básicos de programação usando a linguagem Java.

Sumário

1 O Problema

O ano é 1944 e o Brasil está na Segunda Guerra Mundial. Você está em uma equipe dos Aliados encarregada de decodificar mensagens do Eixo, sob o comando do General Morais. O único dispositivo que sua equipe possui é uma antena que consegue captar ondas de rádio enviadas de um centro do Eixo para sua respectiva força aérea. Sua equipe recebe vários sinais na antena, mas não consegue entender nada do que se passa. Para a sorte de sua equipe, o espião Fonseca conseguiu reunir algumas informações importantes. A primeira informação é sobre como decodificar as ondas de rádios recebidas. Fonseca revelou que o Eixo utiliza de um sistema representado por 0's e 1's e estes bits estão codificados nas ondas de rádio. Ele mostrou como recuperar esses valores das ondas de rádio e como interpretá-los. Para a surpresa de sua equipe, o que estava sendo transmitido pelo canal eram caracteres alfanuméricos. Como uma amostra, sua equipe decodificou algumas sequências de ondas de rádio. Um dos exemplos amostrados foi:

```
jahe?! $paeaoaew11iawwnaaqtaaa(aqw1o0o0,jajaja9aq0qqq1aaa)xx
```

O espião Fonseca, como um dos melhores espiões dos aliados, conseguiu as instruções para extrair a informação dessa sequência de caracteres. Ele mostrou que a mensagem continha a coordenada $(100, 901)$ como alvo de ataque, conforme mostrado abaixo:

```
jahe?! $paeaoaew11iawwnaaqtaaa(aqw1o0o0,jajaja9aq0qqq1aaa)xx
```

Um documento roubado pelo espião mostra as regras para extrair as coordenadas. As regras são descritas abaixo:

- Uma coordenada sempre virá precedida da expressão `p[o]int`, minúscula. Essa expressão indica que os caracteres que estão fora do colchete aparecem apenas uma vez, na ordem mostrada. O caractere `o`, entretanto, pode aparecer uma ou mais vezes. Dessa forma, as strings `point` e `pooooint` são válidas mas `pint` não é.
- A coordenada do ataque é no formato de coordenadas cartesianas (x, y), onde x e y são números inteiros. Além disso, você deve ignorar 0's a esquerda. Por exemplo, 0050 representa o número 50.

- O processo para identificar as coordenadas ocorre em duas etapas: descobrir a expressão `p[o]int` e descobrir a coordenada (x, y).
- Na primeira etapa, a expressão `p[o]int` pode estar com ruídos entre seus caracteres. O ruído pode ser qualquer caracter da tabela ascii[1] e que seja imprimível.
- Não existe um número definido de ruídos entre as coordenadas. Neste caso, esses ruídos devem ser ignorados a fim de recuperar as coordenadas.
- Caracteres da expressão podem virar ruídos após serem lidos, como o exemplo `pooooiooont(20,30)`. Após ler o caractere `p`, são lidos 4 caracteres `o` e um caracter `i`. Os caracteres `o` são ruídos, pois a meta é encontrar o caractere `n`. Portanto eles são desconsiderados. Uma implicação disso é que eles não são considerados para a contagem de tanques.
- Uma vez que a palavra `point` seja descoberta, inicia-se a segunda etapa de descobrir a coordenada. Enquanto um parêntese de abertura (`(`) não for encontrado, a segunda etapa não inicia-se. A segunda etapa é invalidada e o processo voltará para o início da primeira etapa se:
  - for lido mais de um parêntese de abertura (`(`).
  - for lido mais de uma vírgula (`,`).
  - o o valor máximo da coordenada for ultrapassado.
- Assim que ocorrer uma violação em uma das regras acima, o processo deve ser resetado, ou seja, deve-se começar a procura da letra `p` novamente.
- Podem haver ruídos entre os algarismos dos números inteiros x e y bem como entre o padrão (x, y). Esses ruídos deverão ser desconsiderados.
- Uma mesma coordenada pode aparecer mais de uma vez, o que indica que houve mais de um ataque a esta mesma coordenada. Além disso, o número de tanques em cada uma dessas aparições pode variar.

A sua função é criar uma solução que extraia as coordenadas de uma sequência de caracteres, bem como o número de tanques Panzer que serão designados para aquela coordenada. Para identificar o número de tanques Panzer que serão utilizadas no ataque, o espião Fonseca descobriu que é o número de `o` da expressão `p[o]int` que identifica a coordenada. Dessa forma, `pooooint(50,40)` indica que a coordenada $(50, 40)$ será alvo de 4 tanques Panzer enquanto que `point(30,11)` indica que a coordenada $(30, 11)$ será alvo de apenas um tanque. Após identificar as coordenadas do alvo do ataque, você deve calcular a distância euclidiana desse alvo (A) até à sua base (B) (vide Entrada e Saída).

Relembrando distância euclidiana...

A distância euclidiana entre os pontos $A = (x_A, y_A)$ e $B = (x_B, y_B)$ é definida como

$$\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2)}$$

2 Entrada e Saída

Seu programa deverá ler a entrada de um arquivo texto cujo nome é indicado como argumento na linha de comando, e imprimir a saída na saída padrão. A entrada de teste é composta apenas por um caso de teste. A primeira linha da entrada contém 2 inteiros, separados por espaço, que correspondem, respectivamente, aos valores x e y da coordenada de sua base. As próximas linhas são compostas um número não definido de caracteres imprimíveis da tabela ASCII.

A saída deverá ser composta de alguns pares T;C, onde T é o número de tanques Panzer que atacará a coordenada C, sendo C uma coordenada com formato (x,y); e de um valor D que é a distância euclidiana entre o alvo do ataque e a sua base (use 2 casas decimais para representar a

[1] http://pt.wikipedia.org/wiki/ASCII

distância). Se não houver nenhuma coordenada, a saída deverá ser deixada em branco, ou seja, nenhum caractere deverá ser impresso. Ou seja, a primeira linha da saída contém um par T1;(x1,y1); a segunda linha contém a distância D dessa coordenada (x1,y1) até à coordenada base; a terceira linha deve ser deixada em branco; a partir da quarta linha, repete-se a sequência citada anteriormente, mas para o par T2;(x2,y2) e assim sucessivamente...

entrada

```
15 15

uehuaheuahuhaupkkkkokkkikkknnkkktk(8i0,9p0)uahauheauheuaheuh
ueauehuaheuahuhaukkk21321kkkikkknkkktk(8i0000!!!!0000000,9a0
)uahauheauheuaheuhueapoint(10,20)pooooiont(20,30)poooaaaooooooo
ooint(30,40)poo??oint(40,50)uakkkakakijeuquueuhupoint(point100,20
0)[[[[[]]]]]point(,30)point(100,100)%42111point(11point(11,11))11:1
9point(000000010,1)
```

saída

```
1;(80,90)
99.25

1;(10,20)
7.07

4;(20,30)
15.81

11;(30,40)
29.15

3;(40,50)
43.01

1;(100,200)
203.59

1;(100,100)
120.21

1;(10,1)
14.87
```