Aluna: Rebecca Alessandra de Araújo Mendes

Matéria: Qualidade de Software e Governança

Curso: Análise e Desenvolvimento de Sistemas

Segundo Semestre- Matutino- Ceilândia

Relatório de Refatoração e Análise de Código - Sistema de Cadastro de Alunos

---

3. Refatoração de Código

3.1 Identificação de Problemas com Maior Impacto

Após revisar o código do sistema de cadastro de alunos de uma escola de idiomas, foram identificados problemas que afetam a **manutenibilidade** e **legibilidade** do código. Os principais problemas são:

1. **Passagem incorreta de parâmetros na função cadastrarAluno**:
   - O código original não passa o ponteiro necessário para cadastrarAluno, o que impede a função de acessar e modificar a lista global de alunos. Isso prejudica a funcionalidade do código e aumenta a complexidade de manutenção.
2. **Falta de controle de limite máximo de alunos (MAX_ALUNOS)**:
   - Não há uma verificação no código original para impedir o cadastro além do limite especificado. Isso pode levar a falhas, pois o programa pode acessar áreas de memória indevidas.
3. **Atribuição incorreta de contadorAlunos**:
   - contadorAlunos não é incrementado após o cadastro bem-sucedido, o que torna o controle dos IDs incorreto e dificulta a identificação de registros.
4. **Opção de escolha de idioma e validação**:
   - A entrada do idioma estava propensa a erros, pois dependia de uma entrada de texto que podia não corresponder exatamente aos nomes dos idiomas especificados. Isso comprometia a legibilidade e a funcionalidade, aumentando a possibilidade de erros de entrada.
5. **Configuração de setlocale para suporte a caracteres acentuados**:
   - A falta de configuração de setlocale limitava o suporte a caracteres acentuados e dificultava a exibição correta de textos em português, especialmente para sistemas que não possuem configuração regional.

3.2 Plano de Refatoração

Problema 1: Passagem Incorreta de Parâmetros para cadastrarAluno

Justificativa: A função cadastrarAluno foi definida para receber um ponteiro para Aluno, mas a chamada no main não passa esse argumento. Isso dificulta a leitura do código e faz com que o cadastro de alunos falhe, tornando o sistema pouco confiável.

Solução: Corrigir a chamada para cadastrarAluno no main, passando &alunos[contadorAlunos] como argumento para a função. Isso permite que cadastrarAluno preencha corretamente os dados do próximo aluno na lista.

Benefícios:

- **Legibilidade**: Fica claro que cadastrarAluno manipula uma estrutura Aluno existente.
- **Funcionalidade**: O código passa a cadastrar alunos corretamente.

Problema 2: Falta de Controle de Limite Máximo de Alunos

Justificativa: Sem controle do limite MAX_ALUNOS, o código pode permitir o cadastro de mais alunos do que o suportado, resultando em um comportamento indefinido e aumentando a complexidade de manutenção.

Solução: Inserir uma condição no main para verificar se o número total de alunos atingiu MAX_ALUNOS. Caso atinja, o programa deve exibir uma mensagem e evitar novos cadastros.

Benefícios:

- **Estabilidade**: Garante que o código não tente acessar áreas de memória indevidas.
- **Manutenção**: Simplifica o controle de limite para outros desenvolvedores.

Problema 3: Incremento Incorreto de contadorAlunos

Justificativa: contadorAlunos não é atualizado após cada cadastro. Isso causa um problema de contagem, pois o sistema pode reutilizar o mesmo índice, o que compromete o controle dos IDs e o gerenciamento de registros.

Solução: Incluir contadorAlunos++ após cada cadastro bem-sucedido.

Benefícios:

- **Legibilidade**: Cada aluno novo recebe um ID único e crescente.
- **Manutenibilidade**: Facilita o rastreamento e a modificação de registros.

Problema 4: Opção de Escolha de Idioma e Validação

Justificativa: A entrada do idioma era feita através de strings (textos) em uma função que pedia que o usuário digitasse o idioma diretamente. Isso podia causar falhas caso o usuário não escrevesse corretamente. Além disso, a abordagem inicial não era robusta o suficiente para uma funcionalidade de entrada de dados.

Solução: Converter a seleção de idioma para uma entrada numérica com switch para limitar as escolhas do usuário e garantir que apenas os idiomas desejados sejam cadastrados.

