Trabalho de a prog

Temos final ints para os valores:

* Recarga que diz no enunciado que cada recargas custa 5.5 euros
* Dia do carro suplemte que pode ser trocado lofo no início do codgio
* Temos um documento criado por nós com o input no enunciado pode ser trocado por um dos documentos que o professor queira

Temos throws **FileNotFoundException**: para ver se tem um ficheiro se não tiver dá um erro.

Casos para escolher se o utilizador quer dar input do ficheiro ou input do teclado.

# Funcoes:

## Fillarray:

* Objetivo criar uma funcao generica que tenho como input um array, funciona tanto para o txt como para dados introduzidos pelo utilizador.
* Para a quantidade de linhas e quantidade de colunas se número da posicao em que estamos na linha e coluna (matriz) e atribui um input a essa posicao.

## lerTeclado:

* vai ler input do utilizador.
* Temos uma condicao para garantir que nem as colunas nem as linhas sejam nulas porque senao não seria uma matrix.
* As colunas e as linhas sao introduzidas na funcao fillarray, e la que o utilizador dá o seu input mediante a quantidade de linhas e colunas que introduziu na funcao ler do teclado
* E um metodo que dá return ao array preenchido

## Read from file:

* Passa o nome do ficherio com os dados
* Variaveis para as colunas/linhas iguais a 0 para as consegires usar quando o programa le os valores vindos do ficherio, o array tambem está inicializado como null para ser utilizado depois

1. Passamos 4 variaveis vindas do ficherio:
   1. Sendo que a primeira e a descricao e e passada a linha toda para independenetemente do tamanho da descricao e sempre passada a linha toda para garantir que a descricao esteja completa.
   2. Passamos dois ints sendo que o primeiro sera o carro e o segundo sera os dias.
   3. Por fim e passado um array de número que tambem e criado com os valores lidos anteriormente dos dias e dos carros sendo que o array e lido mediante esses valores, que para quando esses valores, forem atingidos.

* Readfile.close: serve para fechar o scanner que le o docuemento, e para garantir que o sistema nao usa mais memoria daquilo que precisa, sao boas práticas.

## Imprimir array:

* Temos duas funcoes que imprimimem arrays
* Sendo que uma imprime ints e outra imprime doubles.
* Imprime ints serve sobretudo para o exercio a e para o exercio c em que os valores imprimidos nao tem de ter virgulas
* Enquando o array double e usado para os exercios: temos row prefix para podermos usar a mesma funcao para diferentes exercicios, e formato expecifio para estarem formatadas da forma como esta o enucido do trabalho neste caso com uma casa decimal. Como doubles e um dos exercios como percentagens

## Funcao recargas:

* Avaliar a quantidade de recargas em cada dia e em cada carro, sendo que se sobrar bateria passa para o dia seguinte como um resto.
* Decidimos fazer que enqaundo o valor na possicao i j desse maior que 100 seria sobtraido esse valor e somado mais 1 a um contador que depois sera impresso num novo array.
* Existe a necessidade de reniciar o contador sempre que passa para o carro seguinte
* O resto e igualado a posicao onde esta e somado para ser considerado do dia seguinte

## Kms diarios:

* Ver a media dos kms feitos num dos dias pelo carro
* Reiniciação da variavel media quando se passa para o carro seguinte
* A media e calculada com a soma de todos os valores de uma linha
* Criado um array com uma linha e soma de todos os valores da mesma linha de um carro

## ValorTotalRecargas:

* Utilizamos a funcao recargas para ser mais intuitivo de avaliar a carga das baterias
* Criado um total com a quantidade de recargas que estao na matriz, sendo que se o valor da posicao i j tiver 1 ou mais recargas sao somadas para depois serem multiplicadas pela variavel global valor recarga.

## Carga baterias:

Variavel inicializadas: carga- nova matriz com carros e dias introduzidos pelo utilizador ou txt, armazena a nivel de carga. Carga inicial, pois, os veiculos comecam com 100% no primeiro dia, bateriaMomento e o nivel da bateria atual resto: quilometros que o carro pode fazer no final do dia.

**Logica:**

* correr o array em todas as suas posicoes.
* Igualar bateria inicial a bateria do momento pois sempre que muda de carro e para garantir que no dia 1 comecam com 100% de bateria caso o consumo seja maior que a bateria disponivel. Atribuir o resto ao valor da posicao i j menos a bateria no momento
* Enquando este resto for maior que a bateria inicial podemos subtrair o resto a bateria inicial podes o carro vai necessiatar de carregar, bateriamomento= cargainicial- resto
* subtrai restante da última bateria condicao para verificar se o cosumo e menor ou igual a bateria disponivel.
* Garantir que a bateria nunca fique negativa. quando a bateria for 0 e necssario fazer uma carga total para que o carro fique a 100% por isso atribuimos a carga na posicao i j a bateria no momento

## MostrarVeiculosAcimaMedia:

Variáveis inicializadas: array criado para armazenar a media dos kmdiarios com a função anteriormente criada anteriormente (kmDiarios). CarrosAcimaMedia, variável boolean que serve como um interruptor para fazer verificações. Variáveis counterDias e counterCarros inicializadas a “0” com o intuito de contar os dias acima da média e a quantidade de carros acima da média.

