



Programa IT Academy – Processo Seletivo – Edição #16

Etapa 2 – Enunciado

Nesta etapa, você vai escrever um programa de computador. Para isso deve ser feita a leitura do arquivo .csv enviado junto com este enunciado. Neste arquivo você encontra dados sobre medicamentos disponíveis no Brasil. Você deve implementar as seguintes funcionalidades:

1. [Consultar medicamentos pelo nome] Permitir que o usuário informe o nome do medicamento (ou parte do nome do medicamento) que deseja e como resultado o programa deverá exibir:
 - a. Uma lista com os medicamentos encontrados e suas informações (Nome, Produto, Apresentação e valor PF Sem Impostos);
Atenção: somente devem aparecer no resultado os registros de produtos que foram comercializados em 2020 (observar a coluna de dados “COMERCIALIZAÇÃO 2020”).
2. [Buscar pelo código de barras] O programa deverá solicitar ao usuário o número correspondente ao código de barras de um produto (coluna de dados “EAN 1”, por exemplo ‘525516020019503’) e então:
 - a. Localizar todos os registros referentes a este produto, independentemente de terem sido comercializados ou não em 2020;
 - b. Dentre todos os registros encontrados, identificar o Preço Máximo ao Consumidor (alíquota de 0%, coluna de dados “PMC 0%”) mais alto e o mais baixo. Exibir na tela o mais alto, o mais baixo e a diferença entre eles.
3. [Comparativo da LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)] Com base somente nos produtos que foram comercializados em 2020, o programa deverá:
 - a. Consultar a coluna de dados “LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)” para determinar o percentual de produtos classificados como “Negativa”, “Neutra” ou “Positiva” para esta coluna.
 - b. Mostrar os respectivos valores percentuais da seguinte maneira (dados fictícios):
[repare que a quantidade de asteriscos é proporcional ao respectivo percentual, por exemplo, neste caso são 21 asteriscos para a classificação Negativa.]*

CLASSIFICACAO	PERCENTUAL	GRAFICO
Negativa	21,33%	*****
Neutra	45,18%	*****
Positiva	33,49%	*****
TOTAL	100,00%	

Observações:

- a) Sugere-se o desenvolvimento de um programa **na linguagem de sua preferência**, com uma **interface também de sua preferência podendo ser gráfica ou textual/console**, com um **menu com as opções enumeradas nos requisitos**;
- b) Juntamente a este enunciado foi fornecido um arquivo no formato CSV contendo nomes, valores em decimais, bem como o respectivo dicionário de dados;
- c) Você deve escrever o código que lê o arquivo e armazena os dados lidos em memória (do jeito que você quiser).
- d) Não é necessário gravar dados em nenhum formato, nem usar sistemas de banco de dados.
- e) O programa deverá lidar com dados de entrada inválidos e informar uma mensagem adequada caso ocorram.
- f) Para facilitar, não é necessário lidar com a acentuação de palavras.
- g) Na escrita do relatório apresente comentários sobre como você realizou os testes. Não esqueça de incluir uma autoavaliação.

Explicação da Solução:

Para solucionar o problema, foram criadas quatro Classes principais, cada uma visando abordar um ponto específico do problema. Além destas, existem três classes auxiliares, uma das quais não faz parte do programa final. A pasta com a solução foi dividida em duas sub-pastas; a primeira (testes) foi criada com o intuito de fazer a leitura e análise do arquivo – como estavam distribuídas as informações na tabela, quais colunas continham quais informações e etc; a segunda, que contém o programa final, é dividido em um Main (classe principal, que chama o método execute da Classe App, que por sua vez contém três chamadas para as outras classes – cada uma de acordo com um dos pontos do exercício). Para ler o arquivo, utilizei a Classe Scanner do java.util e a Classe File do java.io, depois iterei sobre o arquivo, quebrando cada linha lida no Scanner em arranjos unidimensionais. A localização do arquivo e o fato de ele ser separado nos pontos e vírgula (;) estão definidos em um arquivo externo as pastas denominado “CsvConfig.java”, que exporta o caminho e o separador para todas as outras classes. As três classes das etapas dos exercícios foram resolvidas de modos muito similares, possuindo um método principal chamado pelo App, que percorre a tabela e procura pelos valores de acordo com as condições do exercício e classes internas que executam funções – melhores explicadas no código – específicas para apresentar cada informação na tela.

