UDESC - Fundação Universidade do Estado de Santa Catarina Monitor: Vinícius Takeo
Professor: Fabiano Baldo vtkwki@gmail.com
Disciplina: POO 2020.2 github.com/takeofriedrich

A lista deve ser entregue até o dia **07/03/2021**, às 23h59, no Moodle, os arquivos devem ser compactados em um arquivo *.zip* ou *.tar*. O arquivo compactado deverá conter o projeto Eclipse ou Netbeans da lista. **Não serão aceitos projetos com os códigos-fonte no formato *.class*!**

Lista 5: Interface e Exceções

Exercício 1

Construa uma aplicação de console em Java que se comunique em três línguas diferentes. Essa aplicação deve exibir uma mensagem de boas-vindas, depois deve requisitar dois valores inteiros e exibir a soma, a multiplicação e o mínimo múltiplo comum (MMC) deles e, ao final, uma mensagem informando que a aplicação foi encerrada. O diagrama da Figura 1 descreve como a aplicação deve ser estruturada.

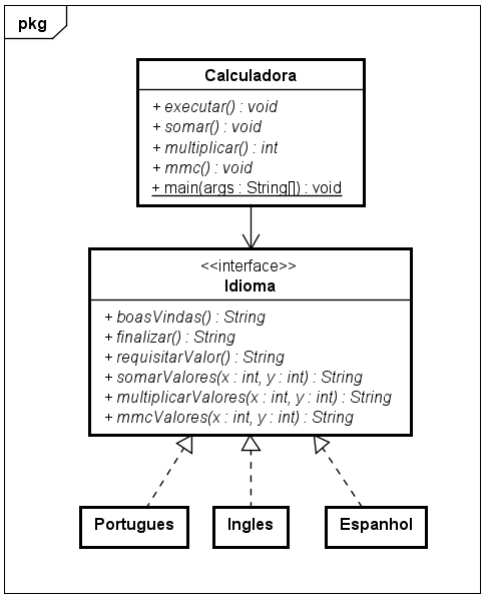

Figure 1: Aplicação de uma calculadora em Java.

As falas que descrevem o que cada uma das classes Portugues, Inglês e Espanhol devem exibir é apresentada na Tabela 1. A interface Idioma define o comportamento esperado das classes Portugues, Ingles e Espanhol. Cada uma dessas classes deve possuir os métodos: **boasVindas()**, que retorna uma String contendo uma mensagem cumprimentando o usuário no idioma em específico, **finalizar()**, que retorna a String contendo a mensagem de despedida ao usuário, **requisitarValor()**, que retorna uma mensagem requisitando um valor inteiro no idioma da classe, e também os métodos **somarValores()**, **multiplicarValores()** e **mmcValores()** que, dados dois valores inteiros, retornam a mensagem contendo o resultado do

cálculo. Por fim, a classe Calculadora possui um atributo idioma que pode representar uma instância de qualquer um dos idiomas (teste para cada uma das classes filhas) e os métodos: **somar()**, **multiplicar()** e **mmc()** que requisitam dois valores ao usuário (utilize a classe Scanner) e executam a operação de mesmo nome (utilizando dos métodos do idioma para exibir as mensagens), além de um método **executar()** que dá as boas-vindas ao usuário, executa os três métodos anteriores e após, se despede do usuário.

métodos	português	inglês	espanhol
boasVindas()	Bem-vindo!	Welcome!	Bienvenido!
finalizar()	Finalizando...	Exiting...	Saliendo...
requisitarValor()	Digite um inteiro:	Type an integer:	Ingresse un número inteiro
somaValores()	X mais Y é: Z	X plus Y is Z	X más Y es Z
multiplicarValores()	X vezes Y é Z	X times Y is Z	X por Y es Z
mmcValores()	O mínimo múltiplo comum entre X e Y é Z	The least common multiple between X and Y is Z	El mínimo común múltiplo entre X e Y es Z

Table 1: Tabela de falas da calculadora.

