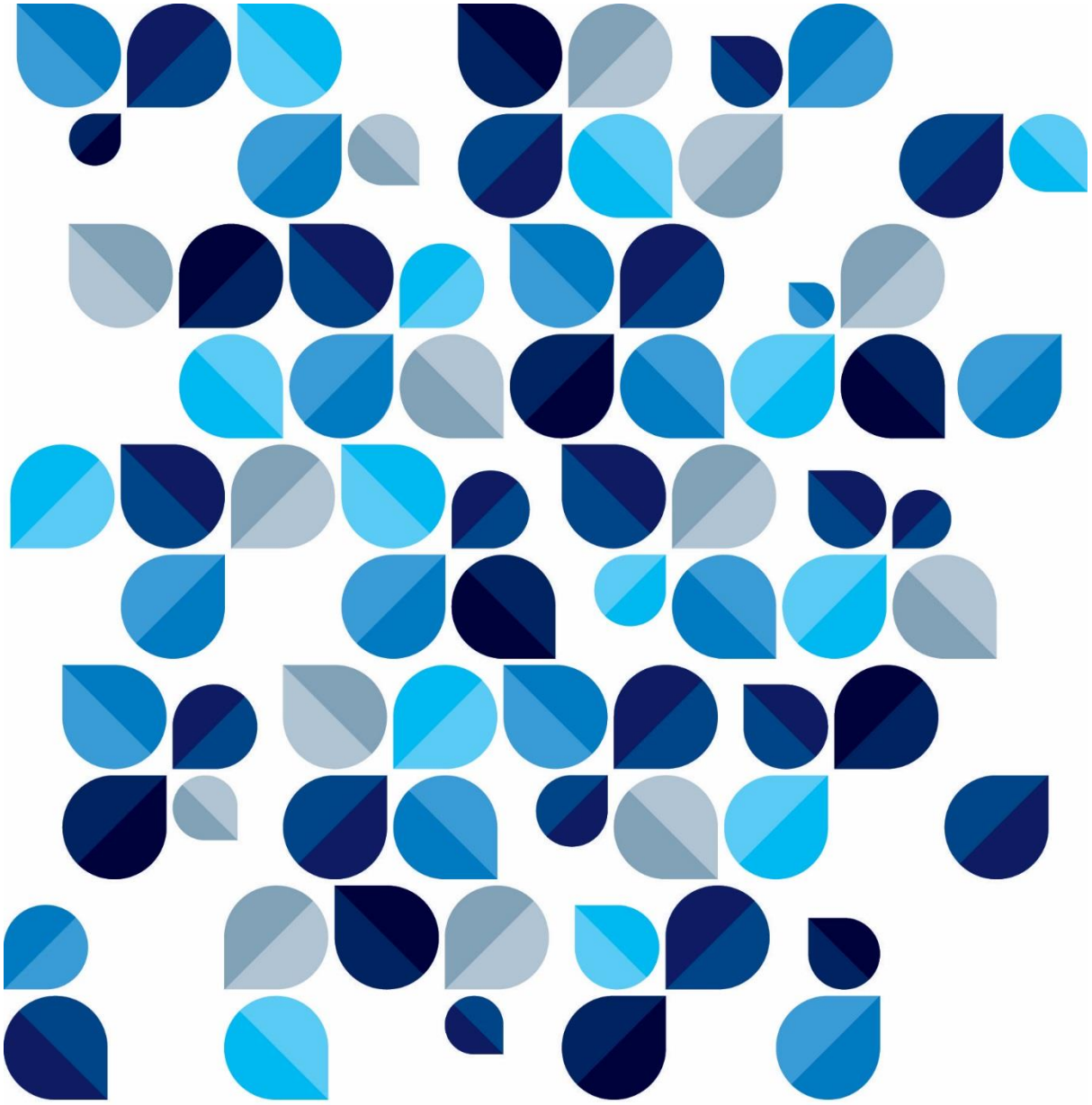

Software Developer

Avaliação Técnica UNIC

Essa é a terceira etapa do processo seletivo para desenvolvedor da UNIC, que é composto de dois exercícios de programação.

Nesse documento você vai encontrar:

- Critérios da avaliação: que descreve o que será levado em consideração na avaliação desse exercício.
- Exercício 1: exercício que simula uma alteração/melhoria num código já existente.
- Exercício 2: que simula uma entrada de dados de um sistema externo, com processamento com as regras de domínio interna e saída de dados.
- Envio: instruções para nos disponibilizar o exercício.

