Faculdade de ciências exatas e engenharia

Licenciatura de Engenharia Informática

Programação Avançada

Resolução das Fichas da TP1 e TP2

Discentes

Alexandre Torra, 2079217

Índice

Índice

Introdução

Na cadeira de Programação Avançada, foi me proposto este trabalho no âmbito de dar-me a conhecer e a melhorar os meus conhecimentos destas tecnologias utilizadas. Tais como o GitHub(colocar o resto).

Este trabalho constituía em duas fichas, cada uma relativa a uma TP diferente. Na primeira tínhamos como objetivo trabalhar com o GitHub e fazer múltiplos testes para verificarmos se tudo estava a ser usado, como utilizar o coverage, se acontecia algum problema nos múltiplos testes que fazíamos e como podemos resolver esses problemas que surgiram.

Testes Em Java

Neste primeiro exercício comecei primeiro por ler o enunciado para ver o que era nos proposto. Apos ler o contexto do problema fiquei a saber que estamos a trabalhar com um software que trata de reservar os lugares dos passageiros para os voos. Este software tem algumas especificações. Existe dois tipos de passageiros (regular e VIP) e dois tipos de voos (económicos e executivo). Os passageiros VIP podem ser adicionados em qualquer voo, mas os regulares só podem ser adicionados ao económico. Em caso de overbooking a empresa prevê que os da classe económica poderão ser removidos dos voos.

Nos primeiros 3 pontos do exercício consegui fazer sem me deparar com qualquer tipo de problemas. Estes pontos consistiam em criar um fork do repositório da conta da UMa para minha conta de GitHub. Fazer clone do próprio para poder trabalhar no IDE e como ia fazer alterações no projeto, tive de criar uma nova branch para poder trabalhar sem alterar nada no master.

Apos já ter o projeto pronto a trabalhar, comecei a fazer testes unitários para ver se tudo estava a funcionar. Realizei @Test e @RepeatedTest para fazer a verificação.

De seguida realizei um teste com coverage para verificar se todas funções estavam a ser usadas. Como pode verificar nem todas a funções estavam a ser usadas. Então seguindo o 5ª ponto criei uma função que verificasse se o nome do passageiro criado era o mesmo do que o do passageiro que estávamos a espera, para assim poder utilizar a função em falta (esta sendo a getName do Passanger). Depois uma vez que estava tudo a ser usado corretamente dei commit da nova versão, seguida de um push comentando o que tinha realizado. Lendo o ponto 7 reparei que estava a acontecer uma falha no sistema, que tinha passageiros que estava a ser adicionados mais que uma vez. Então tentei

procurar a função que isto acontecia, percebi a logica e encontrei o problema que tinha era um for que estava adicionar o mesmo passageiro múltiplas vezes. Apenas foi preciso comentar o for que o problema fico resolvido (Figura 1). Assim testando pela última vez o programa, para verificar se estava tudo ok e dei commit e push. E realizando a seguir o ultimo ponto fazendo um pull-request.