Exercício 2

A partir do diagrama na Figura 2, implemente os itens a seguir.

a) Implemente a classe Fila, que possui uma lista de objetos do tipo genérico T. O método **add()** sempre adiciona uma elemento no início da fila, lançando uma FilaCheiaException caso a fila esteja cheia. O método **tirar()** retira o elemento presente na última posição da lista, retornando-o. Caso a lista esteja vazia, uma FilaVaziaException deve ser lançada.

b) Crie um método **main()** contendo uma fila de 5 pessoas (utilize a classe Pessoa para o tipo genérico T) e uma interface de cadastro. Enquanto o usuário não digitar -1, a interface deve perguntar se o usuário deseja adicionar novas pessoas ou remover novas pessoas. A cada pessoa inserida ou removida, a fila deve ser exibida ao usuário. Trate as exceções adequadamente para que o programa não seja encerrado.

Exercício 3

Crie em Java, as classes descritas no diagrama da Figura 3. Para a manipulação de dados do tipo Date, utilize classe java.util.Date, leia a respeito no link.

a) A ContaBancaria possui um saldo e dois métodos: **sacar()** e **pagarBoleto()**. O método **sacar()** deve retirar da conta o valor, apenas se esse não exceder o limite. Caso o valor presente na conta seja insuficiente, uma exceção deve ser lançada, encapsulando a data em que o boleto venceu. O método **pagarBoleto()** possui a mesma lógica do sacar, porém, possui uma condição a mais. Caso a data de vencimento do boleto seja anterior a data atual do sistema, uma exceção deve ser lançada, encapsulando a quantidade de dias de atraso.

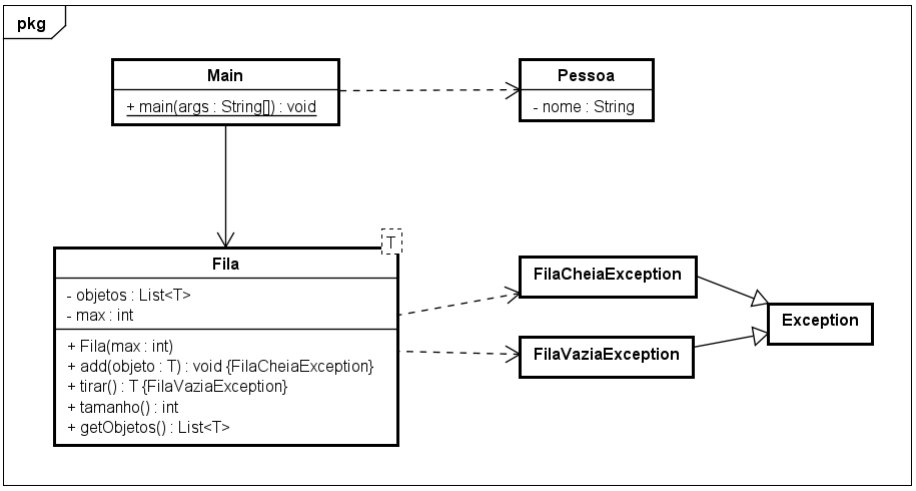

Figure 2: Aplicação de uma calculadora em Java.

b) Crie um método **main()** e instancie alguns casos de teste, tratando adequadamente as exceções.

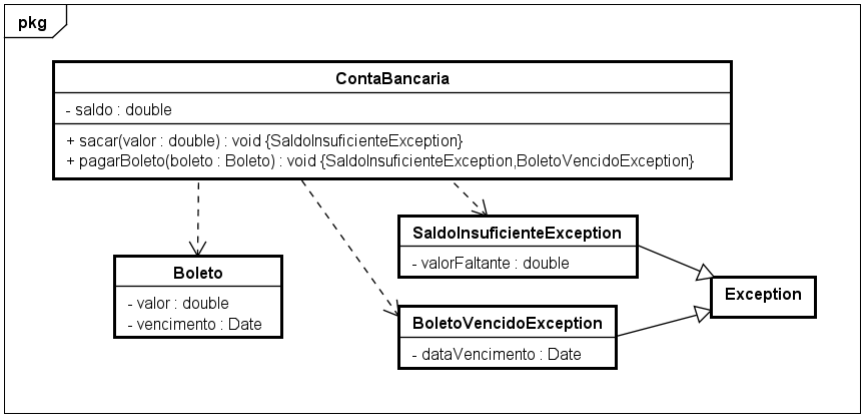

Figure 3: Aplicação de uma calculadora em Java.