Critérios da avaliação

Nessa avaliação serão propostos dois exercícios relativamente simples de programação, onde você poderá aproveitar para demonstrar seu conhecimento em desenvolvimento de software.

Valorizamos o desenvolvimento de um código de boa qualidade, que seja simples, fácil de ler, entender e alterar. Por isso, avaliaremos a simplicidade, clareza e organização do código, ou seja, o quão "limpo" ele está. O design orientado a objetos também é uma característica importante que verificaremos. Buscamos aplicar em nosso dia bons princípios como: DRY, YAGNI e SOLID. E padrões como: DDD e Clean Architecture.

Acreditamos que uma eficaz cobertura de testes automatizados é imprescindível para garantirmos a qualidade e produtividade na manutenção e evolução do software. Por isso, também avaliaremos os testes. Pedimos que realize os exercícios na linguagem Java, podendo utilizar qualquer biblioteca e framework que desejar, inclusive na parte dos testes. Inclua um arquivo chamado README.md com instruções ou decisões tomadas.

Não é esperado e não será analisado códigos de: persistência em banco de dados ou arquivo, front-end ou iteração com console. Por outro lado, separação de classes e pacotes é algo esperado e que será considerado na implementação.

O resultado dos exercícios deve ser enviado conforme as instruções no final deste documento.

Exercício 1

Você está recebendo uma classe Java chamada **GeradorObservacao**. Essa classe é uma classe legada similar ao que temos em alguns pontos do nosso produto que poderia fazer parte de uma geração de relatório para o nosso sistema, e onde precisamos fazer uma alteração por uma necessidade levantada.

A classe atual recebe uma lista de inteiros, e devolve uma String com esses inteiros separados por vírgula concatenados a um texto inicial. Ajustando o texto para singular ou plural de acordo com a quantidade de itens da lista.
Exemplo:
Dado de entrada: Arrays.asList(1, 2, 3, 4, 5)
Retorno esperado: "Fatura das notas fiscais de simples remessa: 1, 2, 3, 4 e 5."

Um dos clientes do nosso sistema que utiliza essa funcionalidade nos trouxe a necessidade de querer ver essas informações junto com o valor, no seguinte formato:
"Fatura das notas fiscais de simples remessa: 1 cujo valor é R$ 10,00, 2 cujo valor é R$ 35,00, 3 cujo valor é R$ 5,00, 4 cujo valor é R$ 1.500,00 e 5 cujo valor é R$ 0,30. Total = 1.550,30."

É uma prática comum nossa manter o comportamento já existente do sistema. Para isso, é necessário suportar que ambas as opções coexistam e estejam disponíveis em funcionamento (o código legado por ser alterado, mas o seu comportamento deve ser mantido com funcionamento igual ao atual).

Ao final da implementação, enviar as classes de produção e testes que atendem as necessidades acima.
Além das classes, detalhar no README.md os aspectos negativos do código atual e os motivos que incentivaram as possíveis mudanças no código.

Exercício 2

O Sienge é um ERP especializado para a gestão de empresas da indústria da construção, sendo que a maior parte dos clientes desse produto são construtoras e incorporadoras. Na realização de suas obras é muito comum os clientes precisarem calcular qual o custo total de uma Obra.

Para conseguir calcular o custo da obra, os engenheiros fazem uma separação das composições e insumos necessários para a realização de cada etapa da obra. Um insumo pode ser qualquer tipo de material (exemplo: cimento, cal, areia e outros) ou também qualquer tipo de mão de obra (pedreiro, engenheiro, servente e outros). Uma composição é um conjunto de insumos ou outras composições, como pode exemplo a construção de uma parede que para ser executada pode precisar da mão de obra de um pedreiro, cal, cimento e outros insumos.

Digamos que para fazer uma obra vamos precisar preparar 3m³ de argamassa, nesse caso poderíamos ter uma composição chamada: “Preparo de argamassa para assentamento”. E para cada 1m³ de argamassa precisamos dos seguintes insumos:

- 0,5 hora(s) de mão de obra de um pedreiro (que custa R$ 30,00 por hora);
- 1 hora(s) de mão de obra de um servente (que custa R$ 18,00 por hora);
- 0,1 sacos do material cimento (que custa R$ 30,00 por saco);
- 5kg do material areia (que custa R$ 20,00 por kg);
- 1kg do material cal (que custa R$ 9,00 por kg);

Nesse cenário acima, o custo para cada 1m³ de argamassa é a soma das multiplicações dos valores unitários por unidade de tempo acima:

0,5 * 30 + 1 * 18 + 0,1 * 30 + 5 * 20 + 1 * 9 = 145

Dessa forma, o custo para cada 1m³ de argamassa seria de R$ 145,00. Sendo que o custo por 3m³ de argamassa seria R$ 435,00.