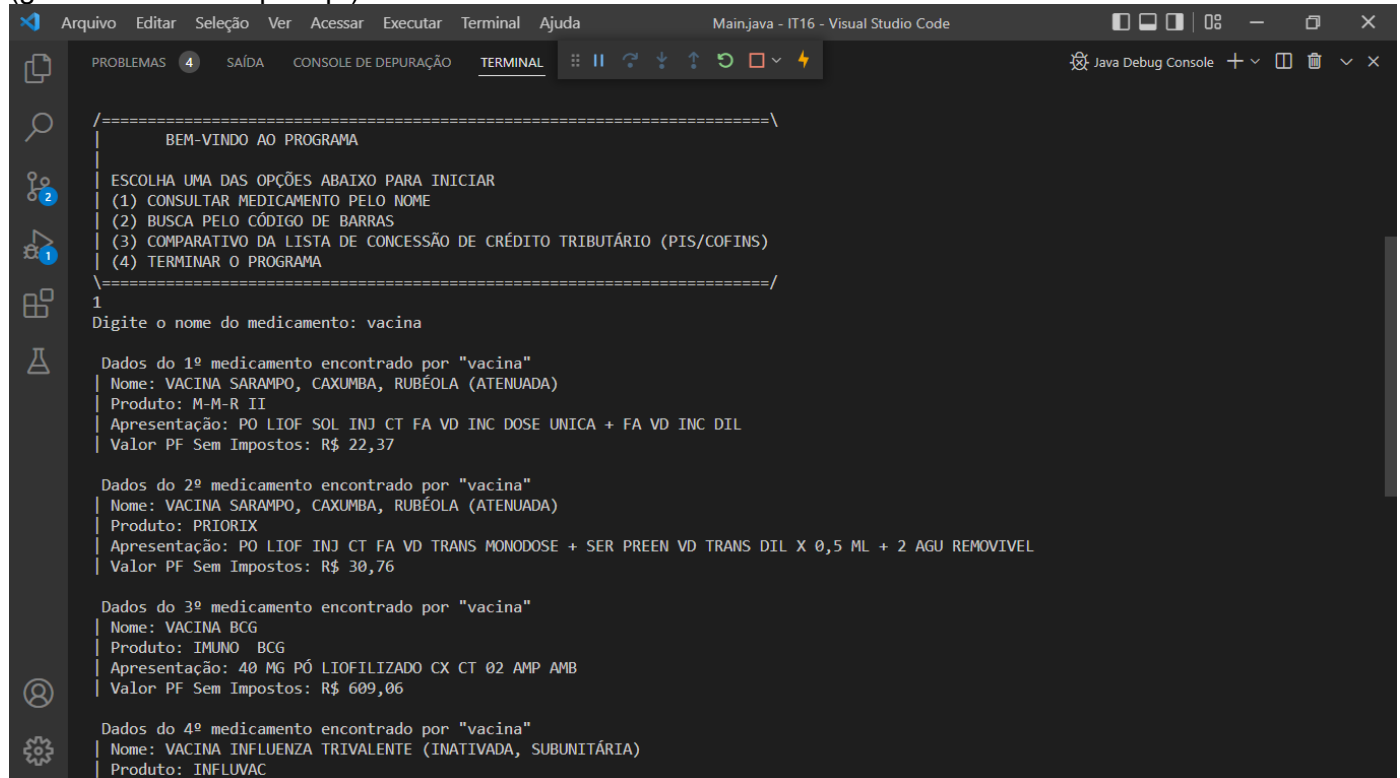
Explaining of the Solution:

In order to solve the problem, four main classes were created, each one addressing a specific bullet point of the problem. In addition to these, there are three auxiliary classes, one of which is not a part of the final program. The folder with the solution was divided into two sub-folders; the first (“testes”) was created with the aim of reading and analyzing the file – how the data was distributed in the table and which columns contained which information; the second, that contains the final program, is divided into a Main (the most important class, which calls the execute method from the App Class, which in turn includes three calls to the other classes – each one according to one of the bullet points of the exercise). To read the file, I used the Scanner Class from java.util and the File Class from java.io, then I iterate over the file, breaking each line read by the Scanner into one-dimensional arrays. The location of the file and the fact that it is separated by semicolons (;) rather than the usual comma (,) are in an external file to the two main folders named “CsvConfig.java”, which exports the path and separator for all the other classes. All the three questions were solved in a very similar way, each Class having a main method called by the App according to a user input, which goes over the table and its values according to the conditions of the exercise and each has some more inner methods that perform functions – better explained in the code – which are specific to presenting each piece information on the screen.

Capturas de tela demonstrando a execução da solução e seus resultados: Screenshots portraying the execution of the solution and its results:

A primeira imagem mostra a primeira questão, que busca um medicamento pelo nome, e alguns dos resultados dado o input “vacina”.

