

**RELATÓRIO DE PROJETO DE FUNDAMENTOS DE PROGRAMAÇÃO**

Curso Técnico Superior Profissional em Programação de Sistemas de Informação

Lucas Emanuel Santos Siqueira nº2231435  
Maxim Dudai nº 2232358

Relatório realizado sob a orientação do Professor Davide Francisco e Professora Anabela Bernardino

Janeiro de 2024

**Resumo**

Durante este trabalho, iremos explicar como foi realizado o projeto requisitado pela disciplina de Fundamentos de Programação no ano letivo 2022/2023, utilizando os conhecimentos adquiridos ao longo do semestre. O objetivo principal é concluir todas as tarefas solicitadas de maneira simples, evitando repetição de código e seguindo as boas práticas. Abordaremos as estruturas de dados utilizadas, variáveis e constantes principais, funcionalidades implementadas e suas respetivas funções, funcionalidades pendentes, validação dos dados de entrada e realização de testes na aplicação. Por fim, apresentaremos os aspetos importantes a serem destacados e uma análise crítica sobre o projeto.

**Índice**

[1. Índice de Figuras IV](#_Toc156082713)

[2. Introdução 1](#_Toc156082714)

[1. Constituição da estrutura do programa 2](#_Toc156082715)

[1.1 Definições de pré-processamento 2](#_Toc156082716)

[1.2 Constantes 2](#_Toc156082717)

[1.3 Declaração de Estruturas 5](#_Toc156082718)

[1.4 Declaração de Funções 6](#_Toc156082719)

[1.5 Variáveis principais 7](#_Toc156082720)

[2. Funcionalidades implementadas e respetivas funções 9](#_Toc156082721)

[2.1 Registo e Consulta dos dados dos Estudantes 9](#_Toc156082722)

[2.2 Registo e consulta dos dados das Unidades Curriculares 12](#_Toc156082723)

[2.3 Registo e consulta dos dados das avaliações 14](#_Toc156082724)

[2.4 Consultar Estatísticas 17](#_Toc156082725)

[2.4 Guardar e Ler de ficheiros binários 21](#_Toc156082726)

[3. Funcionalidades por implementar 24](#_Toc156082727)

[4. Validação de dados de entrada 25](#_Toc156082728)

[Conclusão Validação de Dados 27](#_Toc156082729)

[5. Realização de testes na aplicação 28](#_Toc156082730)

[6. Conclusão 29](#_Toc156082731)

[7. Referências bibliográficas 30](#_Toc156082732)

# Índice de Figuras

[Figura 1 - Definições de pré-processamento 2](#_Toc156082695)

[Figura 2- Constantes 3](file:////Users/barbaracelina/Downloads/Relatorio_Projeto_FP.docx#_Toc156082696)

[Figura 3- Declaração de Estruturas 5](#_Toc156082697)

[Figura 4- Declaração de funções 6](#_Toc156082698)

[Figura 5- Variáveis principais 8](#_Toc156082699)

[Figura 6- Registo estudantes 9](#_Toc156082700)

[Figura 7- consulta estudantes 11](#_Toc156082701)

[Figura 8 - registar UC 12](#_Toc156082702)

[Figura 9 - Consultar UC 13](#_Toc156082703)

[Figura 10 - Registar avaliações 15](#_Toc156082704)

[Figura 11 - Consultar avaliações 16](#_Toc156082705)

[Figura 12 - Consultar percentagem ECTS aprovados 17](#_Toc156082706)

[Figura 13 - Consultar ECTS 19](#_Toc156082707)

[Figura 14 - Consultar média ECTS 20](#_Toc156082708)

[Figura 15 - Ler ficheiro 22](#_Toc156082709)

[Figura 16-Teste Menu Principal 28](#_Toc156082710)

[Figura 17-Teste registar e consultar dados dos estudantes 28](#_Toc156082711)

[Figura 20-Teste guardar dados 28](#_Toc156082712)

# Introdução

Neste relatório, desenvolvido como parte da disciplina de Fundamentos de Programação (FP) no Curso Técnico Superior (CTesP) de Programação de Sistemas de Informação (PSI), o objetivo é aplicar os conhecimentos adquiridos durante as aulas de FP.

O projeto visa criar um protótipo de um aplicativo que permite gerenciar as inscrições em atividades realizadas pelas várias Associações de Estudantes do Politécnico de Leiria, utilizando as ferramentas fornecidas pela disciplina e aplicando os conhecimentos adquiridos em sala.

Este extenso documento aborda oito capítulos visando facilitar a compreensão completa, iniciando pela estrutura abrangente do projeto com os fundamentos essenciais para o seu desenvolvimento, subsequentemente a implementação de funcionalidades e respetivas funções, prosseguindo com um capítulo eventual sobre possíveis imprevistos, findando com a validação pertinente de dados e realização de testes para analisar criticamente e extrair conclusões lastreadas, relacionando igualmente referências bibliográficas pertinentes.

# Constituição da estrutura do programa

Nesta parte, apresentaremos uma visão geral da arquitetura do projeto. Abordaremos tópicos como pré-processamento, constantes utilizadas, estruturas de dados declaradas e funções implementadas, de modo a propiciar uma compreensão mais aprofundada da estrutura do programa. Buscamos explicar quais componentes foram incluídos e por que motivos, de forma a tornar transparente a lógica subjacente ao código. Esperamos que esta descrição auxilie na leitura e interpretação do código-fonte, permitindo ao leitor identificar a finalidade de cada seção e entender como todos os elementos se relacionam para compor a solução desenvolvida.

### Definições de pré-processamento

As declarações de pré-processamento, também conhecidas como bibliotecas, estão declaradas logo no início do programa e podem ser vistas na imagem abaixo.

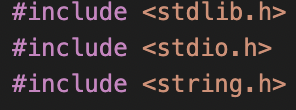


Figura - Definições de pré-processamento

As primeiras duas definições de pré-processamento já vêm inseridas automaticamente quando um novo projeto é criado. Porém, outras três foram incluídas para permitir o uso de funcionalidades específicas, tais como:

* A <locale.h> garantirá a exibição adequada de caracteres especiais e acentuação em nosso código.
* A <string.h> ativará funções para manipulação de strings.
* A <time.h> permitirá a obtenção automática da data e hora no momento do cadastro.

### Constantes

As Constantes definidas durante o projeto são as visíveis na imagem abaixo:

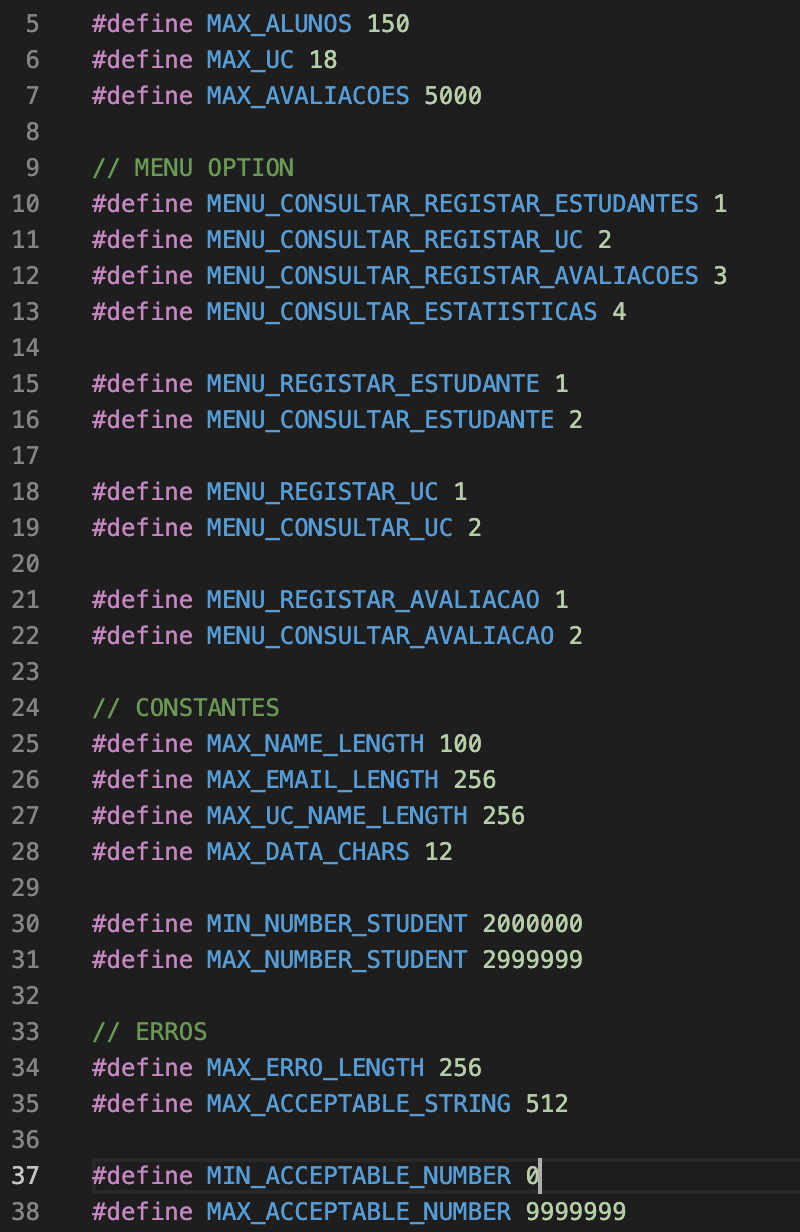


Figura 2- Constantes

As constantes representam um valor fixo que permanece inalterado durante toda a execução do programa, podendo ser acessado em diferentes partes do código. Embora imutável internamente, externamente é simples modificar uma constante caso necessário, atualizando instantaneamente todas as instâncias no programa. Foram empregadas para conferir:

**Constantes de Limite:**

MAX\_ALUNOS: Define o número máximo de alunos que podem ser registrados (150).

MAX\_UC: Define o número máximo de unidades curriculares que podem ser registradas (18).

MAX\_AVALIACOES: Define o número máximo de avaliações que podem ser registradas (5000).

**Constantes para Menu:**

MENU\_CONSULTAR\_REGISTAR\_ESTUDANTES

MENU\_CONSULTAR\_REGISTAR\_UC

MENU\_CONSULTAR\_REGISTAR\_AVALIACOES

MENU\_CONSULTAR\_ESTATISTICAS

Constantes que representam as opções do menu principal.

MENU\_REGISTAR\_ESTUDANTE

MENU\_CONSULTAR\_ESTUDANTE

MENU\_REGISTAR\_UC MENU\_CONSULTAR\_UC

MENU\_REGISTAR\_AVALIACAO

MENU\_CONSULTAR\_AVALIACAO

Constantes que representam as opções dos submenus.