Além disso, o valor de uma composição também pode ser composto de outras composições. Por exemplo, poderia ser segunda composição chamada “Construção de paredes” composta por outra composição chamada “Preparo de argamassa para assentamento”. E para cada 1m² de parede fosse necessário a seguinte composição:

- 0,5 hora(s) de mão de obra de um pedreiro (que custa R$ 30,00 por hora);
- 1 hora(s) de mão de obra de um servente (que custa R$ 18,00 por hora);
- 10 unidade de tijolo de 6 furos (que custa R$ 0,34 por unidade);
- 3m³ de “preparo de argamassa para assentamento” (que custa R$ 145,00 por m³);

Nesse cenário acima, o custo para cada 1m² de parede é a soma das multiplicações dos valores unitários por unidade de tempo acima:

0,5 * 30 + 1 * 18 + 10 * 0,34 + 3 * 145 = 471,4

Dessa forma, o custo para cada 1m² de parede seria de R$ 471,40. Sendo que o custo por 3m³ de argamassa seria R$ 1.305,00.

Nesse exercício você está recendo um JSON que contém composições e queremos que você desenvolva um modelo para o cálculo dos valores das composições do arquivo e retorne o valor unitário de cada uma das composições contidas no arquivo, conforme as instruções a seguir:

- Para essa implementação você está recebendo um JSON pronto como dado de entrada, esse arquivo pode estar num caminho fixo do projeto e ser lido por uma classe Main;
- Nesse JSON cada linha contém o serviço e sua composição, como um serviço pode mais de uma composição as informações do serviço estão repetidas dentro de cada JSON;
- Como auxiliar a implementação e ao entendimento, você pode usar o arquivo XLS que mostra detalhes de como é feito o cálculo dos serviços e suas composições;
- É esperado que a saída seja feita por uma classe Main, sendo impresso via console os valores dos 5 serviços, conforme o modelo a seguir:

94793 REGISTRO DE GAVETA BRUTO, LATÃO, ROSCÁVEL, 1 1/4, COM ACABAMENTO E CANOPLA CROMADOS, INSTALADO EM RESERVAÇÃO DE ÁGUA DE EDIFICAÇÃO QUE POSSUA RESERVATÓRIO DE FIBRA/FIBROCIMENTO FORNECIMENTO E INSTALAÇÃO. AF_06/2016 UN 128,60
98561 IMPERMEABILIZAÇÃO DE PAREDES COM ARGAMASSA DE CIMENTO E AREIA, COM ADITIVO IMPERMEABILIZANTE, E = 2CM. AF_06/2018 M2 28,73
87286 ARGAMASSA TRAÇO 1:1:6 (CIMENTO, CAL E AREIA MÉDIA) PARA EMBOÇO/MASSA ÚNICA/ASSENTAMENTO DE ALVENARIA DE VEDAÇÃO, PREPARO MECÂNICO COM BETONEIRA 400 L. AF_06/2014 M3 289,97
88830 BETONEIRA CAPACIDADE NOMINAL DE 400 L, CAPACIDADE DE MISTURA 280 L, MOTOR ELÉTRICO TRIFÁSICO POTÊNCIA DE 2 CV, SEM CARREGADOR - CHP DIURNO. AF_10/2014 CHP 1,25
88831 BETONEIRA CAPACIDADE NOMINAL DE 400 L, CAPACIDADE DE MISTURA 280 L, MOTOR ELÉTRICO TRIFÁSICO POTÊNCIA DE 2 CV, SEM CARREGADOR - CHI DIURNO. AF_10/2014 CHI 0,22

Instruções para envio:

Para o envio dos exercícios pedimos que seja criado um repositório **privado** no github:

- Adicionar como contribuidor o usuário: devunic
- Fazer apenas 2 commits: um para cada exercício (sendo que o feedback será dado em cima do commit)
- Lembrar de colocar observações, decisões e outros comentários que possam ser úteis no arquivo README.