Dicas:

- Para comparar objetos do tipo Date, utilize a o método **compareTo()**, como visto em aulas e listas anteriores, tal método, ao comparar 'a' e 'b', retornará 1 se 'a' for maior que 'b', 0 se forem iguais e -1 caso 'a' seja menor que 'b'.
- A classe Date possui vários construtores, na documentação é possível ver todos os principais métodos, entretanto, dois se destacam:

- **Date()**: constrói um objeto contendo a hora atual do sistema.
- **Date(int ano, int mes, int dia, int horas, int minutos, int segundos)**: constrói um objeto com os parâmetros passados.

Fontes:

- https://docs.oracle.com/javase/8/docs/api/java/util/Date.html;
- https://www.geeksforgeeks.org/date-compareto-method-in-java-with-examples/;

Questão Extra

Crie um algoritmo em Python que seja capaz de cadastrar pessoas até que o usuário digite -1. O programa deve requisitar o nome, idade, cpf e email da pessoa e exibir a lista de cadastrados a cada vez que uma nova pessoa for inserida. Crie e lance exceções para garantir que:

- A idade deve ser menor que 120 anos;
- O cpf possua 11 digitos e seja válido; (pesquise alguma implementação de validação em Python, ou implemente de acordo com o algoritmo abaixo).
- Email possua o caracter @ (caso deseje, utilize expressões regulares para fazer a verificação do email).

A classe Pessoa deve ser criada, e os métodos de setar idade, cpf e email devem lançar as exceções especializadas, caso os valores informados sejam incorretos.

Algoritmo para determinar se um CPF é valido (caso deseje implementar você mesmo)

O CPF é composto por onze algarismos, onde os dois últimos são chamados de dígitos verificadores, ou seja, os dois últimos dígitos são criados a partir dos nove primeiros. O cálculo é feito em duas etapas utilizando a divisão inteira por 11. Para exemplificar melhor será usado um CPF hipotético, por exemplo, 222.333.444-XX.

O primeiro dígito é calculado com a distribuição dos dígitos colocando-se os valores 10,9,8,7,6,5,4,3,2 conforme a representação abaixo:

Na sequência multiplica-se os valores de cada coluna:

2	2	2	3	3	3	4	4	4
10	9	8	7	6	5	4	3	2
20	18	16	21	18	15	16	12	8

Em seguida efetua-se o somatório dos resultados (20+18+...+12+8), o resultado obtido (144) deve ser divido por 11. Considere como quociente apenas o valor inteiro, o resto da divisão será responsável pelo cálculo do primeiro dígito verificador. 144 dividido por 11 tem-se 13 de quociente e 1 de resto da divisão. Caso o resto da divisão seja menor que 2, o primeiro dígito verificador se torna 0 (zero), caso contrário, subtrai-se o valor obtido de 11. Como o resto é 1 então o primeiro dígito verificador é 0 (222.333.444-0X).

Para o cálculo do segundo dígito será usado o primeiro dígito verificador já calculado. Monta-se uma tabela semelhante a anterior só que desta vez é usada na segunda linha os valores 11,10,9,8,7,6,5,4,3,2, já que é incorporado mais um algarismo para esse cálculo.

2	2	2	3	3	3	4	4	4	0
10	9	8	7	6	5	4	3	2	2
22	20	18	24	21	18	20	16	12	0

Depois efetua-se o somatório dos resultados: 22+20+18+24+21+18+20+16+12+0=171.

Agora, pega-se esse valor e divide-se por 11. Considere novamente apenas o valor inteiro do quociente, e o resto da divisão, no caso 6, usa-se para o cálculo do segundo dígito verificador, assim como na primeira parte. Se o valor do resto da divisão for menor que 2, esse valor passa automaticamente a ser zero, caso contrário, é necessário subtrair o valor obtido de 11 para se obter o dígito verificador, nesse caso 11-6=5. Portanto, ao final dos cálculos é possível perceber que os dígitos verificadores do CPF hipotético são os números 0 e 5, portanto o CPF fica:

222.333.444-05