The first picture shows the first question, to find a medicine according to its name, and some of its results (given the “vacina” prompt).



```
Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda Main.java - IT16 - Visual Studio Code
PROBLEMAS 4 SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL Java Debug Console

/=====
| BEM-VINDO AO PROGRAMA
|
| ESCOLHA UMA DAS OPÇÕES ABAIXO PARA INICIAR
| (1) CONSULTAR MEDICAMENTO PELO NOME
| (2) BUSCA PELO CÓDIGO DE BARRAS
| (3) COMPARATIVO DA LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)
| (4) TERMINAR O PROGRAMA
|=====
1
Digite o nome do medicamento: vacina

Dados do 1º medicamento encontrado por "vacina"
| Nome: VACINA SARAMPO, CAXUMBA, RUBÉOLA (ATENUADA)
| Produto: M-M-R II
| Apresentação: PO LIOF SOL INJ CT FA VD INC DOSE UNICA + FA VD INC DIL
| Valor PF Sem Impostos: R$ 22,37

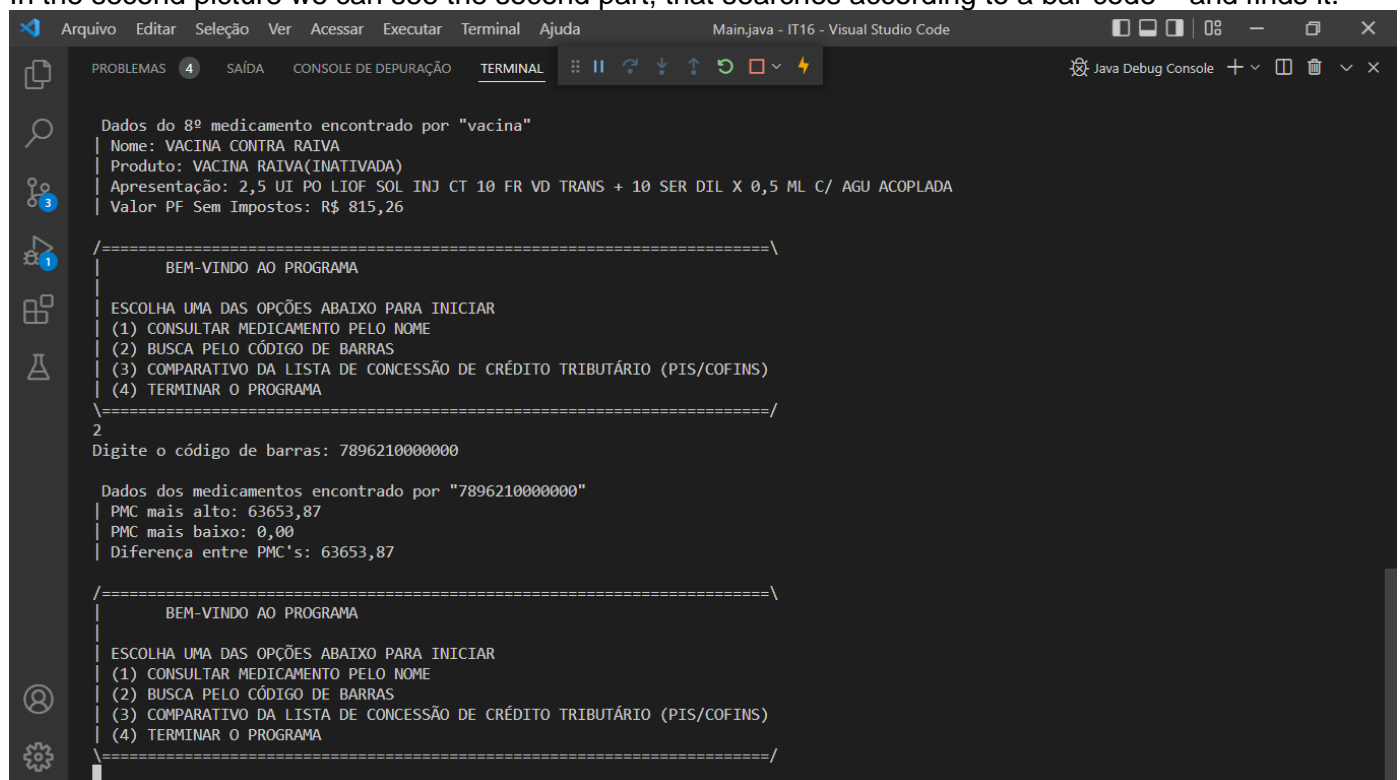
Dados do 2º medicamento encontrado por "vacina"
| Nome: VACINA SARAMPO, CAXUMBA, RUBÉOLA (ATENUADA)
| Produto: PRIORIX
| Apresentação: PO LIOF INJ CT FA VD TRANS MONODOSE + SER PREEN VD TRANS DIL X 0,5 ML + 2 AGU REMOVIVEL
| Valor PF Sem Impostos: R$ 30,76

Dados do 3º medicamento encontrado por "vacina"
| Nome: VACINA BCG
| Produto: IMUNO BCG
| Apresentação: 40 MG PÓ LIOFILIZADO CX CT 02 AMP AMB
| Valor PF Sem Impostos: R$ 609,06

Dados do 4º medicamento encontrado por "vacina"
| Nome: VACINA INFLUENZA TRIVALENTE (INATIVADA, SUBUNITÁRIA)
| Produto: INFLUVAC
```