Benefícios:

- **Redução de Erros**: Ao usar uma entrada numérica, evitamos problemas com erros de digitação.
- **Manutenibilidade**: A lista de idiomas pode ser expandida facilmente, bastando adicionar novos case no switch.

Problema 5: Configuração de setlocale para Suporte a Caracteres Acentuados

Justificativa: A ausência de setlocale dificultava a exibição correta de caracteres acentuados, especialmente importante para sistemas em português.

Solução: Adicionar setlocale(LC_ALL, "Portuguese"); na função main para habilitar o suporte completo a caracteres acentuados.

Benefícios:

- **Legibilidade**: Permite a exibição correta de caracteres como ç e ã, melhorando a comunicação com o usuário.
- **Experiência do Usuário**: Facilita a interação com o sistema para falantes nativos.

3.3 Implementação e Validação

As mudanças foram implementadas no código, e testes foram realizados para confirmar que o sistema mantém o comportamento esperado:

- Testes manuais foram feitos para verificar a criação de registros, o controle de limite e o ID dos alunos.
- O código se comportou conforme esperado após a refatoração, sem alterar as funcionalidades originais.

---

4. Relatório Final

4.1 Descrição do Projeto

O projeto é um sistema básico de **cadastro de alunos** para uma escola de idiomas, permitindo o gerenciamento de dados como nome, idade e idioma estudado (Inglês, Espanhol ou Francês). Este projeto simula um sistema de gestão estudantil em linguagem C, explorando o uso de estruturas e funções básicas. O repositório do projeto está disponível em: [link para o repositório].

4.2 Relatório de Análise Estática

A análise estática foi realizada com foco nas seguintes métricas:

- **Complexidade Ciclomática**: Mantida baixa ao dividir as funções de modo modular.
- **Cobertura de Código**: Testes manuais foram realizados para garantir a cobertura completa das funcionalidades.
- **Dívida Técnica**: Corrigidos problemas estruturais e de controle, como passagem de argumentos e limites de memória.
- **Código Duplicado**: Não foram encontradas duplicações desnecessárias.
- **Más Práticas**: Corrigidas práticas inadequadas, como a falta de controle de limites e erros de passagem de parâmetros.

4.3 Plano de Refatoração

As principais alterações incluídas no plano foram:

1. Passagem correta de parâmetros para a função cadastrarAluno, permitindo o cadastro correto de alunos.
2. Controle do limite máximo de alunos (MAX_ALUNOS), prevenindo problemas de overflow.

3. Incremento de contadorAlunos após cada cadastro, assegurando IDs corretos e não repetidos.
4. Conversão da escolha de idioma para uma entrada numérica com switch, evitando problemas de digitação.
5. Configuração de setlocale para exibir caracteres acentuados corretamente.

Essas refatorações melhoraram a **legibilidade**, **estabilidade**, e **manutenção** do projeto, facilitando futuras adições de funcionalidades.

4.4 Código Refatorado

O código atualizado com as mudanças foi enviado ao repositório. As alterações estão disponíveis no branch “refactor”, acessível no link: [link para o branch de refatoração].

---

5. Apresentação

Slide 1: Visão Geral do Projeto e Análise

- **Propósito**: Sistema para gerenciamento de alunos em uma escola de idiomas.
- **Métricas**: Complexidade baixa, cobertura de código adequada e eliminação de más práticas.

Slide 2: Problemas Encontrados

- **Passagem de Parâmetros**: Código original não passava ponteiro necessário.
- **Controle de Limite**: Sem verificação do número máximo de alunos.
- **Incremento de Contador**: Falta de atualização após o cadastro, comprometendo o gerenciamento de IDs.
- **Escolha de Idioma e Validação**: Propensa a erro, exigindo entrada textual.
- **Configuração de setlocale**: Ausente, prejudicando a exibição de caracteres especiais.

Slide 3: Plano de Refatoração e Implementação

- **Passagem Correta de Parâmetros**: Permitiu manipulação correta da lista de alunos.
- **Controle de Limite Máximo**: Previne falhas ao evitar overflow.
- **Incremento de Contador**: Assegura IDs únicos e facilita o gerenciamento de registros.
- **Escolha Numérica para Idioma**: Reduz erros e simplifica a entrada.
- Uso de setlocale