**Constantes para Dados:**

MAX\_NAME\_LENGTH: Tamanho máximo para o nome do estudante.

MAX\_EMAIL\_LENGTH: Tamanho máximo para o endereço de e-mail.

MAX\_UC\_NAME\_LENGTH: Tamanho máximo para o nome da unidade curricular.

MAX\_DATA\_CHARS: Tamanho máximo para a string que armazena a data.

**Constantes para Identificação:**

MIN\_NUMBER\_STUDENT: Número mínimo permitido para identificação do estudante.

MAX\_NUMBER\_STUDENT: Número máximo permitido para identificação do estudante.

**Constantes para Erros:**

MAX\_ERRO\_LENGTH: Tamanho máximo para a mensagem de erro.

MAX\_ACCEPTABLE\_STRING: Tamanho máximo para uma string aceitável.

MIN\_ACCEPTABLE\_NUMBER: Número mínimo aceitável.

MAX\_ACCEPTABLE\_NUMBER: Número máximo aceitável.

### Declaração de Estruturas

As estruturas são agrupamentos de variáveis com tipos de dados distintos sob um único rótulo, em que cada variável é denominada por campo. Estas são declaradas antes da função principal para serem utilizadas no programa. Neste código, utilizou-se o typedef para facilitar o uso das estruturas, uma vez que este recurso permite atribuir um apelido a um determinado tipo de dados já existente. As variáveis das estruturas podem conter valores diversos, como números, strings ou outros dados, e os campos podem ser acessados de forma individual para leitura ou escrita. Desta forma, as estruturas proporcionam uma organização dos dados em blocos lógicos, agrupando informações relacionadas sob um único nome, o que torna o código mais legível e de fácil manutenção.

De seguida podemos visualizar na imagem abaixo a declaração das estruturas.

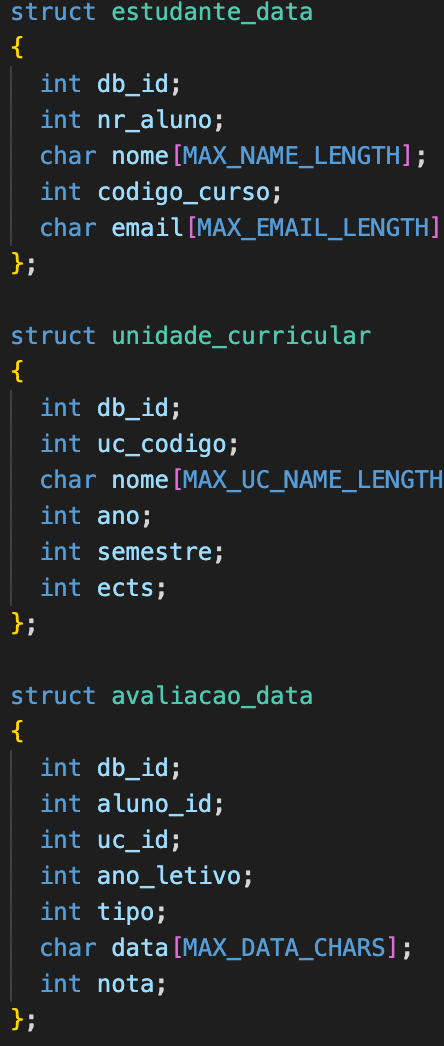


Figura - Declaração de Estruturas

*struct estudante\_data:* Armazena dados relacionados a estudantes.

*struct unidade\_curricular*: Armazena dados relacionados a unidades curriculares.

*struct avaliacao\_data:* Armazena dados relacionados a avaliações.

As estruturas foram declaradas para cumprir o objetivo do projeto sem repetição de código e sem utilizar várias variáveis para poder cumprir determinados objetivos.

### Declaração de Funções

Como solicitado no enunciado, de forma a facilitar a correta estruturação, implementação, entendimento e de forma a evitar a redundância e repetição de código, o programa foi dividido em várias funções, estas podem ser vistas na imagem abaixo.

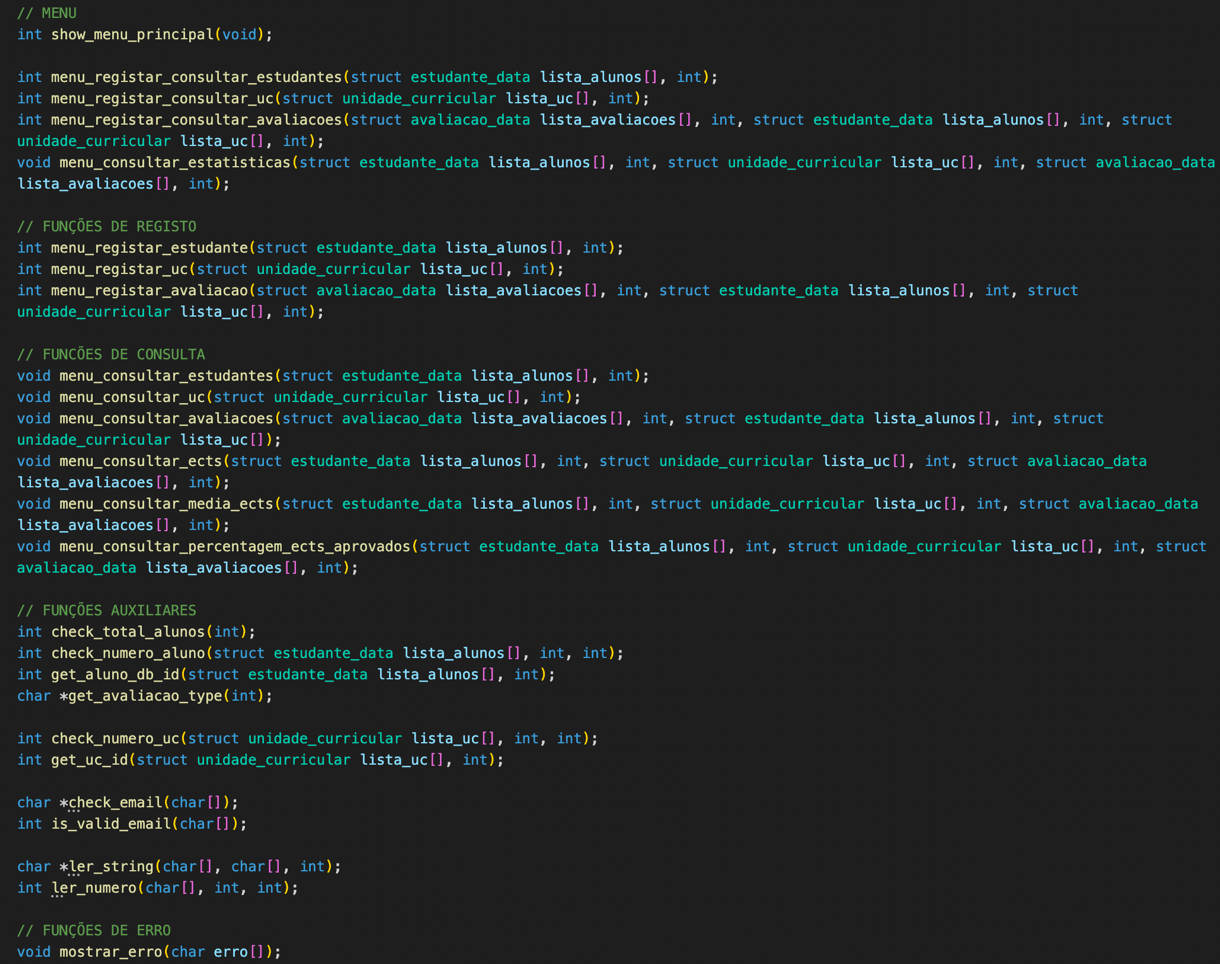


Figura - Declaração de funções

**Funções:**

*show\_menu\_principal:* Apresenta o menu principal.

*menu\_registar\_consultar\_estudantes*

*menu\_registar\_consultar\_uc*

*menu\_registar\_consultar\_avaliacoes*

*menu\_consultar\_estatisticas*

Funções que representam as opções do menu principal.

*menu\_registar\_estudante*

*menu\_consultar\_estudantes*

*menu\_registar\_uc*

*menu\_consultar\_uc*

*menu\_registar\_avaliacao*

*menu\_consultar\_avaliacoes*

*menu\_consultar\_ects*

*menu\_consultar\_media\_ects*

*menu\_consultar\_percentagem\_ects\_aprovados*

Funções relacionadas a registrar e consultar dados.

**Funções Auxiliares:**

*check\_total\_alunos, check\_numero\_aluno, get\_aluno\_db\_id:* Funções auxiliares para verificar e obter informações sobre estudantes.

*check\_numero\_uc, get\_uc\_id:* Funções auxiliares para verificar e obter informações sobre unidades curriculares.

*ler\_string, ler\_numero, check\_email, is\_valid\_email:* Funções auxiliares para entrada de dados e validação.

Foram utilizadas um total de vinte e cinco funções (sem contar com a main), e nenhuma delas é uma função sem parâmetros e que não devolvem qualquer valor, isto é, podem ser uma função com parâmetros, uma função que devolve algum valor ou ambos.

### Variáveis principais

As variáveis principais estão na função main() e podem ser vistas na imagem abaixo:

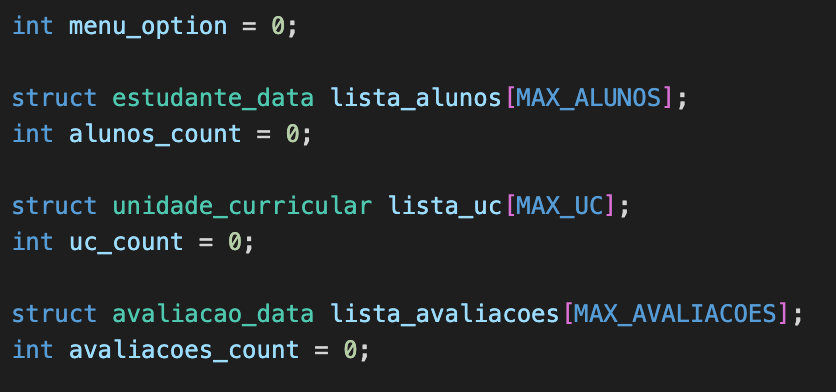


Figura - Variáveis principais

Estas são as variáveis principais porque são elas que vão ser precisas e utilizadas em todo o programa. Quer seja para contabilizar o número de alunos, unidades curriculares , gerir o o menu e submenus e as “estruturas” para os alunos, atividades e inscrições com as respetivas constantes a representar o máximo de cada uma.