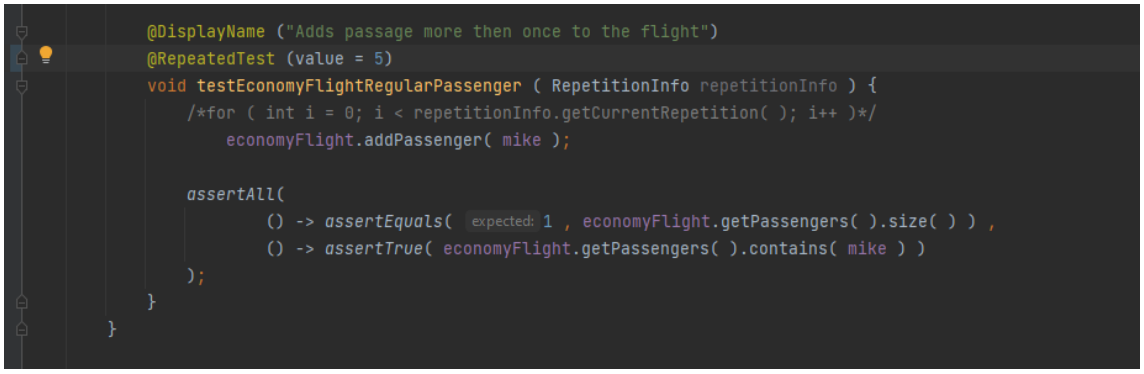

Figura 1- Resolução do problema

Após completar o primeiro exercício comecei a realizar o próximo. Começando primeiro por criar um ficheiro .gitignore, tive um pouco dificuldade a criar o ficheiro porque não tinha a certeza de como escrever, mas após procurar e com base no outros consegui criar o ficheiro corretamente. De seguida comecei por escrever o código inicialmente o exercício diz que a primeira tarefa consistia em que um cliente pode-se criar uma conta e movimentá-la (depositar e retirar dinheiro). O cliente possuía certos detalhes (tal como nome, idade, devida e número de filhos).

Com base nisto primeiramente criei a classe do cliente com os detalhes básicos que pediam. Só que agora faltava alguma maneira de poder ligar o cliente e a conta. Então eu criei uma class para a conta onde estava as ações das contas (como depositar e retirar dinheiro). Depois dentro da class cliente criei uma lista da class conta para poder assim conseguir associar cada cliente tenha a sua própria conta.

Ao longo deste processo de criação testei múltiplas vezes (com vários @Test e @RepeatedTest) para ver se estava a funcionar corretamente inicialmente estava-me a dar um erro na lista porque na função que eu criei para adicionar a conta ao cliente não estava a ser bem executada, porem depois de um tempo a testar consegui resolver o erro e por a funcionar. Na segunda tarefa deste exercício tínhamos de criar um sistema que pode se verificar se o cliente podia ou não pedir um empréstimo (isto, com base em múltiplos cálculos feitos para verificar se ele está apto ou não).

Primeiramente tentei fazer essas contas dentro da class do cliente, mas imediatamente depois apercebi me que estava mal. A seguir a, alguns teste pensei em criar uma class empréstimo onde depois com essa class chamava para determinar assim se o cliente era apto para o empréstimo ou não. E assim com esta resolução, com base em múltiplos testes consegui resolver o exercício. Com isto concluindo a primeira TP.

Agora começando na segunda TP comecei, como anteriormente, por ler o primeiro exercício e ver de que se tratava. Neste primeiro exercício nos tínhamos de reutilizar o primeiro exercício da outra TP e criar usando o Javadoc e IntelliJ IDEA a plataforma HTML da documentação do projeto. Após estudar os slides da aula, isto foi facilmente atingindo seguindo o método que estava lá (Figura 2).

Figura 2- Como gera o Documento HTML através do JavaDoc

Agora chegando ao último exercício, este exercício dividia se em duas partes(usando os dois o GitHub Actions), primeiro era testar o projeto em vários sistemas operativos nomeadamente o ubuntu, windows e macos. E outra parte era gerar ficheiros ZIP para a plataforma da documentação, para as métricas dos testes unitários executados e para a build do projeto. Deverei dizer que neste exercício foi o que senti mais dificuldade a realizar, porque não sabia bem o colocar nos ficheiros. yml para quando desse pull-request conseguir executar as tarefas. Depois rever algumas vezes os slides, fazer pesquisa na internet, ajuda dos professores e alguns testes falhados consegui finalmente por realizar as duas parte corretamente. Assim concluindo a segunda TP.

Conclusão

Assim acabei por concluir o que foi proposto no âmbito da cadeira de Programação Avançada. Deverei dizer que deparei-me com algumas dificuldades umas sendo estar um pouco esquecido como programar algumas partes em java mas o que foi resolvido após programar e testar durante os exercícios, mas a maior dificuldade deverei dizer que senti quando estava a criar e a fazer os ficheiros .yml como nunca tinha feito não sabia o muito bem o que era esperado e de como realizar, no inicio. Mas como foi dito anteriormente consegui superar este obstáculo e finalizar tudo o que foi proposto.