Na segunda imagem podemos ver a parte dois, que pesquisa por um código de barras – e encontra.

In the second picture we can see the second part, that searches according to a bar code – and finds it.



```
Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda Main.java - IT16 - Visual Studio Code
PROBLEMAS 4 SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL Java Debug Console

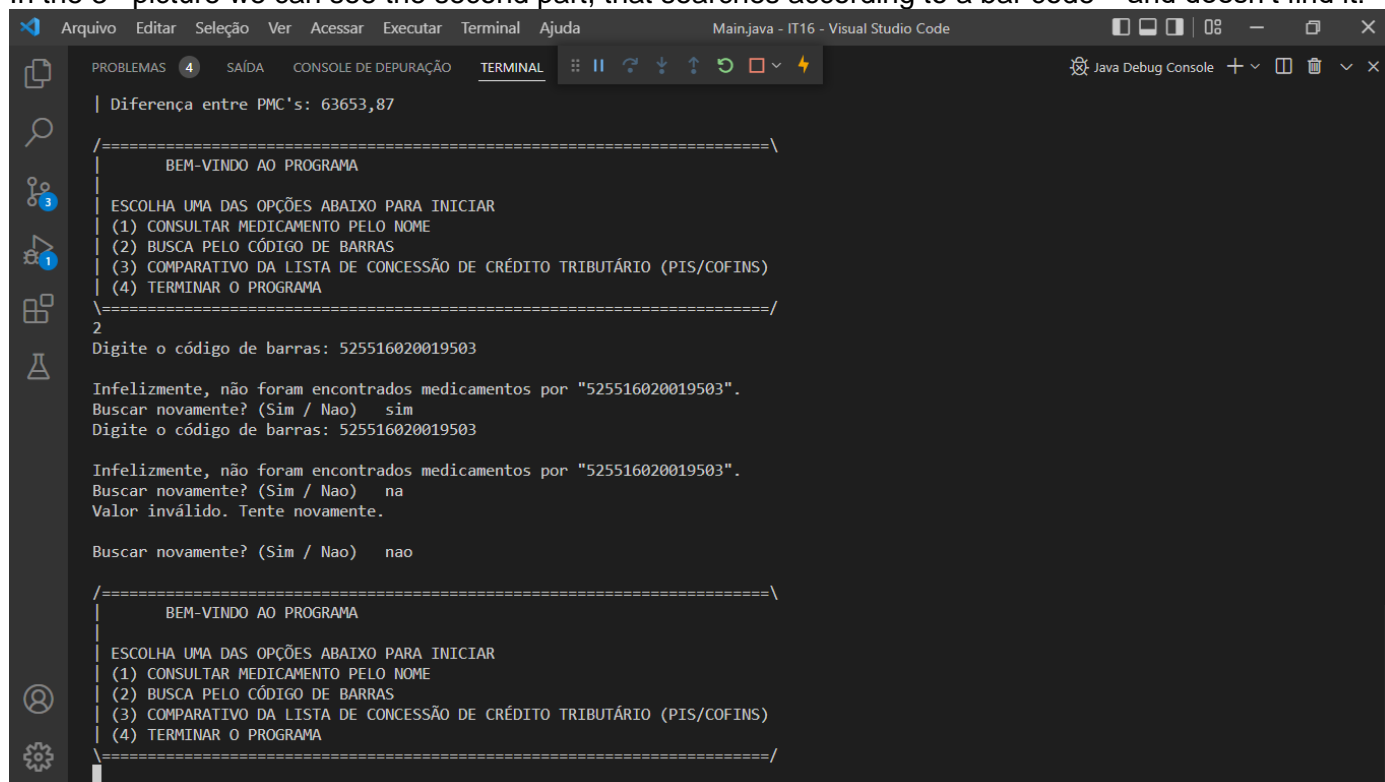
Dados do 8º medicamento encontrado por "vacina"
| Nome: VACINA CONTRA RAIVA
| Produto: VACINA RAIVA(INATIVADA)
| Apresentação: 2,5 UI PO LIOF SOL INJ CT 10 FR VD TRANS + 10 SER DIL X 0,5 ML C/ AGU ACOPLADA
| Valor PF Sem Impostos: R$ 815,26

/=====
| BEM-VINDO AO PROGRAMA
|
| ESCOLHA UMA DAS OPÇÕES ABAIXO PARA INICIAR
| (1) CONSULTAR MEDICAMENTO PELO NOME
| (2) BUSCA PELO CÓDIGO DE BARRAS
| (3) COMPARATIVO DA LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)
| (4) TERMINAR O PROGRAMA
|=====
2
Digite o código de barras: 7896210000000

Dados dos medicamentos encontrado por "7896210000000"
| PMC mais alto: 63653,87
| PMC mais baixo: 0,00
| Diferença entre PMC's: 63653,87

/=====
| BEM-VINDO AO PROGRAMA
|
| ESCOLHA UMA DAS OPÇÕES ABAIXO PARA INICIAR
| (1) CONSULTAR MEDICAMENTO PELO NOME
| (2) BUSCA PELO CÓDIGO DE BARRAS
| (3) COMPARATIVO DA LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)
| (4) TERMINAR O PROGRAMA
|=====
```