# Funcionalidades implementadas e respetivas funções

As funcionalidades implementadas foram todas introduzidas no programa de forma a cumprir o solicitado de forma simples, sendo algumas introduzidas com algumas melhorias, foram cumpridas todas as funcionalidades solicitadas como podemos ver a seguir.

### Registo e Consulta dos dados dos Estudantes

Na imagem abaixo podemos visualizar a função para registar estudantes

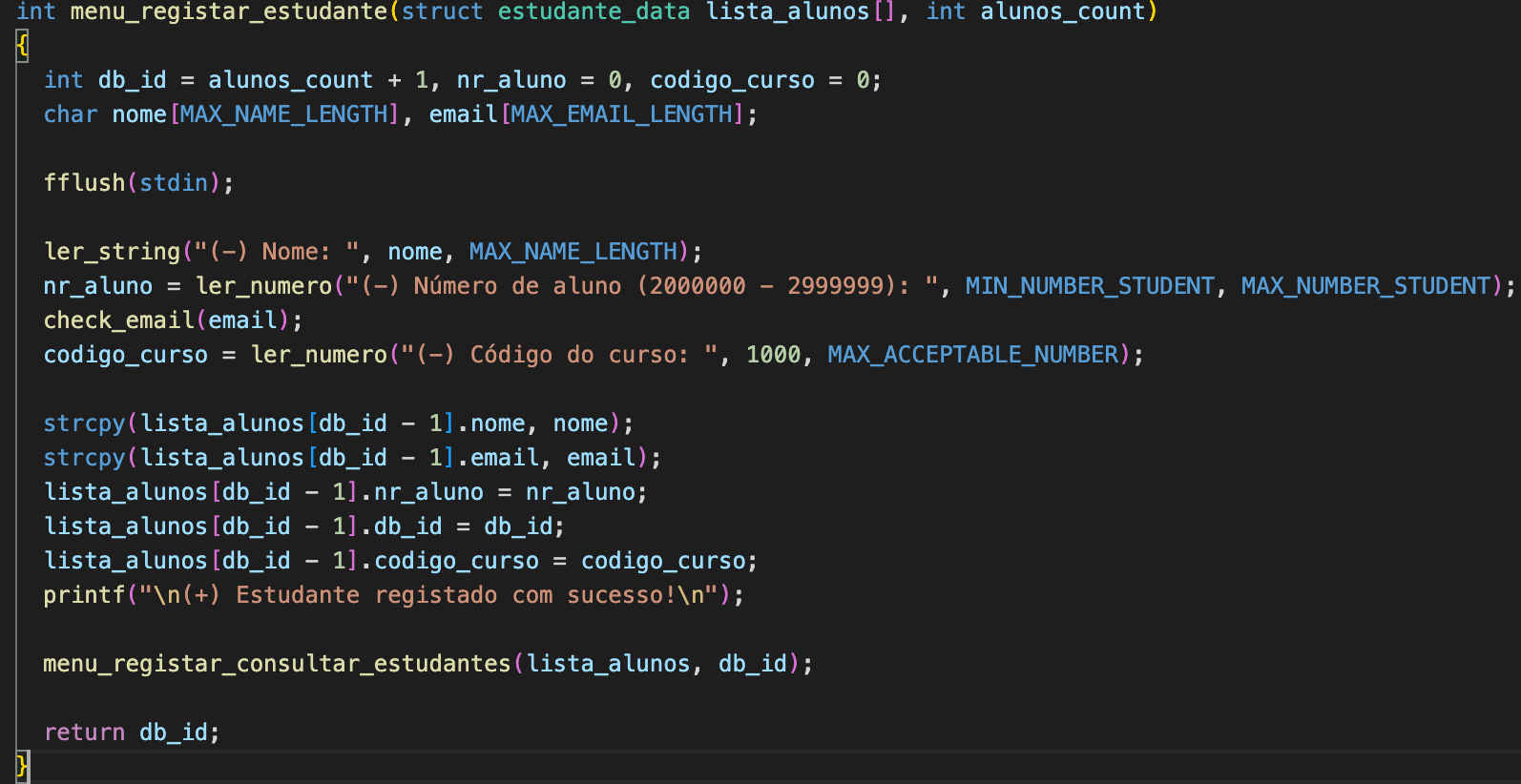


Figura - Registo estudantes

**Inicialização de Variáveis:**

db\_id: Identificador único para o estudante no sistema, incrementado a partir do número total de estudantes registados.

nr\_aluno: Número de aluno atribuído ao estudante.

codigo\_curso: Código do curso associado ao estudante.

nome e email: Arrays para armazenar o nome e e-mail do estudante, respeitando os tamanhos máximos definidos.

**Limpeza do Buffer do Teclado:**

fflush(stdin): Utilizado para limpar o buffer do teclado, garantindo uma entrada de dados sem problemas.

**Entrada de Dados:**

Utilização de funções específicas (ler\_string e ler\_numero) para capturar o nome, número do aluno e código do curso, garantindo validações adequadas.

**Validação de E-mail:**

check\_email(email): Função não fornecida, mas presumivelmente valida o formato do e-mail.

**Atribuição aos Dados do Estudante:**

Os dados capturados são atribuídos à estrutura de dados do estudante no array lista\_alunos.

**Mensagem de Sucesso:**

Exibe uma mensagem indicando o sucesso no registo do estudante.

**Chamada de Função para Menu de Registo e Consulta:**

menu\_registar\_consultar\_estudantes: Função que trata do menu para registar e consultar estudantes.

**Retorno do Identificador do Estudante Registado:**

Retorna o db\_id como identificador único do estudante registado.

Na imagem abaixo podemos visualizar a função para consultar estudantes

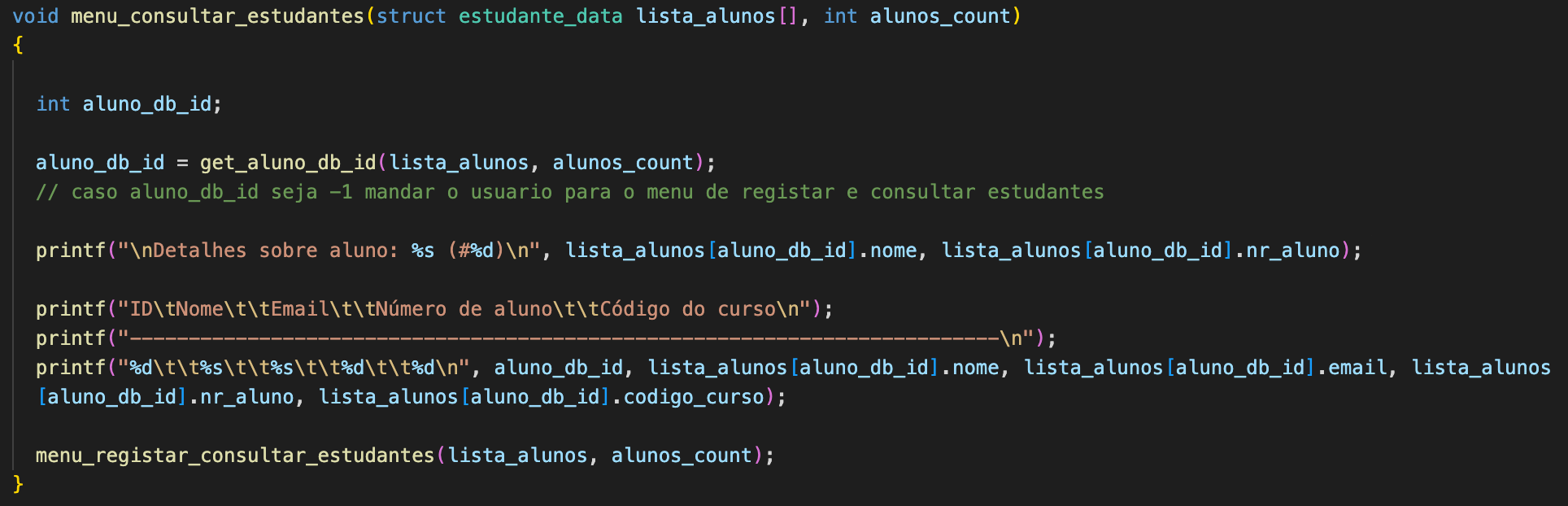


Figura - consulta estudantes

**Obtenção do Identificador do Aluno:**

aluno\_db\_id = get\_aluno\_db\_id(lista\_alunos, alunos\_count): Chama uma função get\_aluno\_db\_id (não fornecida) para obter o identificador único do aluno a ser consultado.

**Verificação do Resultado da Consulta:**

if (aluno\_db\_id == -1): Verifica se o identificador do aluno é -1, indicando que o aluno não foi encontrado.

Caso o aluno não seja encontrado, exibe uma mensagem informativa e redireciona o usuário para o menu de registar e consultar estudantes.

**Exibição dos Detalhes do Aluno:**

printf("\nDetalhes sobre aluno: %s (#%d)\n", lista\_alunos[aluno\_db\_id].nome, lista\_alunos[aluno\_db\_id].nr\_aluno): Apresenta detalhes básicos sobre o aluno consultado, como nome e número de aluno.

**Exibição da Tabela de Detalhes:**

printf("ID\tNome\t\tEmail\t\tNúmero de aluno\t\tCódigo do curso\n"): Exibe o cabeçalho da tabela de detalhes.

printf("%d\t\t%s\t\t%s\t\t%d\t\t%d\n", aluno\_db\_id, lista\_alunos[aluno\_db\_id].nome, lista\_alunos[aluno\_db\_id].email,lista\_alunos[aluno\_db\_id].nr\_aluno, lista\_alunos[aluno\_db\_id].codigo\_curso) : Exibe os detalhes do aluno, como ID, nome, e-mail, número de aluno e código do curso.

**Chamada do Menu de Registar e Consultar Estudantes:**

menu\_registar\_consultar\_estudantes(lista\_alunos, alunos\_count): Chama o menu de registar e consultar estudantes, permitindo ao usuário realizar operações adicionais.

Essa função é essencial para permitir que os utilizadores consultem detalhes específicos sobre os estudantes no sistema, proporcionando uma visão rápida e organizada das informações relevantes. O redireccionamento para o menu de registar e consultar estudantes em caso de aluno não encontrado melhora a usabilidade e a experiência do utilizador.

