Code Review - Napp Academy

Semana 11

Amostra 1:

Na amostra foi atendido o requisito de implementação (Implementar o algoritmo de “texto_1” e de “texto_2” e suas respectivas estratégias para extrair os registros do arquivo) porém não foram criados os testes para essas implementações. Existem pontos de melhorias nas estratégias, na classe “Estrategia_Texto_1” do arquivo Estrategias.py poderia ter removido os comentários de códigos que não foram usados (linhas 84, 88, 89 e 93-97) em ambas estratégias poderia ter usado os contextos para abertura do arquivo, também poderia ter sido implementada a lógica de pular o cabeçalho no próprio for de gerar as linhas

Amostra 2:

Na amostra foi atendido o requisito de implementação (Implementar o algoritmo de “texto_1” e de “texto_2” e suas respectivas estratégias para extrair os registros do arquivo) os testes também foram implementados. Existem pontos de melhorias em ambas estratégias, não existe a necessidade de usar o método “txtfile.readlines()” já que na própria iteração de “txtfile” ele iria iterar entre as linhas, poderia ter sido usada outra lógica para gerar a “lista_valores” sendo feito apenas um único split na linha ao invés dos “replace” usados.

Amostra 3:

Na amostra foi atendido o requisito de implementação (Implementar o algoritmo de “texto_1” e de “texto_2” e suas respectivas estratégias para extrair os registros do arquivo) porém não foram criados os testes para essas implementações. Existem pontos de melhorias nas estratégias, nas duas estratégias poderia ser feita outra estratégia para pular as linhas inválidas, não usando tantos “ if ” poderia ser aproveitada a ideia do split line e verificado apenas o “ len ” desse “ split ” já que essas linhas inválidas teriam um tamanho diferente das válidas.

Code Review - Napp Academy

Semana 12

Amostra 1:

Na amostra foram feitas as implementações da classe abstrata “Produto” sendo “Dolly” e “GuaranaAntartica” e as implementações de “Caracteristica” sendo “Tamanho2litros” e “Tamanho3litros” foi ajustado o codigo de “produtos.py” com as novas características e produtos. A classe abstrata “Produtos” contém a implementação no construtor para gerar uma Exception caso o parametro “implementation” não seja subclasse de “Caracteristica” porém a linha de importação de “Caracteristicas” esta comentada no código, os testes para tal implementação não estão implementados.

Amostra 2:

Na amostra foram feitas as implementações da classe abstrata “Produto” sendo “Dolly” e “GuaranaAntartica” e as implementações de “Caracteristica” sendo “Tamanho2litros” e “Tamanho3litros” foi ajustado o codigo de “produtos.py” com as novas características e produtos. A classe abstrata “Produtos” contém a implementação no construtor para gerar uma Exception caso o parametro “implementation” não seja subclasse de “Caracteristica” os testes também foram implementados.

Amostra 3:

Na amostra foram feitas as implementações da classe abstrata “Produto” sendo “Dolly” e “GuaranaAntartica” e as implementações de “Caracteristica” sendo “Tamanho2litros” e “Tamanho3litros” foi ajustado o codigo de “produtos.py” com as novas características e produtos. A classe abstrata “Produtos” contém a implementação no construtor para gerar uma Exception caso o parametro “implementation” não seja subclasse de “Caracteristica” os testes também foram implementados.

Code Review - Napp Academy

Semana 13

Amostra 1:

Na amostra foram feitas as implementações de ERP1 e ERP2 para retornar somente o “total” e “vendido_em”, mas nesses não foram implementados os testes. Foi implementado também a classe “Relatorio_CSV” seguindo o padrão pedido e os testes.

Amostra 2:

Na amostra não foram feitas as implementações de ERP1 e ERP2 para retornar somente o “total” e “vendido_em”. Foi implementado também a classe “Relatorio_CSV” seguindo o padrão pedido e os testes. Ficaram códigos comentados no final do arquivo “Relatorios.py” que não estão sendo usados.