Na 3ª imagem podemos ver a parte dois, que pesquisa por um código de barras, mas não encontra.
In the 3rd picture we can see the second part, that searches according to a bar code – and doesn't find it.



```
Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda Main.java - IT16 - Visual Studio Code
PROBLEMAS 4 SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL Java Debug Console

| Diferença entre PMC's: 63653,87

/=====\
|      BEM-VINDO AO PROGRAMA
|
| ESCOLHA UMA DAS OPÇÕES ABAIXO PARA INICIAR
| (1) CONSULTAR MEDICAMENTO PELO NOME
| (2) BUSCA PELO CÓDIGO DE BARRAS
| (3) COMPARATIVO DA LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)
| (4) TERMINAR O PROGRAMA
|=====|
2
Digite o código de barras: 525516020019503

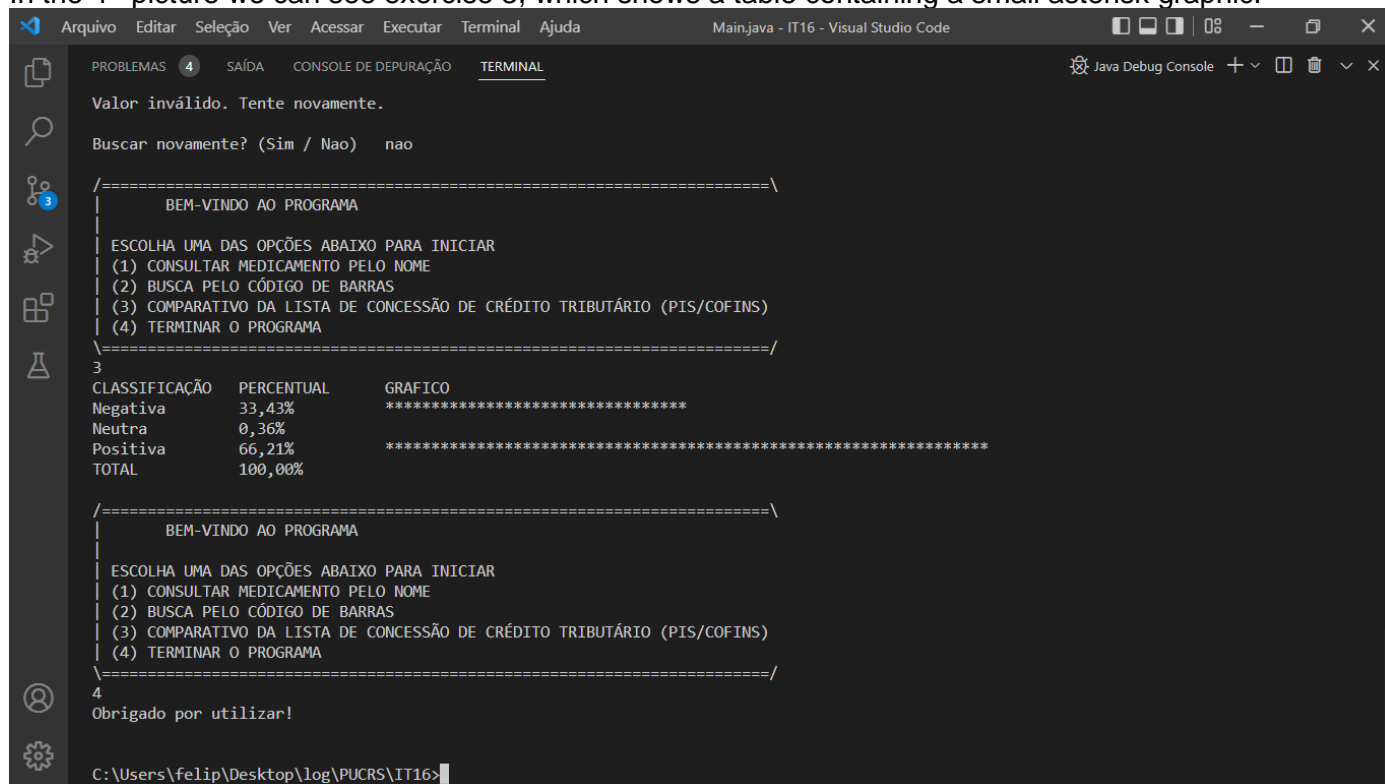
Infelizmente, não foram encontrados medicamentos por "525516020019503".
Buscar novamente? (Sim / Nao)  sim
Digite o código de barras: 525516020019503

Infelizmente, não foram encontrados medicamentos por "525516020019503".
Buscar novamente? (Sim / Nao)  na
Valor inválido. Tente novamente.

Buscar novamente? (Sim / Nao)  nao

/=====\
|      BEM-VINDO AO PROGRAMA
|
| ESCOLHA UMA DAS OPÇÕES ABAIXO PARA INICIAR
| (1) CONSULTAR MEDICAMENTO PELO NOME
| (2) BUSCA PELO CÓDIGO DE BARRAS
| (3) COMPARATIVO DA LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)
| (4) TERMINAR O PROGRAMA
|=====|
```