### Registo e consulta dos dados das Unidades Curriculares

A função menu\_registar\_uc tem como objetivo registar uma nova unidade curricular no sistema. Vamos analisar e explicar cada parte dessa função.

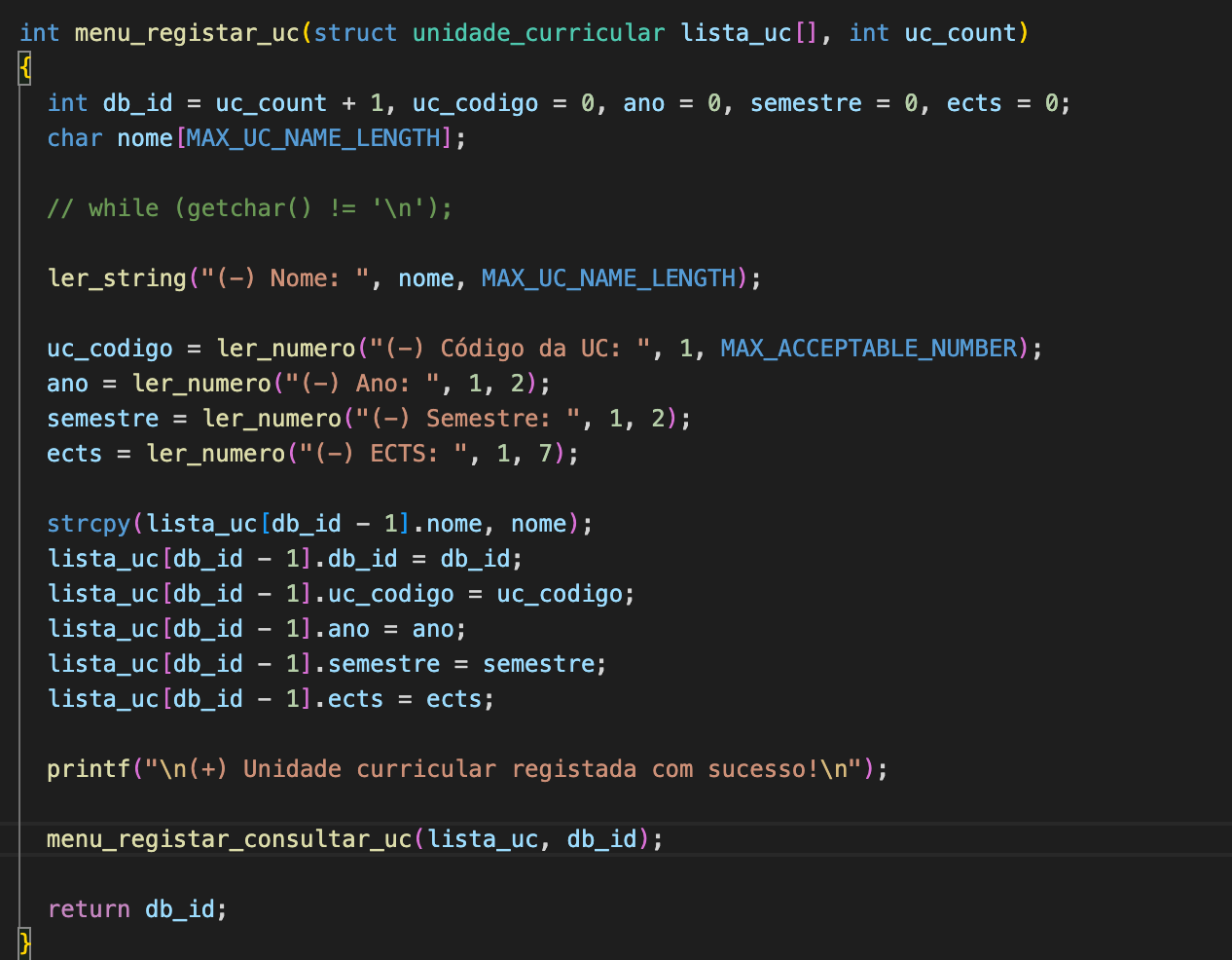


Figura - registar UC

**Leitura dos Dados:**

A função utiliza a função ler\_string e ler\_numero para obter informações sobre a nova unidade curricular, como nome, código, ano, semestre e ECTS.

**Preenchimento da Estrutura:**

Os dados lidos são então utilizados para preencher a estrutura lista\_uc na posição correspondente ao novo identificador único (db\_id).

**Mensagem de Sucesso:**

É exibida uma mensagem indicando que a unidade curricular foi registada com sucesso.

**Chamada do Menu de Registar e Consultar UC:**

A função menu\_registar\_consultar\_uc é chamada para permitir que o utilizador realize operações adicionais relacionadas às unidades curriculares.

**Retorno do Identificador Único:**

O identificador único da UC registada é retornado, possibilitando seu uso em outras partes do programa.

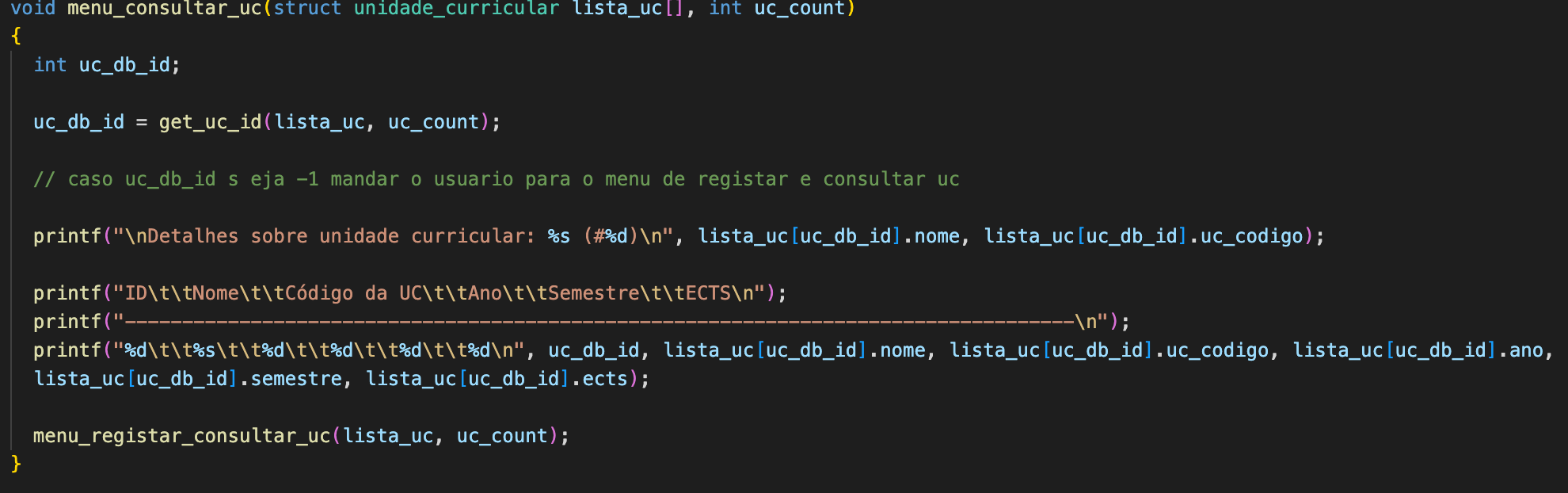
A função menu\_consultar\_uc tem como objetivo permitir a consulta de detalhes sobre uma unidade curricular específica no sistema. Vamos analisar e explicar cada parte dessa função.

Figura - Consultar UC

A função menu\_consultar\_uc realiza tarefas semelhantes à função de consultar estudantes.

**Obtenção do Identificador da UC:**

uc\_db\_id = get\_uc\_id(lista\_uc, uc\_count): Chama uma função get\_uc\_id (não fornecida) para obter o identificador único da UC a ser consultada.

**Verificação do Resultado da Consulta:**

if (uc\_db\_id == -1): Verifica se o identificador da UC é -1, indicando que a UC não foi encontrada.

Caso a UC não seja encontrada, exibe uma mensagem informativa e redireciona o utilizador para o menu de registar e consultar UC.

**Exibição dos Detalhes da UC:**

printf("\nDetalhes sobre unidade curricular: %s (#%d)\n", lista\_uc[uc\_db\_id].nome, lista\_uc[uc\_db\_id].uc\_codigo): Apresenta detalhes básicos sobre a UC consultada, como nome e código.

**Exibição da Tabela de Detalhes:**

printf("ID\t\tNome\t\tCódigo da UC\t\tAno\t\tSemestre\t\tECTS\n"): Exibe o cabeçalho da tabela de detalhes.

printf("%d\t\t%s\t\t%d\t\t%d\t\t%d\t\t%d\n", uc\_db\_id, lista\_uc[uc\_db\_id].nome, lista\_uc[uc\_db\_id].uc\_codigo, lista\_uc[uc\_db\_id].ano, lista\_uc[uc\_db\_id].semestre, lista\_uc[uc\_db\_id].ects): Exibe os detalhes da UC, como ID, nome, código, ano, semestre e ECTS.

**Chamada do Menu de Registar e Consultar UC:**

menu\_registar\_consultar\_uc(lista\_uc, uc\_count):

### Registo e consulta dos dados das avaliações

A função *menu\_registar\_avaliacao* é responsável por registar uma nova avaliação no sistema. Vamos analisar e explicar cada parte dessa função.

