

[av1:ap3] Atividade Prática 3: Usando structs, functions e alocação dinâmica

Disciplina: Estrutura de Dados

Prof.: Jean Nunes

Aluno(a): ____

Instruções:

- É permitido e recomendado a utilização de códigos desenvolvidos durante as atividades da disciplina.
- O uso de IA e de códigos da *internet* implicará em um decréscimo de 50% na nota final.
- Organize seu código de forma clara e bem comentada.
- Certifique-se de que seu código compila e executa corretamente.

Objetivos

- Utilizar `structs` para organizar os dados de clientes e empréstimos.
- Implementar funções para modularizar as funcionalidades do programa.
- Aplicar ponteiros para manipular dados e alocar memória dinamicamente.
- Calcular o valor da parcela do empréstimo considerando a taxa de juros de `5%`.
- Implementar uma lógica de aprovação/reprovação de empréstimos baseada na relação parcela/salário, onde: se o valor da parcela for maior que `*20%*` do salário, o empréstimo será reprovado, caso contrário, o empréstimo será aprovado.
- Manter um histórico de empréstimos para cada cliente. Se o cliente tiver 1 ou mais empréstimos ativos, o valor da parcela desse empréstimo ativo deverá ser somado a parcela do novo empréstimo para fins de implementação da lógica de aprovação/reprovação.

Materiais Fornecidos:

- `main.c` : Arquivo principal do programa.
- `utils.c` : Arquivo com funções utilitárias (a ser complementado).
- `utils.h` : Arquivo de cabeçalho com declarações de tipos e funções.
- `clientes.csv` : Arquivo contendo cadastro inicial de clientes.
- `emprestimos.csv` : Arquivo contendo cadastro inicial de empréstimos.

Funções a serem implementadas

Você deverá implementar as seguintes funções no arquivo `utils.c` :

Funções de Realocação de Memória

- `Cliente *realocar_memoria_cliente(Cliente *clientes, int novo_tamanho);`
 - Esta função deverá receber um ponteiro para um array de estruturas `Cliente` e um novo tamanho para este array.
 - Ela deve alocar ou realocar memória para o array de clientes para o novo tamanho especificado.
 - A função deve retornar um ponteiro para o novo bloco de memória alocado.
- `Emprestimo* realocar_memoria_emprestimo(Emprestimo *emprestimos, int novo_tamanho);`
 - Esta função deverá receber um ponteiro para um array de estruturas `Emprestimo` e um novo tamanho para este array.
 - Ela deve alocar ou realocar memória para o array de empréstimos para o novo tamanho especificado.
 - A função deve retornar um ponteiro para o novo bloco de memória alocado.

Funções de Cadastro e Solicitação

- `Cliente *cadastrar_novo_cliente(Cliente *clientes, int *num_clientes);`
 - Esta função deverá permitir o cadastro de um novo cliente.
 - Ela deverá solicitar ao usuário as informações necessárias para criar um novo cliente (nome, CPF, etc.).
 - A função deverá utilizar a função `realocar_memoria_cliente` para aumentar o tamanho do array de clientes, se necessário. (A função deverá retornar o ponteiro atualizado para o array de clientes e incrementar o número total de clientes (`num_clientes`)).
- `void solicitar_novo_emprestimo(Cliente *clientes, int num_clientes);`
 - Esta função deverá permitir que um cliente existente solicite um novo empréstimo.
 - Ela deverá solicitar ao usuário o CPF do cliente solicitante e verificar se ele existe no sistema.
 - Caso o cliente exista, a função deverá solicitar os detalhes do empréstimo (valor, número de parcelas, taxa de juros, etc.).
 - Após a entrada dos dados, a função deverá chamar `calcular_valor_parcela` e `aprovar_reprovar_emprestimo` para processar a solicitação.

Funções de Cálculo e Aprovação

- `void calcular_valor_parcela(Emprestimo *emprestimo);`
 - Esta função deverá receber um ponteiro para uma estrutura `Emprestimo`.
 - Ela deverá calcular o valor de cada parcela do empréstimo com base no valor total, número de parcelas e taxa de juros (a fórmula de cálculo deve ser implementada).
 - O valor da parcela deve ser armazenado na estrutura `Emprestimo`.
- `void aprovar_reprovar_emprestimo(Cliente *cliente, Emprestimo *novo_emprestimo);`
 - Esta função deverá receber um ponteiro para a estrutura `Cliente` que solicitou o empréstimo e um ponteiro para a estrutura `Emprestimo` recém-criada.
 - Ela deverá implementar uma lógica de aprovação ou reprovação do empréstimo (por exemplo, com base na renda do cliente, histórico de crédito, etc. - a lógica específica fica a critério do aluno).
 - A função deverá atualizar o status do empréstimo na estrutura `Emprestimo` (aprovado ou reprovado) e, opcionalmente, adicionar o empréstimo à lista de empréstimos do cliente (se implementado).

Critérios de Avaliação

- A avaliação considerará os seguintes aspectos:
 - Implementação adequada das funções solicitadas.
 - Utilização adequada de alocação dinâmica de memória.
 - Tratamento de erros e entradas inválidas.
 - Cálculo correto do valor da parcela e lógica de aprovação.
 - Organização, legibilidade e comentários no código.
 - Funcionamento correto do sistema.

Uso de IA e consulta a códigos da *internet*

- [] Declaro que os códigos fornecidos por mim para resolver esta atividade prática são de minha autoria. Não utilizei inteligência artificial, ferramentas correlatas ou códigos da *internet* para resolver os problemas apresentados.
- [] Declaro que os códigos fornecidos por mim para resolver esta atividade prática não são inteiramente de minha autoria, pois utilizei inteligência artificial, ferramentas correlatas ou códigos da internet para ajudar a resolver os problemas apresentados. Estou ciente de que o uso de tais recursos resultará em um decréscimo de 50% na minha nota final.