Na 4ª imagem podemos ver o exercício 3, que mostra uma tabela com um pequeno gráfico.
In the 4th picture we can see exercise 3, which shows a table containing a small asterisk graphic.



```
Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda Main.java - IT16 - Visual Studio Code
PROBLEMAS 4 SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL Java Debug Console

Valor inválido. Tente novamente.

Buscar novamente? (Sim / Nao)  nao

/=====\
|      BEM-VINDO AO PROGRAMA
|
| ESCOLHA UMA DAS OPÇÕES ABAIXO PARA INICIAR
| (1) CONSULTAR MEDICAMENTO PELO NOME
| (2) BUSCA PELO CÓDIGO DE BARRAS
| (3) COMPARATIVO DA LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)
| (4) TERMINAR O PROGRAMA
|=====|
3
CLASSIFICAÇÃO  PERCENTUAL  GRAFICO
Negativa      33,43%      *****
Neutra        0,36%
Positiva      66,21%      *****
TOTAL         100,00%

/=====\
|      BEM-VINDO AO PROGRAMA
|
| ESCOLHA UMA DAS OPÇÕES ABAIXO PARA INICIAR
| (1) CONSULTAR MEDICAMENTO PELO NOME
| (2) BUSCA PELO CÓDIGO DE BARRAS
| (3) COMPARATIVO DA LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)
| (4) TERMINAR O PROGRAMA
|=====|
4
Obrigado por utilizar!

C:\Users\felip\Desktop\log\PUCRS\IT16>
```

Na 5ª imagem começamos a parte dos testes, pela impressão do cabeçalho da tabela.
In the 5th picture we start the testing, by printing all the columns of the header.

```

Arquivo  Editar  Seleção  Ver  Acessar  Executar  Terminal  Ajuda  ReadCSV.java - IT16 - Visual Studio Code

PROBLEMAS 4  SAÍDA  CONSOLE DE DEPUÇÃO  TERMINAL  Java Debug Console

C:\Users\felip\Desktop\log\PUCRS\IT16> c: && cd c:\Users\felip\Desktop\log\PUCRS\IT16 && cmd /C ""C:\Program Files\OpenJDK\openjdk-8u302-b08\bin\java.exe" -agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:63861 -cp C:\Users\felip\AppData\Roaming\Code\User\workspaceStorage\23ffb62fca03403b601006dd91f95b31\redhat.java\jdt_ws\IT16_e72b3398\bin etapa2.testes.ReadCSV ""

Line 1: SUBSTÂNCIA;CNPJ;LABORATÓRIO;CÓDIGO GGREM;REGISTRO;EAN 1;EAN 2;EAN 3;PRODUTO;APRESENTAÇÃO;CLASSE TERAPÊUTICA;TIPO DE PRODUTO (STATUS DO PRODUTO);REGIME DE PREÇO;PF Sem Impostos;PF 0%;PF 12%;PF 17%;PF 17% ALC;PF 17,5%;PF 17,5% ALC;PF 18%;PF 18% ALC;PF 20%;PMC 0%;PMC 12%;PMC 17%;PMC 17% ALC;PMC 17,5%;PMC 17,5% ALC;PMC 18%;PMC 18% ALC;PMC 20%;RESTRIÇÃO HOSPITALAR;CAP;CONFAZ 87;ICMS 0%;ANÁLISE RECURSAL;LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS);COMERCIALIZAÇÃO 2020;TARJA
Column 0: SUBSTÂNCIA
Column 1: CNPJ
Column 2: LABORATÓRIO
Column 3: CÓDIGO GGREM
Column 4: REGISTRO
Column 5: EAN 1
Column 6: EAN 2
Column 7: EAN 3
Column 8: PRODUTO
Column 9: APRESENTAÇÃO
Column 10: CLASSE TERAPÊUTICA
Column 11: TIPO DE PRODUTO (STATUS DO PRODUTO)
Column 12: REGIME DE PREÇO
Column 13: PF Sem Impostos
Column 14: PF 0%
Column 15: PF 12%
Column 16: PF 17%
Column 17: PF 17% ALC
Column 18: PF 17,5%
Column 19: PF 17,5% ALC
Column 20: PF 18%
Column 21: PF 18% ALC
Column 22: PF 20%
Column 23: PMC 0%
Column 24: PMC 12%
Column 25: PMC 17%