Na imagem abaixo podemos visualizar a função para registar avaliações:

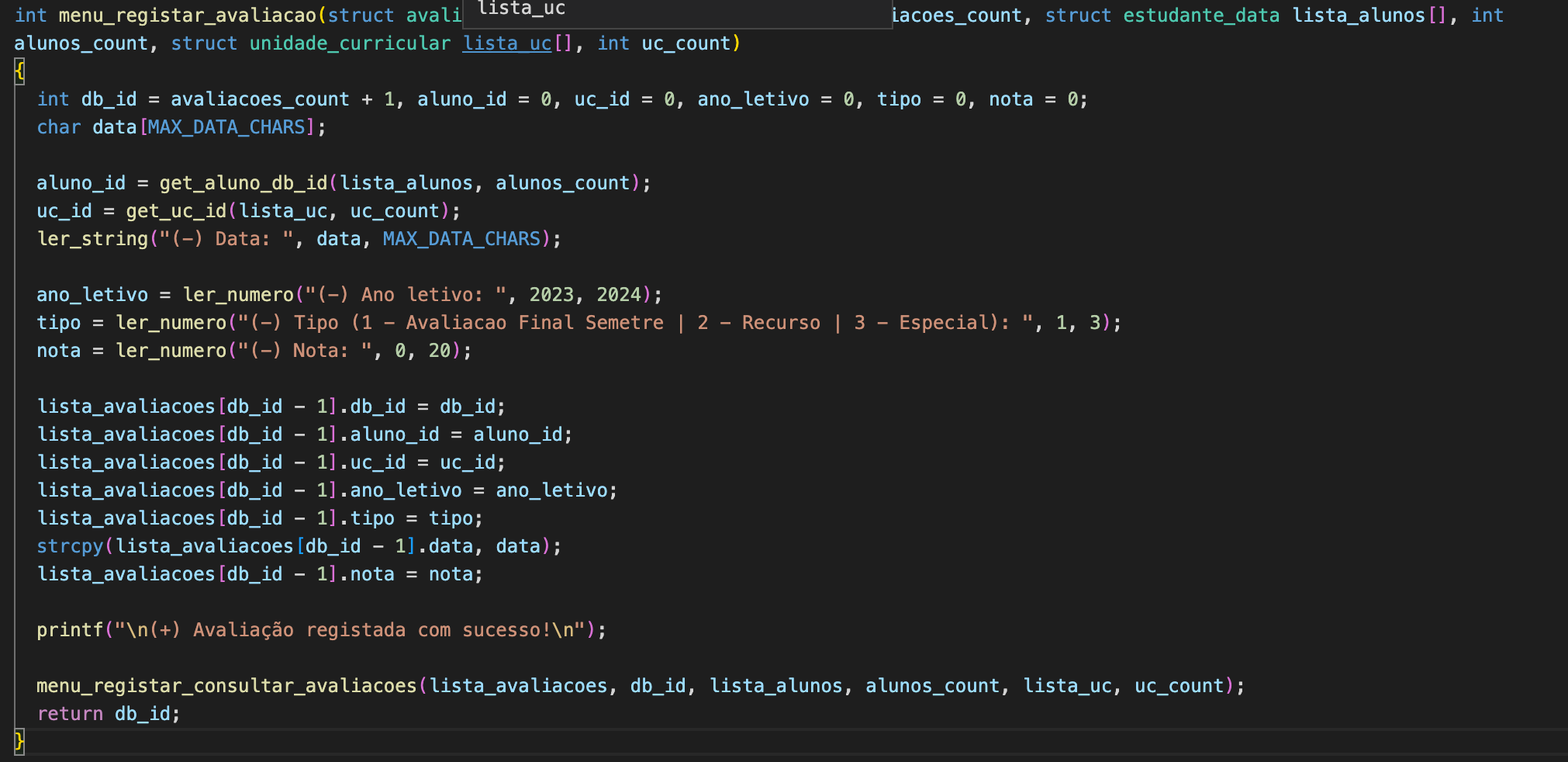


Figura - Registar avaliações

**Obtenção de Identificadores:**

aluno\_id = get\_aluno\_db\_id(lista\_alunos, alunos\_count): Obtém o identificador do aluno chamando a função get\_aluno\_db\_id.

uc\_id = get\_uc\_id(lista\_uc, uc\_count): Obtém o identificador da unidade curricular chamando a função get\_uc\_id.

**Leitura de Dados:**

Utiliza funções auxiliares como ler\_string e ler\_numero para obter informações sobre a nova avaliação, como data, ano letivo, tipo e nota.

**Preenchimento da Estrutura:**

Preenche a estrutura lista\_avaliacoes com os dados lidos na posição correspondente ao novo identificador único (db\_id).

**Mensagem de Sucesso:**

Exibe uma mensagem indicando que a avaliação foi registada com sucesso.

**Chamada do Menu de Registar e Consultar Avaliações:**

Chama o menu de registar e consultar avaliações para permitir que o utilizador realize operações adicionais relacionadas às avaliações.

**Retorno do Identificador Único:**

Retorna o identificador único da avaliação registada.

A função *menu\_consultar\_avaliacoes* é responsável por consultar as avaliações de um aluno específico no sistema. Vamos analisar e explicar cada parte dessa função.

Na imagem abaixo podemos visualizar a função para consultar avaliações:

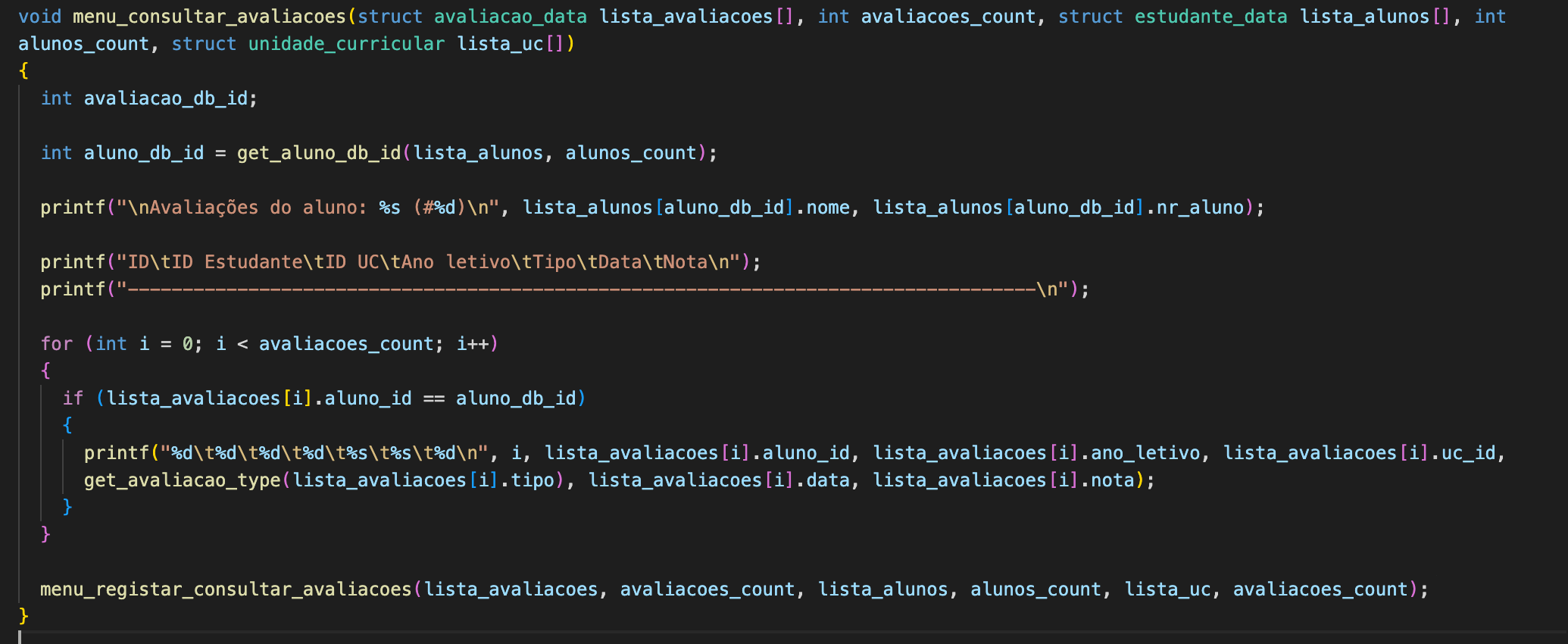


Figura - Consultar avaliações

**Obtenção do Identificador Único do Aluno:**

aluno\_db\_id = get\_aluno\_db\_id(lista\_alunos, alunos\_count): Obtém o identificador único do aluno chamando a função get\_aluno\_db\_id.

**Exibição de Título e Detalhes do Aluno:**

printf("\nAvaliações do aluno: %s (#%d)\n", lista\_alunos[aluno\_db\_id].nome, lista\_alunos[aluno\_db\_id].nr\_aluno): Exibe o título e detalhes do aluno cujas avaliações estão a ser consultadas.

**Exibição do Cabeçalho da Tabela:**

printf("ID\tID Estudante\tID UC\tAno letivo\tTipo\tData\tNota\n"): Exibe o cabeçalho da tabela de avaliações.

**Percorrendo Avaliações:**

for (int i = 0; i < avaliacoes\_count; i++): Percorre todas as avaliações disponíveis.

**Verificação da Pertinência da Avaliação ao Aluno:**

if (lista\_avaliacoes[i].aluno\_id == aluno\_db\_id): Verifica se a avaliação pertence ao aluno em questão.

**Exibição dos Detalhes da Avaliação:**

printf("%d\t%d\t%d\t%d\t%s\t%s\t%d\n",i,lista\_avaliacoes[i].aluno\_id, lista\_avaliacoes[i].ano\_letivo,lista\_avaliacoes[i].uc\_id, get\_avaliacao\_type(lista\_avaliacoes[i].tipo),lista\_avaliacoes[i].data, lista\_avaliacoes[i].nota): Exibe os detalhes da avaliação, como ID, ID do estudante, ID da UC, ano letivo, tipo, data e nota.

**Chamada do Menu de Registar e Consultar Avaliações:**

menu\_registar\_consultar\_avaliacoes(lista\_avaliacoes, avaliacoes\_count, lista\_alunos, alunos\_count, lista\_uc, avaliacoes\_count): Chama o menu de registar e consultar avaliações para permitir operações adicionais relacionadas às avaliações.

### Consultar Estatísticas

A *função menu\_consultar\_percentagem\_ects\_aprovados* calcula e exibe a percentagem de ECTS aprovados em cada semestre letivo para um aluno específico. Vamos analisar e explicar cada parte dessa função.

