ANÁLISE CRÍTICA SOBRE O CÓDIGO 36

Autor: Ryan Almeida Silva RA:2040482412036

1. ENTENDIMENTO SOBRE A ATIVIDADE

Analisar o código enviado pelo e-mail e criar um relatório de avaliação com os critérios a seguir a fim de garantir que o código contenha: modularização; elementos conceituais como a utilização do camelcase no nome de variáveis e funções, comentários coerentes e objetivos;o uso de constantes para que o código não esteja com números fixos em suas linhas de código a fim de garantir que no futuro alterar o programa seja mais fácil; garantir também que o código segue as regras de negócio estipulados pelo cliente; garantir que haja tratamento de erros e que os elementos de negócio não estejam sendo feridos, essas são as chamadas regras implícitas.

Após o fim da análise, redigir um documento contendo o conteúdo do que foi analisado, neste relatório deve constar e entregar o programa funcionando de acordo com o que foi analisado e já refatorado, sendo um ambiente de programação de linguagem C da Microsoft.

2. MACRO ENTENDIMENTO DO PROBLEMA

Uma empresa solicitou a um programador um programa simples em C para uma loja de produtos eletrônicos. O programa deve ser capaz de incluir produtos e aplicar descontos a esse preço.

2.1. Objetivo do programa

Solicitar ao usuário que inclua um produto no sistema, permitindo o cadastro de novas informações. Também deve ser possível excluir um produto já existente.

Além disso, o sistema deve oferecer uma funcionalidade para consultar produtos, exibindo detalhes relevantes, como ID, nome, preço e quantidade em estoque. Outra funcionalidade importante é permitir que, posteriormente, o preço dos produtos cadastrados seja alterado de forma prática e segura, garantindo flexibilidade nas atualizações.

2.2. Regras de negócio explícitas

O programa é um CRUD, ou seja, deve conter as funcionalidades de Create, Read, Update e Delete, que traduzindo para o português e para este ambiente significa que o programa deve incluir, consultar, atualizar e deletar produtos.

Deve-se realizar a criação do programa em um ambiente de linguagem C, utilizando nesse caso o struct, cujo a estrutura já foi dada pelo orientador que conversou com o cliente, sendo esta estrutura ID, nome do produto, quantidade no estoque e valor do produto, tendo isso em mente, foram identificados os tipos de cada dado da estrutura sendo o ID, um inteiro; o nome do produto, uma string; a quantidade no estoque, um dado também de tipo inteiro; e por último o valor do produto deve ser um dado do tipo double. Além desse requisito técnico, há também o requisito do uso de ponteiros a fim de evitar a criação de múltiplas instâncias que podem causar um gasto desnecessário de memória.

2.3. Regras de negócio implícitas

No que tange às funcionalidades, todas elas devem ter um tratamento de erro e mensagens claras que facilitem o entendimento do usuário do programa, não sendo lacônico e nem prolixo.

As funcionalidades do programa podem possuir algumas exceções e elas devem ser corrigidas ainda no código, sendo assim as funções devem conter uma verificação de carácter ou de campo relacionado a estrutura antes já citada, ou seja, strings não podem permitir que os campos fiquem vazios, não se pode permitir números negativos e em alguns casos não se deve permitir zerar como é o caso da situação de inclusão de produtos.

Na questão das regras de negócio implícitas, dentro da estrutura deve conter um ID automático e não pode-se permitir a inclusão de produtos com o mesmo ID.

A função de deletar ou consultar um produto também não deve permitir a execução da ação com produtos inexistentes e sim apresentar uma resposta a exceção.

3. ANÁLISE DO CÓDIGO

No e-mail institucional foi enviado um código que no caso do autor da análise era o de número 36, tendo sido analisado de acordo com os critérios já citados. Esse código foi cuidadosamente verificado, levando em consideração todos os parâmetros estabelecidos anteriormente.

3.1 Modularização

O código foi bem modularizado e contém muitas funções que compõem o CRUD alterarProduto, que tem como função alterar os produtos que já existem no sistema; comprarProduto, que executa a função de comprar produtos; checarCodigoValido, responsável por verificar se o código inserido é válido; consultarProduto, que permite a consulta dos detalhes de um produto utilizando apenas o ID; confirmarResposta, que confirma a escolha do usuário em determinadas etapas do processo; excluirProduto, utilizada para remover produtos do sistema; incluirProduto, que permite a inclusão de novos produtos ao sistema; listarProdutos, que exibe uma lista de todos os produtos disponiveis e seus detalhes; menuOpcoes, que exibe o menu principal com as opções disponíveis para o usuário; e mudarProdutos, que facilita a alteração de atributos específicos dos produtos.

