Avaliação

Nome: Tallya Jesus Sousa Barbosa

Até o momento foram disponibilizados 18 tópicos no portal da disciplina (https://github.com/kyriosdata/oo). Abaixo seguem questões para serem respondidas com base no que você disponibilizou no seu repositório do Github da disciplina.

Obs.: como no momento já foi disponibilizado até o tópico 20, estou considerando 20 nos pontos onde está escrito 18

RESPOSTA	Responda o que se pede no espaço reservado para resposta.
20	Quantos tópicos você leu? Se leu todos eles, então a resposta é 18. É possível que não tenha lido todos e, neste caso, forneça a quantidade daqueles lidos.
sim	Você esclareceu pelo menos 50% das suas dúvidas, pertinentes aos tópicos? Se afirmativo, a resposta é SIM, caso contrário, NÃO..
20	Temos 18 tópicos. Houve aprendizado em quantos deles? Se em cada tópico você aprendeu algo, então a resposta é 18.
sim	Seu repositório `poo-2023-01` contém o arquivo README.md que informa a finalidade deste repositório para os visitantes? Para obter informações há várias fontes, por exemplo, https://www.makeareadme.com/.
sim	Suas respostas no seu repositório poo-2023-01 estão em diretórios em conformidade com o que foi especificado, ou seja, seus diretórios tem como nomes t07, t08 e assim por diante. Observe que são apenas 3 caracteres e o t segue em minúscula. Responda SIM caso afirmativo ou NÃO, caso contrário.
9 (considerando apenas os tópicos com algum artefato)	Você respondeu quantos tópicos? (conte cada tópico respondido, ou respondido parcialmente, como uma unidade).Ou seja, no máximo a resposta é 18..
256	Quantas classes em Java você criou? Em sua cópia local, se estiver usando Windows, você pode usar o comando `dir /s /b *.java | find /c /v ""`. O resultado será a quantidade de arquivos .java disponíveis no diretório e subdiretórios onde o comando for executado. No Linux o comando é similar. Em caso de dúvida, tente o ChatGPT para te ajudar.
1707	Conte quantas linhas de código você criou. Naturalmente, você pode abrir cada arquivo e ir acumulando o total de linhas, mas alerto que há formas mais eficientes para se fazer isso, novamente, via linha de comandos.
sim	Seu código está organizado em diretórios conforme a convenção introduzida pela ferramenta Maven? Por exemplo, o código fonte está em um diretório contido no diretório java que, por sua vez, está em um diretório src? Contém o arquivo pom.xml que informa como compilar, por exemplo, o código produzido? Responda com um SIM se positivo para todas estas questões e NÃO caso contrário.
Foi empregado o repositório	Durante as aulas foi sugerido um repositório exemplo para sua orientação. Caso não tenha empregado ele, sabe diferençar o que você ganha e o que você perde quando não o emprega?
sim	Quando se faz uso de uma ferramenta de controle de versão como Git, não se persiste diretórios como target, bin e outros que são gerados por compiladores a partir do código fonte. Esta é uma orientação tão importante que o uso do Git, por exemplo, faz uso de um arquivo de nome especial `.gitignore` para conter, em dada linha, o que deve ser

	“ignorado” pelo Git em suas operações. Dado que cada linguagem tem suas próprias convenções e ferramentas, faz-se necessário um .gitignore para cada uma delas. Um projeto com mais de 150k estrelas no Github é https://github.com/github/gitignore. Ele contém sugestões para .gitignore para muitas linguagens. Responda SIM se compreende a finalidade do arquivo .gitignore ou NÃO, caso contrário.
sim	Suas respostas que envolvem código contém um arquivo .gitignore correspondente?
sim	Código em Java usa linhas em branco para demarcar a separação de um tópico de outro, seja dentro de um mesmo método ou entre métodos e também entre elementos de uma classe, por exemplo, package é separado dos imports por uma linha em branco e os imports, por sua vez, separados da declaração da classe, por uma linha em branco. Seu código usa linhas em branco para essa finalidade?
não	Há linhas em branco, por exemplo, duas ou mais em seu código, simplesmente deixadas lá simplesmente porque não foram removidas? Naturalmente, isto não é desejável.
sim	Você entendeu para que serve o arquivo pom.xml, a ferramenta Maven e porque o código em Java é organizado desta forma? Responda SIM caso tenha entendido ou NÃO, caso contrário.
todos	Ao baixar seu repositório poo-2023-01 e em cada diretório criado para responder o tópico em questão, quantos deles o comando `mvn compile` compila de forma satisfatória o código produzido?
sim	A convenção para nomear packages em Java é usar apenas letras minúsculas. Ou seja, diferente da nomeação de classes, por exemplo, que deve se iniciar por maiúsculas. Responda SIM caso tenha atendido esta convenção ou NÃO, caso contrário.
sim	As classes criadas começam com letra maiúscula e não possuem acento? Java segue a convenção conhecida por CamelCase, por exemplo, uma classe para classificar os guarda-chuvas seria normalmente nomeada por GuardaChuva. Os nomes das suas classes seguem esta convenção?
sim	Nomes de packages devem seguir a convenção de usar “domínios reversos”. Por exemplo, se estuda na UFG, cujo domínio é “ufg.br”, então o domínio reverso é “br.ufg”. O domínio do Instituto de Informática é “inf.ufg.br”. Ou seja, o reverso é br.ufg.inf. Responda SIM se usou esta convenção na nomeação dos packages ou NÃO, caso contrário.
sim	Após o nome da classe deve seguir um espaço, ou seja, a classe Saude, porque classes em Java não empregam acentos, deve ser declarada como `class Saude {` com espaço entre o abre chaves e a palavra Saude.
sim	Nomes de propriedades e variáveis em Java, por convenção, são iniciadas por letra minúscula. Responda SIM se seguiu esta convenção ou NÃO, caso contrário. Por exemplo, se seu código tem algo como `Casa C = new Casa()`, então não segue esta convenção.
sim	Quando se faz uso da orientação a objetos existe um princípio ou orientação conhecida por “programação para interface”. Nestes tópicos houve oportunidade de fazer uso desta orientação, por exemplo, ao usar a estrutura de dados “lista”. Em Java existe a interface List e várias classes que implementam esta interface, dentre elas, ArrayList, Stack, Vector e outras. Neste caso, a sugestão é fazer uso de uma referência para a interface, em vez da referência para classe. Desta forma, pode-se, se preciso, trocar a implementação, por exemplo, ArrayList por outra implementação, sem afetar o restante do código. Você pode consultar https://www.baeldung.com/cs/program-to-interface para mais informações. Você compreendeu esta orientação?
sim	Você fez uso da orientação ou princípio sugerido acima?
Sim, mas não foi usado nenhum padrão como o conventional commits	Quando se usa o git, ao realizar um “commit”, você pede para este software que considere todas as mudanças realizadas como uma unidade. Para indicar o porquê destas mudanças ou informação relevante que permita orientar você ou outros em instante futuro, as mensagens correspondentes devem ser definidas de forma criteriosa. Em geral, fará uso de orientações da sua empresa, ou orientações como https://www.conventionalcommits.org/en/v1.0.0/. Houve cuidado na definição das mensagens associadas aos commits ao longo da realização das tarefas deste tópico?
sim	O tópico 18 requisita a criação de um programa, você entendeu o que foi requisitado?
não	O tópico 18 requisita a criação de um programa, você implementou o programa?