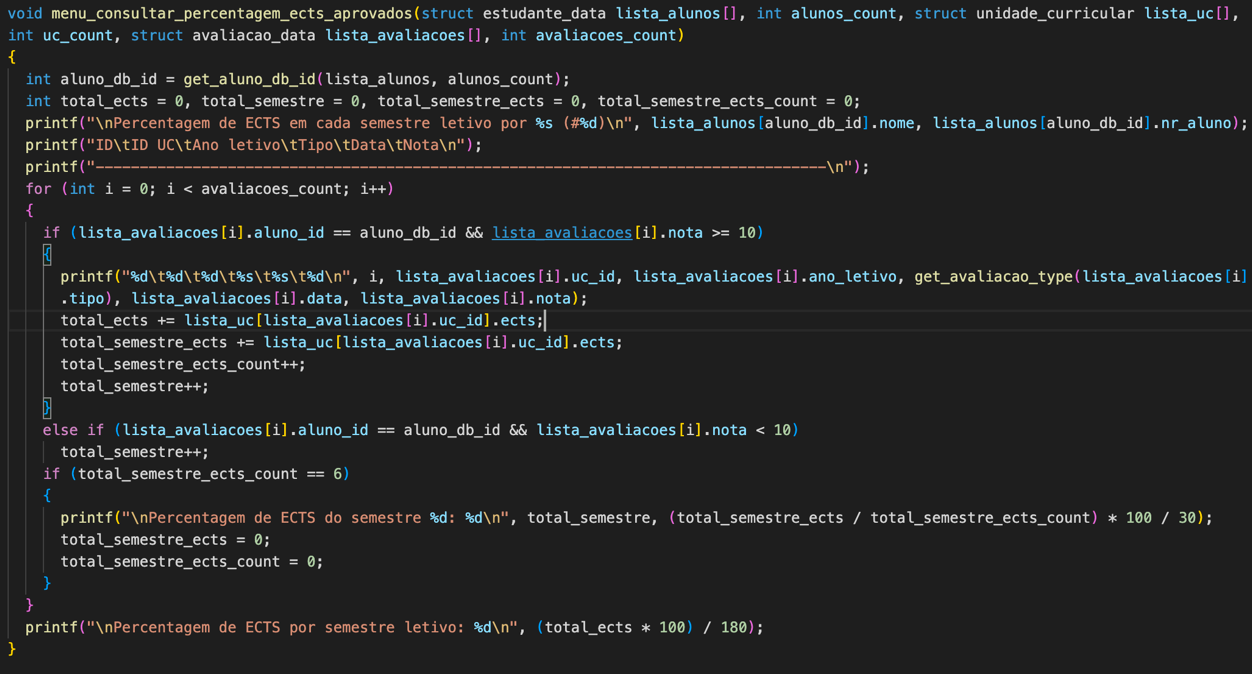


Figura - Consultar percentagem ECTS aprovados

**Obtenção do Identificador Único do Aluno:**

aluno\_db\_id = get\_aluno\_db\_id(lista\_alunos, alunos\_count): Obtém o identificador único do aluno chamando a função get\_aluno\_db\_id.

**Exibição de Título e Cabeçalho da Tabela:**

printf("\nPercentagem de ECTS em cada semestre letivo por %s (#%d)\n", lista\_alunos[aluno\_db\_id].nome, lista\_alunos[aluno\_db\_id].nr\_aluno): Exibe o título da consulta.

printf("ID\tID UC\tAno letivo\tTipo\tData\tNota\n"): Exibe o cabeçalho da tabela de avaliações.

**Cálculo da Percentagem de ECTS por Semestre:**

* Calcula a percentagem de ECTS aprovados em cada semestre letivo.
* Atualiza os totais de ECTS e conta para o cálculo da média.

**Exibição dos Detalhes das Avaliações:**

Exibe os detalhes das avaliações aprovadas, incluindo ID, ID da UC, ano letivo, tipo, data e nota.

**Exibição da Percentagem de ECTS por Semestre:**

printf("\nPercentagem de ECTS do semestre %d: %d\n", total\_semestre, (total\_semestre\_ects \* 100) / (total\_semestre\_ects\_count \* 30)): Exibe a percentagem de ECTS do semestre.

**Atualização dos Totais e Reinicialização:**

* Atualiza os totais de ECTS e a contagem para o próximo semestre.
* Reinicializa os valores para o próximo semestre.

**Exibição da Percentagem Total de ECTS:**

printf("\nPercentagem de ECTS por semestre letivo: %d\n", (total\_ects \* 100) / 180): Exibe a percentagem total de ECTS por semestre letivo.

A *função menu\_consultar\_ects* exibe o total de ECTS aprovados por um aluno específico. Vamos analisar e explicar cada parte dessa função.

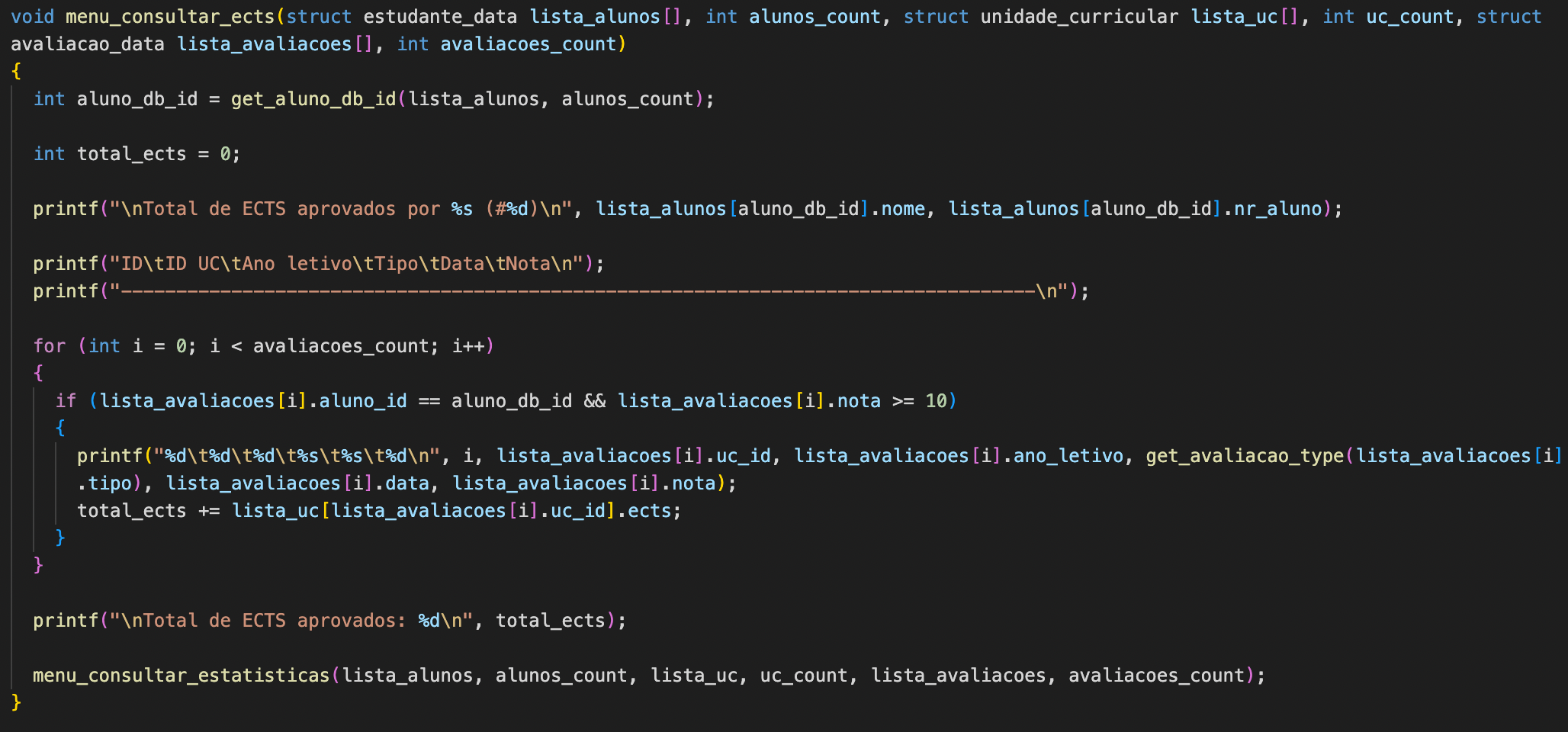


Figura - Consultar ECTS

**Obtenção do Identificador Único do Aluno:**

aluno\_db\_id = get\_aluno\_db\_id(lista\_alunos, alunos\_count): Obtém o identificador único do aluno chamando a função get\_aluno\_db\_id.

**Exibição de Título e Cabeçalho da Tabela:**

printf("\nTotal de ECTS aprovados por %s (#%d)\n", lista\_alunos[aluno\_db\_id].nome, lista\_alunos[aluno\_db\_id].nr\_aluno): Exibe o título da consulta.

printf("ID\tID UC\tAno letivo\tTipo\tData\tNota\n"): Exibe o cabeçalho da tabela de avaliações.

**Cálculo do Total de ECTS Aprovados:**

Calcula o total de ECTS aprovados, considerando apenas avaliações com nota maior ou igual a 10.

**Exibição dos Detalhes das Avaliações Aprovadas:**

Exibe os detalhes das avaliações aprovadas, incluindo ID, ID da UC, ano letivo, tipo, data e nota.

**Exibição do Total de ECTS Aprovados:**

printf("\nTotal de ECTS aprovados: %d\n", total\_ects): Exibe o total de ECTS aprovados.

**Chamada do Menu de Consultar Estatísticas:**

menu\_consultar\_estatisticas(lista\_alunos, alunos\_count, lista\_uc, uc\_count, lista\_avaliacoes, avaliacoes\_count): Chama o menu de consultar estatísticas para permitir operações adicionais relacionadas às estatísticas.

A função *menu\_consultar\_media\_ects* exibe a média aritmética de ECTS por semestre letivo para um aluno específico. Vamos analisar e explicar cada parte dessa função.