```

Na 6ª imagem temos o final do cabeçalho e o começo da primeira linha com medicamentos de fato.
In the 6th picture we have the rest of the header and the beginning of the first line with medications.

```

Arquivo  Editar  Seleção  Ver  Acessar  Executar  Terminal  Ajuda  ReadCSV.java - IT16 - Visual Studio Code

PROBLEMAS 4  SAÍDA  CONSOLE DE DEPUÇÃO  TERMINAL  Java Debug Console

Column 25: PMC 17%
Column 26: PMC 17% ALC
Column 27: PMC 17,5%
Column 28: PMC 17,5% ALC
Column 29: PMC 18%
Column 30: PMC 18% ALC
Column 31: PMC 20%
Column 32: RESTRIÇÃO HOSPITALAR
Column 33: CAP
Column 34: CONFAZ 87
Column 35: ICMS 0%
Column 36: ANÁLISE RECURSAL
Column 37: LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)
Column 38: COMERCIALIZAÇÃO 2020
Column 39: TARJA

Line 2: SALICILATO DE FENILA,ÁCIDO SALICÍLICO,ÓXIDO DE ZINCO,ENXOFRE,MENTOL;33.379.884/0001-96;LABORATORIO SIMOES LTDA.;'520501000000000;'57600510011;'7896210000000;' - ;' - ;TALCO ALÍVIO;TALQUEIRA C/ 100 G;D10A - ANTIACNEICOS TÓPICOS;Similar;Regulado;5,04;5,65;6,53;6,98;6,08;7,03;6,11;7,08;6,15;7,29;7,58;8,72;9,3;8,41;9,37;8,45;9,43;8,5;9,7;Não;Não;Não;Não;0;Negativa;Não;Tarja - (*)
Column 0: SALICILATO DE FENILA,ÁCIDO SALICÍLICO,ÓXIDO DE ZINCO,ENXOFRE,MENTOL
Column 1: 33.379.884/0001-96
Column 2: LABORATORIO SIMOES LTDA.
Column 3: 520501000000000
Column 4: 57600510011
Column 5: 7896210000000
Column 6: -
Column 7: -
Column 8: TALCO ALÍVIO
Column 9: TALQUEIRA C/ 100 G
Column 10: D10A - ANTIACNEICOS TÓPICOS
Column 11: Similar
Column 12: Regulado
Column 13: 5,04
Column 14: 5,65

```

Na 7ª imagem temos o final das colunas contendo informações sobre um medicamento.
 In the 7th picture we have the end of the columns containing information about a medication.

```

Column 8: TALCO ALÍVIO
Column 9: TALQUEIRA C/ 100 G
Column 10: D10A - ANTIACNEICOS TÓPICOS
Column 11: Similar
Column 12: Regulado
Column 13: 5,04
Column 14: 5,65
Column 15: 6,53
Column 16: 6,98
Column 17: 6,08
Column 18: 7,03
Column 19: 6,11
Column 20: 7,08
Column 21: 6,15
Column 22: 7,29
Column 23: 7,58
Column 24: 8,72
Column 25: 9,3
Column 26: 8,41
Column 27: 9,37
Column 28: 8,45
Column 29: 9,43
Column 30: 8,5
Column 31: 9,7
Column 32: Não
Column 33: Não
Column 34: Não
Column 35: Não
Column 36: 0
Column 37: Negativa
Column 38: Não
Column 39: Tarja -(*)
  