**Lógica:**

* Primeiro loop, sempre que mudamos para um carro (linha) diferente o contador de dias dá reset. Dentro de cada diase o valor de km dentro do array principal (input pelo utilizador ou txt) for maior do que o valor no array da média para esse mesmo dia, aumento o counterDias(counterDias++).
* Se todos os dias forem acima da média daquele dia, o interruptor torna-se “true” naquele dia.
* Verifica em cada carro, todos os dias estão acima da média, se todos os dias estiverem acima da média, o contador de carros aumento +1 valor (counterCarros++).

**Formatação**

* Fazemos um print com a quantidade de carros com todos os dias acima da média.
* Fazemos um loop para dar print de cada carro a que a condição de cima se verifique verdade.

## DiasConsecutivosRecarga

Variáveis inicializadas: array 2d listaRecargas utilizando a função anterior (recargas) para armazenar a lista das recargas. Array 1d listaRcargasConsecutivas com tamanho igual ao número de linhas do array principal.

**Lógica:**

* Inicialização do loop básico para iterar entre cada valor do array listaRecargas, sempre que mudamos de linha, a variável recargasConsecutivas é inicializada a “0”.
* Se em cada coluna de cada linha tiver um valor acima de 0, a variável recargasConsecutivas aumenta um valor(recargasConsecutivas++), caso o valor não seja maior do que 0(neste caso, seja simplesmente “0”), a listaRecargasConsecutivas do respetivo carro dá reset e numa próxima iteração, caso a quantidade de recargas consecutivas seja maior do que o valor guardado no array para esse carro, atualiza o valor de acordo com o maior valor (Math.max(a, b)).
* Se o último valor la listaRecargasConsecutivas for maior que 0, como o “else” nunca seria verificado, o valor nunca seria atualizado, logo foi necessário implmentar a mesma lógica fora do condicionamento “if else” para que o valor seja atualizado em qualquer situação.
* Criamos uma variável maxRecargas para fazer a comparação entre todos os valores do array listaRecargasConsecutivas.
* Comparamos o primeiro valor do array com todos os outros valores e guardamos o valor mais alto de modo a dar print desse mesmo valor mais tarde

**Formatação**

* Print para mostrar a maior quantidade de dias consecutivos
* Caso exista mais do que um carro com o valor máximo de dias consecutivos, criamos um loop para dar print dessa mesma quantidade de carros

## recarregarDiaMaisTardio

Variáveis inicializadas: Array 2D listaRecargas, mais uma vez a utilizar a função criada anteriormente(recargas) para armazenar a quantidade de recargas para cada carro em cada dia. CounterCarros para contar a quantidade de carros que precisam de recarregar, guardarDia, inicializado a “-1” conforme os parâmetros estipulados para o exercicio.

**Lógica:**

* Loop básico para iterar por cada carro em cada dia, mudando de carro, o counterCarros dá reset.
* Enquanto estamos a iterar pelos dias de cada carro, se o determinado carro precisar de recarga, o contador de carros aumento um valor(counterCarros++)
* No final do loop, verificamos se todos os carros necessitam de recarga (“if (counterCarros == linhas”) e guardamos o dia em que isso acontece na variável “guardarDia”, como o loop continuará a iterar, automaticamente vai buscar o dia mais tardio.

## ObterVeiculoSuplente

Variáveis inicializadas: veiculoSuplente para guardar o ID do veículo que vai substituir o veículo no determinado dia, listaCargas, array 2D que utiliza a função anteriormente feita(cargaBaterias) que vamos utilizar para seguir os critérios do veículo ser suplente.

**Lógica:**

* Loop básico para iterar em cada carro do dia escolhido préviamente.
* Dentro desse mesmo loop fazemos a verificação dos critérios estipulados no exercicio por ordem de importância.

1ª Condição - Se o carro veiculoSuplente tiver mais quilometragem do que o carro “i” no dia escolhido, o carroSuplente muda para o carro “i” (veiculoSuplente = i),

2ª Condição - Se o valor de quilometragem dos dois veículos for igual, o veículo é escolhido através do veículo com mais bateria restante.

3ª Condição - Se a quilometragem for a mesma e a quantidade de bateria for a mesma, é escolhido o carro suplente através do carro com ID mais baixo.