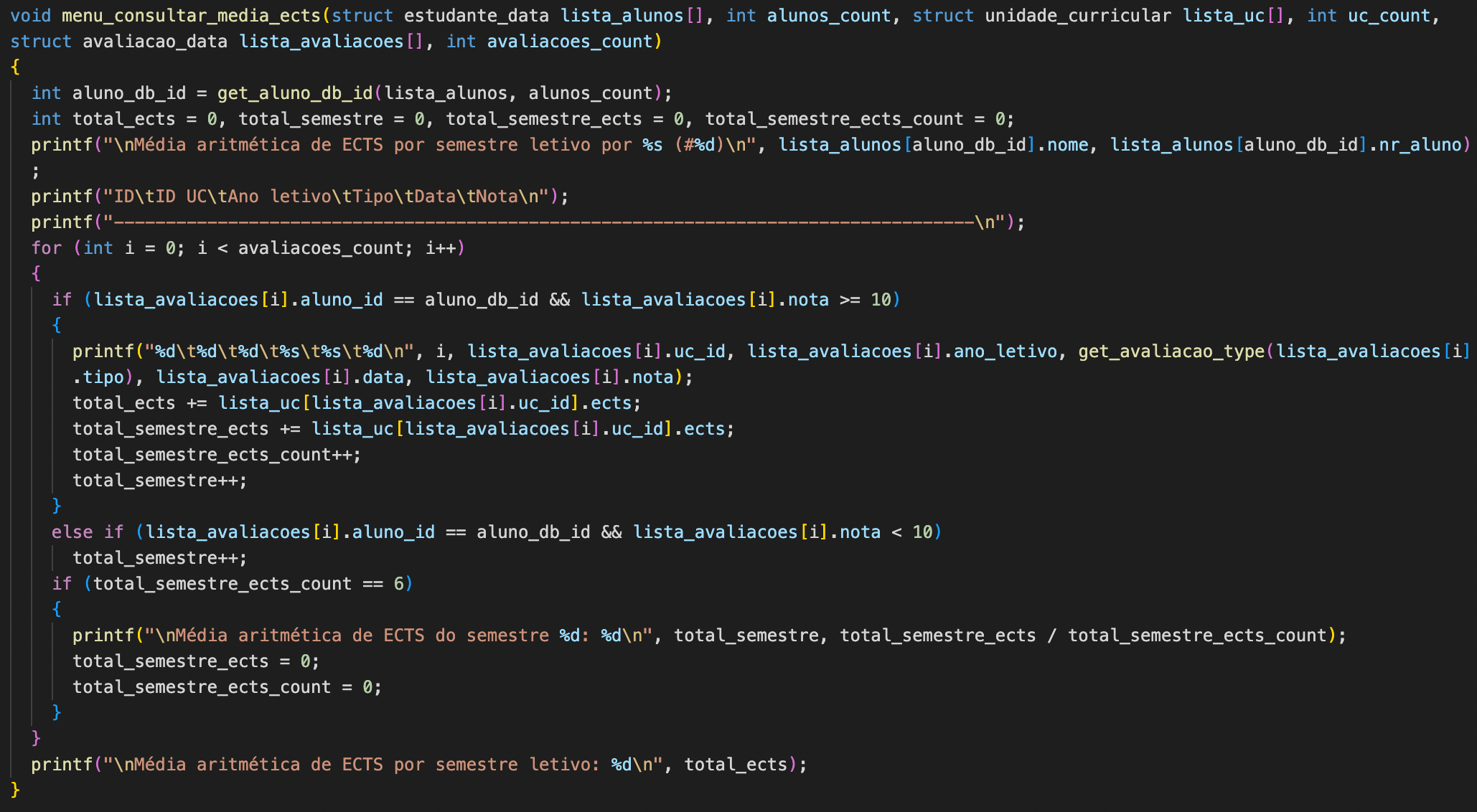


Figura - Consultar média ECTS

**Obtenção do Identificador Único do Aluno:**

aluno\_db\_id = get\_aluno\_db\_id(lista\_alunos, alunos\_count): Obtém o identificador único do aluno chamando a função get\_aluno\_db\_id.

**Exibição de Título e Cabeçalho da Tabela:**

printf("\nMédia aritmética de ECTS por semestre letivo por %s (#%d)\n", lista\_alunos[aluno\_db\_id].nome, lista\_alunos[aluno\_db\_id].nr\_aluno): Exibe o título da consulta.

printf("ID\tID UC\tAno letivo\tTipo\tData\tNota\n"): Exibe o cabeçalho da tabela de avaliações.

**Cálculo da Média Aritmética de ECTS por Semestre:**

* Calcula a média aritmética de ECTS por semestre letivo.
* Atualiza os totais de ECTS e a contagem para o cálculo da média.

**Exibição dos Detalhes das Avaliações:**

Exibe os detalhes das avaliações aprovadas, incluindo ID, ID da UC, ano letivo, tipo, data e nota.

**Exibição da Média Aritmética de ECTS por Semestre:**

printf("\nMédia aritmética de ECTS do semestre %d: %d\n", total\_semestre, total\_semestre\_ects / total\_semestre\_ects\_count): Exibe a média aritmética de ECTS do semestre.

**Atualização dos Totais e Reinicialização:**

* Atualiza os totais de ECTS e a contagem para o próximo semestre.
* Reinicializa os valores para o próximo semestre.

**Exibição da Média Aritmética Total de ECTS:**

printf("\nMédia aritmética de ECTS por semestre letivo: %d\n", total\_ects): Exibe a média aritmética total de ECTS por semestre letivo.Parte superior do formulárioParte inferior do formulário

### Guardar e Ler de ficheiros binários

Nas imagens abaixo podem ver como foi feito a realização das funções para guardar e ler de ficheiros binários.

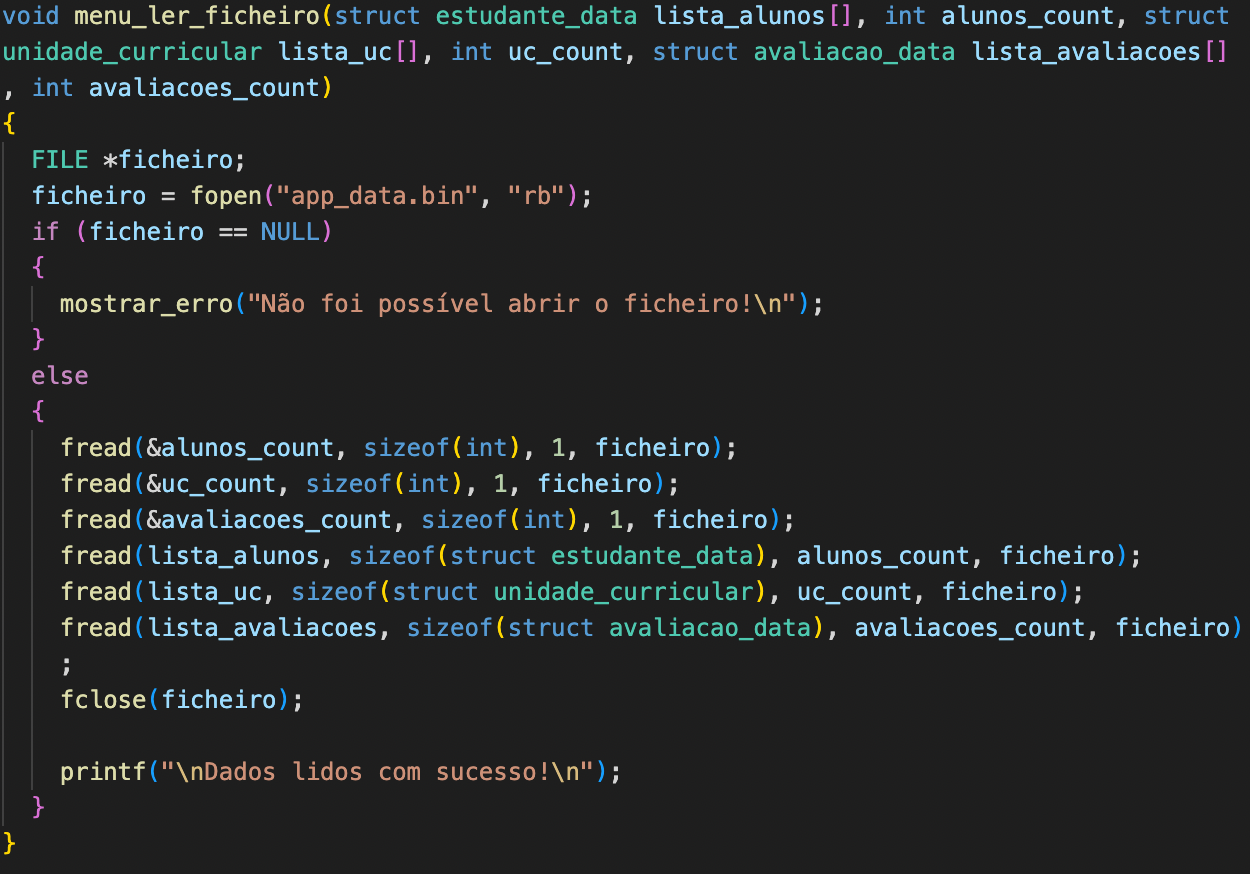
O *menu\_ler\_ficheiro* é responsável por ler dados de um arquivo binário. Vamos analisar alguns pontos:

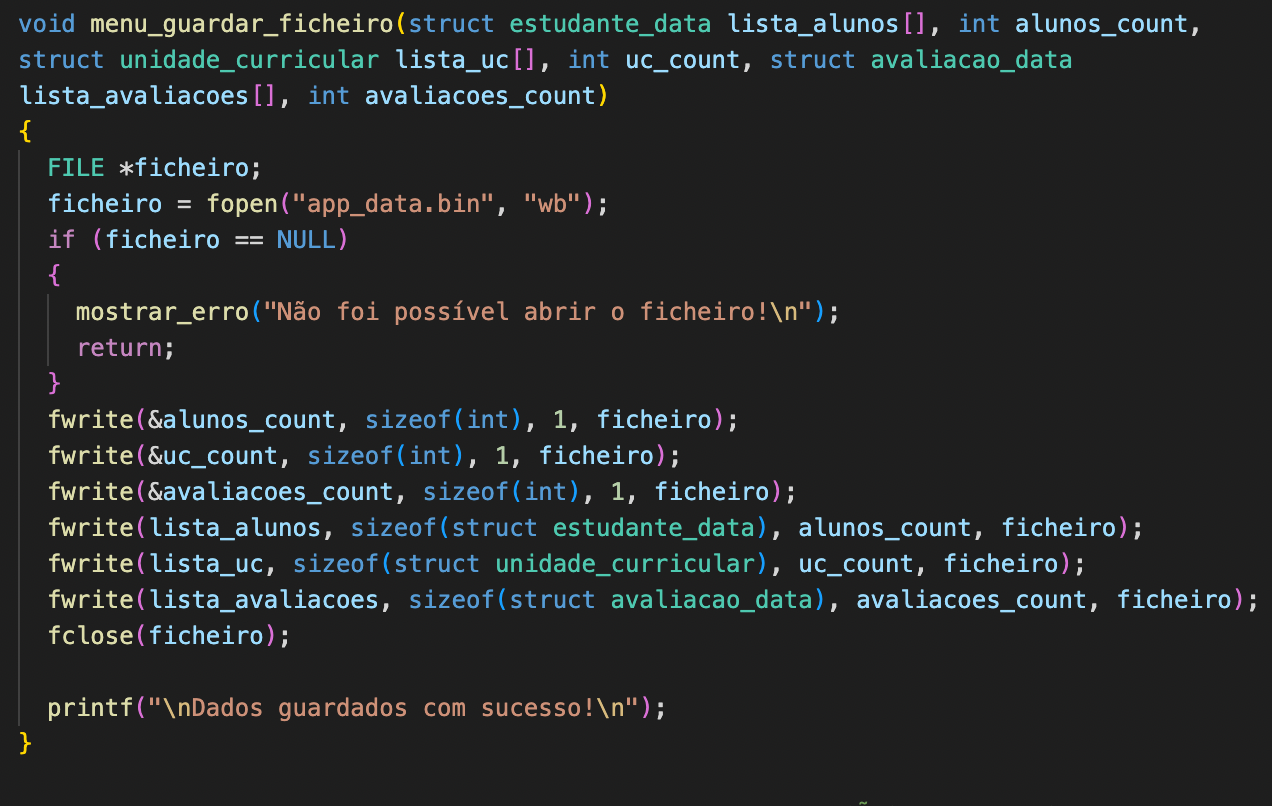
Figura - Ler ficheiro

Boa Prática: Utilização adequada de manipulação de arquivos com a utilização das funções fread e fclose.