```

Na 8ª linha temos um dos momentos de debug – no caso – para observar a conversão do código de barras de String para número.
 In the 8th line we are debugging – in this case – to observe a bar code being converted from String to number.

```

// converting the String containing the [PMC 0%] value to a number
// the replace function is there because in the csv file numbers
// are represented
// by a comma (,), but in Java by a period (.)
double pmc = Double.parseDouble(fileLineContent[23].replace
(target: ",", replacement: "."));
pmcAmount.add(pmc);
foundMedicine = true;
  
```

```

st:63927 -cp C:\Users\felip\AppData\Roaming\Code\User\workspaceStorage\23ffb62fca03403b601006dd91f95b31\redhat.j
ava\jdt_ws\IT16_e72b3398\bin etapa2.main.Main
/=====
|
| BEM-VINDO AO PROGRAMA
|
| ESCOLHA UMA DAS OPÇÕES ABAIXO PARA INICIAR
| (1) CONSULTAR MEDICAMENTO PELO NOME
| (2) BUSCA PELO CÓDIGO DE BARRAS
| (3) COMPARATIVO DA LISTA DE CONCESSÃO DE CRÉDITO TRIBUTÁRIO (PIS/COFINS)
| (4) TERMINAR O PROGRAMA
|
|=====
2
Digite o código de barras: 7897337712965
  
```

Autoavaliação

Apesar de estar muito feliz com o resultado da minha aplicação, foi certamente um dos desafios mais complicados que fiz nos últimos tempos. Me vi forçado a estudar melhor como funcionava um arquivo .csv, como atualizar uma planilha em Excel com uma quantidade muito além de uma quantidade legível de colunas e linhas e como identificar pontos de problema externos – como o idioma do Windows, que impactou em muito no começo do meu desafio, visto que até eu entender que era este o problema tive que testar muitas e muitas coisas para corrigir dentro do código. No final, ainda tive que fazer algumas alterações no arquivo .csv, sendo estas; trocar todos os ponto e vírgula (;) da primeira coluna por vírgulas (,) , para não atrapalhar a leitura pelo código e ocasionar números diferentes de colunas por linha; adicionar aspas simples (') antes das colunas 4-8 (que continham números) para transformá-los em textos, já que o Excel por padrão entendia como sendo números muito grandes e utilizava notação científica, o que ocasionava um problema no código, que não convertia de volta os números corretamente; por fim, adicionei 0's no lugar de células vazias (que deveriam conter números), para a conversão em número partir de 0 e não de vazio. Finalizada as partes da tabela, que ocupou muito mais do meu tempo do que esperava, pude começar o código, que foi relativamente simples por eu já ter participado do último desafio; eu já tinha uma ideia pronta de como fazer e acessar o menu, além de como partir dele para as classes auxiliares. Do mesmo modo, dentro de cada classe a leitura dos arquivos foi tranquila, pois utilizei da mesma técnica da última vez – desta, porem, fiz mais tentativas e consegui, acredito, fazer um código mais limpo e legível, além de possuir métodos úteis e outras ferramentas para lidar com dados inválidos e etc. No geral, foi um desafio tão estressante e trabalhoso quanto gratificante e recompensador.

Self-evaluation

Despite being very happy with the result of my application, it was certainly one of the most complicated challenges I've done in recent times. I found myself forced to study how a .csv file worked, how to update an Excel spreadsheet with an amount far beyond a legible number of columns and rows and how to identify external trouble spots - such as the Windows language, which had a huge impact at the beginning of my challenge, since until I understood that this was the problem I had to test many, many things to correct within the code. In the end, I still had to make some changes to the .csv file, these being; replace all semicolons (;) in the first column with commas (,) , so as not to disturb the code reading and cause different numbers of columns per line; add single quotes (') before columns 4-8 (which contained numbers) to turn them into text, as Excel by default understood them to be very large numbers and used scientific notation, which caused a problem in the code, which didn't convert numbers back correctly; finally, I added 0's in place of empty cells (which should contain numbers), for the conversion to be from a number (0) and not from a null cell. After finishing the parts of the table, which took up much more of my time than I had expected, I was able to start the code, which was relatively simple as I had already participated in the last challenge; I already had an idea of how to make and access the menu, as well as how to use it for calling the auxiliary classes. Likewise, within each class the reading of the files was relatively easy, as I used the same technique as the last time – this time, however, I made more attempts and managed, I believe, to make code cleaner and more readable, in addition to having useful methods and other tools to deal with invalid data, etc. Overall, it was a challenge that was as stressful and laborious as it was gratifying and rewarding.