Amostra 3:

Na amostra não foram feitas as implementações de ERP1 e ERP2 para retornar somente o “total” e “vendido_em”. A classe “Relatorio_CSV” foi implementada com o nome de maneira incorreta “Relatorio_csv” devido a isso deveria ser feito o ajuste na linha 12 do arquivo “Abstaracao.py” para “Relatorio_csv” ou ajustar o nome da classe e suas importações para que seja atendida “Relatorio_CSV”

Code Review - Napp Academy

Semana 14

Amostra 1:

Na amostra foram implementados os testes para de redes sociais (redes_sociais.py) e sessões (sessoes.py). Também foram implementadas as redes sociais GitHub junto a sessão “UploadCode” também na rede social Instagram. Os testes poderiam estar separados por arquivos dos tipos redes sociais (redes_sociais.py) e sessões (sessoes.py).

Amostra 2:

Na amostra foram implementados os testes para de redes sociais (redes_sociais.py) e sessões (sessoes.py). Também foram implementadas as redes sociais GitHub junto a sessão “UploadCode” (UploadCodeSection) também na rede social Instagram. Os testes poderiam estar separados por arquivos dos tipos redes sociais (redes_sociais.py) e sessões (sessoes.py).

Amostra 3:

Na amostra foram implementados os testes para de redes sociais (redes_sociais.py) e sessões (sessoes.py). Também foram implementadas as redes sociais GitHub junto a sessão “UploadCode” (UploadCodeSection) também na rede social Instagram. Os testes poderiam estar separados por arquivos dos tipos redes sociais (redes_sociais.py) e sessões (sessoes.py).

Code Review - Napp Academy

Semana 15

Amostra 1:

Na amostra sua implementação atende ao requisito (separar o conteúdo do CSV em diversos CSVs menores), faz uso de geradores ao invés de abrir todo o arquivo CVS antes de manipulá-lo. O código não apresenta testes unitários. E talvez não atenda a um requisito de “desenvolver um projeto”, já que se trata de código onde a extração só funcionaria nesse modelo de arquivo.

Amostra 2:

Na amostra sua implementação atende ao requisito (separar o conteúdo do CSV em diversos CSVs menores), não faz uso de geradores ao invés de abrir todo o arquivo CVS antes de manipulá-lo. Existem funções que não tem finalidade no arquivo, import de uma lib “write” não usada, e a lista de anos está colocada de maneira estática. O código não apresenta testes unitários. E talvez não atenda a um requisito de “desenvolver um projeto”, já que se trata de código onde a extração só funcionaria nesse modelo de arquivo.

Amostra 3:

Na amostra sua implementação atende ao requisito (separar o conteúdo do CSV em diversos CSVs menores), faz uso de geradores ao invés de abrir todo o arquivo CVS antes de manipulá-lo. Porém abre o arquivo a cada ano passado no for. O código não apresenta testes unitários. E talvez não atenda a um requisito de “desenvolver um projeto”, já que se trata de código onde a extração só funcionaria nesse modelo de arquivo.

Amostra 4:

Na amostra sua implementação atende ao requisito (separar o conteúdo do CSV em diversos CSVs menores), faz o uso da lib pandas, dessa forma não fazendo o uso de geradores. O código não apresenta testes unitários. E talvez não atenda a um requisito de “desenvolver um projeto”, já que se trata de código onde a extração só funcionaria nesse modelo de arquivo.

Code Review - Napp Academy

Amostra 5:

A amostra 5 (amostra5.py) atende ao requisito (separar o conteúdo do CSV em diversos CSVs menores), faz uso de geradores ao invés de abrir todo o arquivo CVS antes de manipulá-lo. Possui pontos de melhoria: o código percorre o arquivo CSV duas vezes. Na primeira vez que percorre o arquivo, descobre quais são os anos presentes (linhas 9-15). Na segunda vez, efetivamente faz a separação do conteúdo em arquivos distintos. Poderia ter refatorado para percorrer todo o arquivo CSV uma única vez. O código não apresenta testes unitários. E talvez não atenda a um requisito de “desenvolver um projeto”, já que se trata de código onde a extração só funcionaria nesse modelo de arquivo.