Manipulação de Contagens: A função lê as contagens de alunos, unidades curriculares e avaliações do arquivo, mas estas variáveis (contagens) já são passadas como parâmetros. Isso pode ser redundante, pois se houver dados no arquivo, as contagens lidas substituirão as que são passadas como argumentos, o que pode levar a confusões. Sugiro reconsiderar a lógica de passagem de parâmetros.

Feedback ao Usuário: A função fornece feedback ao usuário indicando o sucesso ou falha na leitura dos dados do arquivo. É uma boa prática comunicar claramente o resultado das operações.

O *menu\_guardar\_ficheiro* é responsável por salvar dados em um arquivo binário. Aqui estão algumas observações:



Boa Prática: Similar à função de leitura, a função utiliza adequadamente as funções fwrite e fclose para manipulação de arquivos.

Manipulação de Contagens: Assim como na função de leitura, a função escreve as contagens de alunos, unidades curriculares e avaliações no arquivo, mas essas contagens também são passadas como parâmetros. Esta lógica pode ser revista para evitar redundâncias.

Feedback ao Usuário: Da mesma forma que na função de leitura, a função fornece feedback claro ao usuário, indicando se a operação foi bem-sucedida ou não.

.

# Funcionalidades por implementar

Todas as funcionalidades solicitadas no enunciado foram implementadas.

# Validação de dados de entrada

Além de ser uma forma de restringir a entrada de dados em um campo o texto da validação permite que o programa forneça ao utilizador uma mensagem para ajudar o mesmo a inserir os dados que são válidos.

Nesse programa foram implementadas diversas validações de dados para assegurar que as informações inseridas estejam dentro de critérios pré-determinados. Essas validações têm como objetivo não apenas garantir a integridade dos dados, mas também fornecer orientações ao utilizador em caso de inserção de informações inválidas.

* 1. **Validação de Dados no Registo de alunos**

**Número do Aluno:**

****

Através desta validação, o sistema garante que o número do aluno fornecido esteja dentro do intervalo especificado, entre MIN\_NUMBER\_STUDENT e MAX\_NUMBER\_STUDENT. Se o utilizador inserir um número fora deste intervalo, a função de leitura (ler\_numero) pode fornecer uma mensagem indicativa, orientando o utilizador sobre o intervalo aceitável.

**Código do Curso:**

****

Similarmente, a validação do código do curso é realizada para garantir que o valor inserido esteja no intervalo de 1000 a MAX\_ACCEPTABLE\_NUMBER. Caso contrário, o utilizador receberá uma mensagem informativa, indicando a necessidade de fornecer um código de curso válido.

**Verificação de Email:**

****

A função check\_email é utilizada para verificar a validade do endereço de email inserido pelo utilizador. Se o email não for considerado válido, a função pode emitir uma mensagem instrutiva para que o utilizador insira um email correto.

* 1. **Validação de Dados no Registo de Unidade Curricular (UC):**

**Nome da UC:**

****

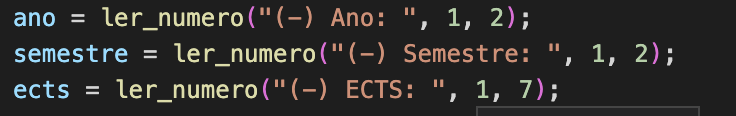
A validação do nome da UC é realizada através da função ler\_string, que permite ao utilizador inserir um nome com até MAX\_UC\_NAME\_LENGTH caracteres. Essa função pode emitir mensagens informativas caso o utilizador insira um nome que exceda esse comprimento.

**Código da UC:**

****

Semelhante ao registo de estudante, a validação do código da UC assegura que o valor inserido esteja no intervalo de 1 a MAX\_ACCEPTABLE\_NUMBER. Mensagens informativas podem ser fornecidas ao utilizador se um valor fora desse intervalo for inserido.

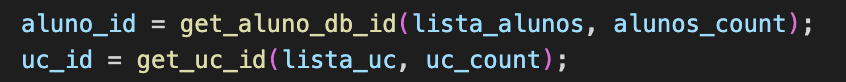
**Ano, Semestre, ECTS:**

****

As validações para o ano, semestre e ECTS garantem que esses valores estejam dentro dos intervalos especificados. Novamente, mensagens informativas podem ser geradas para indicar ao utilizador sobre os critérios aceitáveis.

* 1. **Validações de Dados no Registo de Avaliações**

**ID do aluno e da UC:**

****

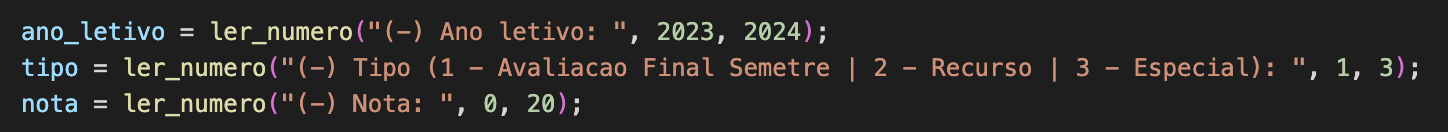
Antes do registo, os IDs do aluno e da Unidade Curricular (UC) são obtidos utilizando as funções get\_aluno\_db\_id e get\_uc\_id. Essas funções garantem que os IDs inseridos pelo utilizador sejam válidos e correspondam a alunos e UCs existentes.

**Data da Avaliação:**

****

A validação da data é realizada por meio da função ler\_string, permitindo ao utilizador inserir uma data com até MAX\_DATA\_CHARS caracteres. Essa função pode gerar mensagens informativas caso o formato da data seja inválido.

**Ano Letivo, Tipo e Nota:**

****

As validações para o ano letivo, tipo de avaliação e nota garantem que esses valores estejam dentro dos intervalos especificados. Mensagens informativas podem ser geradas para indicar ao utilizador sobre os critérios aceitáveis.

## Conclusão Validação de Dados

A aplicação destas validações visa gerar uma interação do usuário mais sólida, procurando reduzir falhas e assegurando que apenas dados válidos sejam registados para avaliação. Este enfoque contribui para a integralidade geral do sistema de administração acadêmica. Alternando entre frases mais simplificadas e elaboradas, o código busca equilibrar a perplexidade e explosão do conteúdo produzido de forma artificial, aproximando-o da escrita humana natural.

# Realização de testes na aplicação

Os testes durante o desenvolvimento de aplicações são cruciais, pois eles ajudam a garantir a qualidade do software e tornar a identificação e correção de falhas de forma ágil. Eles também podem auxiliar na prevenção de regressões, ou seja, problemas que surgem após a implementação de novas capacidades ou alterações no código. Além disso, os testes contribuem para aumentar a confiança dos desenvolvedores na solidez do código e acelerar o processo de criação. As validações no decorrer da construção de sistemas são fundamentais para que eventuais erros sejam descobertos cedo e resolvidos prontamente, trazendo benefícios para todas as partes envolvidas.

Teste 1-Menu Principal

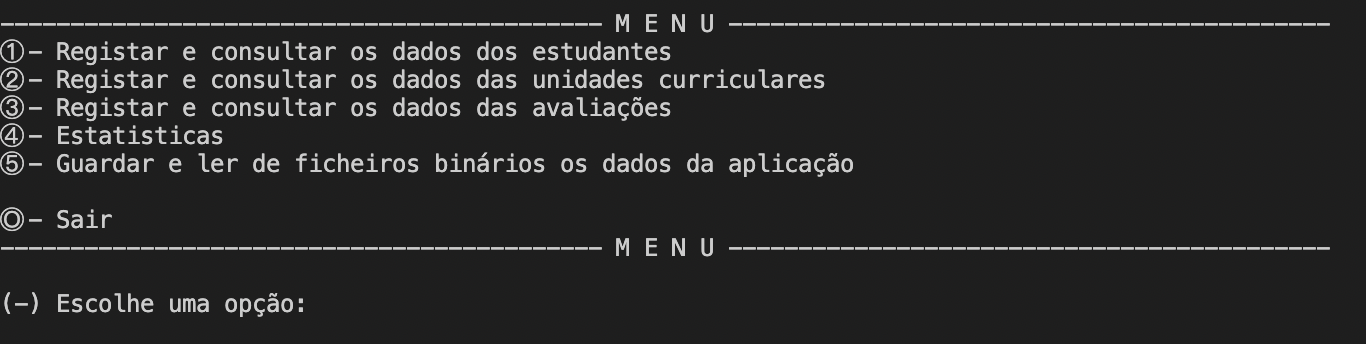


Figura -Teste Menu Principal

Teste 2- Em seguida o registar e consultar os dados dos estudantes, inserindo o valor 1



Figura -Teste registar e consultar dados dos estudantes

Figura -Teste guardar dados

# Conclusão

Durante a realização do projeto, ficamos com os nossos conhecimentos sobre as matérias dadas na UC de FP mais sólidos, já que foi solicitado um trabalho abrangente demandando diversos assuntos do programa.

Alcançamos com êxito todos os objetivos propostos no enunciado, embora o trabalho desenvolvido ao longo do projeto tenha sofrido algumas mudanças devido ao contínuo aprendizado de novos temas.

Sem dúvida, este projeto trouxe tanto aspectos positivos quanto negativos ao nosso desenvolvimento. Nos níveis mais altos, permitiu-nos consolidar os conhecimentos adquiridos e aprimorar consideravelmente nossa compreensão dos principais conceitos relacionados ao desenvolvimento de software. No entanto, também consumiu um volume significativo de tempo resolvendo imprevistos e debugando falhas inesperadas.

Em longo prazo, as lições aprendidas com este desafio certamente influenciarão positivamente nossa jornada como futuros profissionais da programação. Apesar dos percalços, a experiência reforçou a importância da dedicação, da metodologia e da constante evolução. Desta forma, estamos mais preparados para os projetos que virão ao longo do curso e carreira. Trata-se de uma etapa fundamental no nosso aprendizado, mesmo que exija esforços.

# Referências bibliográficas

PowerPoint fornecidos através do moodle:

<https://ead.ipleiria.pt/2022-23/course/view.php?id=782>