3.2 Elementos conceituais

O código seguiu bem as boas condutas relacionadas a programação, nota-se que o autor do código utilizou de camelcase em suas variáveis e em suas funções, manteve um bom padrão de identação, nomes relevantes de variáveis, utilizou de forma correta struct, também

fez uso das constantes e de ponteiros. No entanto, notou-se uma grande ausência de comentários, o código não possuía nenhum comentário em cerca de 250 linhas.

3.3 Elementos de negócio

No que se diz aos elementos de negócio que deveriam ser usados na hora de construir o programa no caso: a utilização de ponteiros e a estrutura solicitada pelo cliente, seguindo fielmente os tipos de dados que foram pedidos e atribuindo a eles seus devidos valores.

Dentro da área dos elementos de negócio relacionados à questão de funcionalidades, o programa atende a sua necessidade primária de criar, alterar, consultar e por fim deletar produtos, o programador ainda colocou duas funções a mais para o cliente sendo essas, as de listar os produtos e de comprar produtos. Logo, pode-se dizer que todas as funcionalidades estão funcionando adequadamente como deveriam

Avançando para as regras de negócio implícitas temos que o programador pensou devidamente que o ID tinha que ser auto incrementável, ou seja, adicionando um por um e não digitado pelo usuário e acima de tudo único, também pode-se observar tratamentos de erros nas funções de menu, compra, exclusão e consulta em que não se permite comprar menos que uma unidade, consultar ou excluir um ID que não existe e caracteres são inválidos caso fuja do padrão do menu. Além disso, a exclusão ainda conta com uma mensagem de interação com o usuário a fim de saber se deve ou não excluir realmente aquele produto. No entanto, percebe-se que ao executar esse código há uma grande falta de verificação de caracteres nas funções de alterar e incluir o que ao digitar um caractere errado e pressionar enter causam erros, sem falar na questão de que incluir não exibe uma mensagem com todos os detalhes e nem uma mensagem de confirmação.

Por último, o código tem um extenso bloco de código dentro da função principal que deveria estar destinada apenas para as funções e não para entrada e saída de dados, ou estruturas de laço de repetição.

4. SUGESTÕES DE MELHORIA

Como sugestões de melhoria para o código, pode-se dizer que a criação de uma função para reduzir o tamanho da função principal, já seria de grande ajuda, para auxiliar aqueles que possam querer entender o que se passa no código e como corrigi-lo eventualmente colocar comentários é uma boa ideia e também uma boa prática de programação.

Na questão dos usuários, criar mais confirmações principalmente para a função de incluir produtos, pois quando se tem uma confirmação a chance de erros ocorrerem é menor, além de exibir já todos os detalhes e não ser necessário colocar a lista. Também, com a ideia de minimizar erros e poupar tempo, colocar verificações de caractere a fim de evitar cadastros errados ou quebra do código.

É importante também ressaltar que quanto mais mensagens claras e objetivas o programa tiver, melhor. Portanto, considerar colocar mensagens como “Digite a sua opção escolhida aqui”, podem ser uma boa ideia tanto para pensar usuários mais leigos quanto para a estética do programa em si.

Outra coisa que é interessante resolver, é a questão da função comprarProduto, pois não se trata de um sistema em que o usuário é um comprador e sim aquele que vende, já que não faz sentido um cliente comprador adicionar, excluir, consultar e alterar produtos aplicando descontos. Então, essa mudança pode ser considerada algo muito importante, pois não está de acordo com a proposta do programa.

MUDANÇAS REALIZADAS NO CÓDIGO

- Redução do tamanho do int main() ao criar a função executarPrograma().
- Código com comentários.
- Mensagem de confirmação na inclusão junto a exibição de detalhes.
- Verificação de caracteres para a inclusão.
- Verificação de caracteres para a alteração.
- Alteração do nome da função comprarProduto para venderProduto.

CONCLUSÃO

Após o término da análise pode-se dizer que o código cumpre o seu papel muito bem e foi escrito de forma coerente com o que o cliente pediu, houveram apenas pequenos erros como é o exemplo da função de compra que na verdade deveria ser de venda, mas são pequenos erros que são fáceis de corrigir e que não anulam as principais funcionalidades. Portanto, após os testes e análise da estrutura do código, pode-se dizer que ele é bem robusto e está estruturado de uma forma boa e bem escrita, seguindo algumas das boas práticas de programação e modularizado.