IPCA

Relatório de Projeto de PI e Laboratórios de Informárica

Relatório

Autores: David Faria 31517, João Pereira 31505, Rodrigo Pinheiro 31502
Docente: Oscár Ribeiro

5 de janeiro de 2025

Conteúdo

Capítulo 1

Introdução

O presente relatório foi elaborado no âmbito da Unidade Curricular de Laboratórios de Informática do curso de Licenciatura em Engenharia de Sistemas Informáticos, durante o ano letivo de 2024/2025. Este trabalho tem como objetivo consolidar competências práticas relacionadas com o desenvolvimento colaborativo de projetos utilizando a linguagem de programação C, adotando o paradigma de programação imperativa. Ao longo do desenvolvimento do projeto, são abordadas boas práticas de programação, incluindo a modularização de código, o uso de ferramentas de automação como o Makefile e a documentação utilizando ferramentas como o Doxygen. Adicionalmente, promove-se a utilização de sistemas de gestão de versões, com ênfase na colaboração em grupo por meio da plataforma GitHub. Outro aspeto central do trabalho é a utilização avançada da linha de comandos em ambientes Linux, promovendo uma interação eficiente com o sistema operativo para tarefas de desenvolvimento, compilação e execução do programa. Este relatório documenta todas as etapas do trabalho como, por exemplo: a definição do problema e sua análise, até a implementação, resultados obtidos e considerações finais.

- Definição do problema e sua análise;
- Implementação de funções;
- Resultados obtidos;
- Considerações finais;

Capítulo 2

Desenvolvimento

2.1 Conceito do programa

O programa desenvolvido apoia a gestão do Espaço Social de uma Instituição Pública, responsável por servir refeições aos seus funcionários. Usando a linguagem C, o sistema permite gerir dados dos funcionários, ementas semanais e escolhas de refeições, garantindo assim a sua eficiência.

Principais funcionalidades:

- Carregamento de Dados: Leitura de ficheiros com informações dos funcionários, ementas e escolhas de refeições.
- Gestão e Visualização: Listagem de refeições diárias, relatórios semanais para Recursos Humanos, e análise de calorias consumidas.
- Análise e Relatórios: Cálculo de médias de calorias e geração de tabelas detalhadas por funcionário.
- Flexibilidade: Suporte a ficheiros de texto e binários, com interface de linha de comandos para personalização.

O código é documentado com Doxygen, com um Makefile para compilação automatizada, e promove boas práticas de programação e trabalho em equipa.

Funcoes .h: Declaram funções e estruturas de dados, o que garante o armazenamento dos dados necessários.

Funcoes .c: Implementam e desenvolve as funções declaradas anteriormente.

Makefile: Automatiza o processo de compilação, especificando dependências e otimizando a geração do executável

Main.c: Coordena a execução do programa - processa argumentos da linha de comandos, chama funções e interage com o utilizador.

ementas.txt: Contém as ementas semanais, organizadas por dia, com pratos de peixe, carne, dieta e vegetariano, incluindo as calorias de cada prato. Exemplo: Segunda;18.11.2024;peixe grelhado;180;bife de vaca;330;frango grelhado;150;lasanha de vegetais;200;

escolhas.txt: Registra as escolhas dos funcionários, indicando o dia da semana, número de funcionário e o tipo de prato escolhido. Exemplo: Segunda;1;Peixe

funcionarios.txt: Armazena os dados dos funcionários, como número, nome, NIF e telefone. Exemplo: 1;Paulo Silva;179204181;963358792

2.1.1 Exemplos de funções implementadas

Segue abaixo exemplos de funções implementadas no código:

Listing 2.1: carregarFuncionarios

```
1  /**
2  @brief  Fun  o para carregar os dados dos funcion rios a
       partir de um ficheiro
3  @param nomeFicheiro  Nome do ficheiro
4  @param funcionario  Array de funcion rios
5  @return int  Retorna o total de funcion rios carregados*/
6  int carregarFuncionarios(char* nomeFicheiro, Funcionario
       funcionario[]) {
7      FILE *ficheiro = fopen(nomeFicheiro, "r");
8      if (ficheiro == NULL) {
9          printf("Erro ao abrir o ficheiro: %s\n", nomeFicheiro);
10         return 0;
11     }
12
13     int totalFuncionarios = 0;
14     while (totalFuncionarios < MAX_FUNCIONARIOS) {
15
16         // L  os dados do ficheiro e atribui aos campos da
               estrutura
17         if (fscanf(ficheiro, "%d;%99[^;];%14[^;];%14[^\n]\n",
18                    &funcionario[totalFuncionarios].numero,
19                    funcionario[totalFuncionarios].nome,
20                    funcionario[totalFuncionarios].nif,
21                    funcionario[totalFuncionarios].telefone) ==
                          4) {
22             totalFuncionarios++;
23         } else {
24             break; // Encerra a leitura caso ocorra erro ou
                   final do arquivo
25         }
26     }
27
28     fclose(ficheiro);
29
30     printf("Carregamento dos dados dos funcion rios foi
           realizado com sucesso!\n");
31     printf("Total de funcion rios carregados: %d\n",
           totalFuncionarios);
32     return totalFuncionarios;
33     }
```

Listing 2.2: carregarEmenta

```
/**

@brief  Fun  o para carregar a ementa semanal a partir de um
   ficheiro
@param nomeficheiro  Nome do ficheiro
@param ementas  Array de ementas
@return int  ]Retorna o total de ementas carregadas*/
int carregarEmenta(char* nomeficheiro, Ementa ementas[]) {
    FILE* ficheiro = fopen(nomeficheiro, "r");
    if (ficheiro == NULL) {
        printf("Erro ao abrir ficheiro de ementa\n");
        return 0;
    }

    int totalEmentas = 0;
    while (fscanf(ficheiro,
       "%9[^;];%11[^;];%99[^;];%d;%99[^;];%d;%99[^;];%d;%99[^;];%d\n",
                  ementas[totalEmentas].diaSemana,
                  ementas[totalEmentas].data,
                  ementas[totalEmentas].pratoPeixe,
                     &ementas[totalEmentas].caloriasPeixe,
                  ementas[totalEmentas].pratoCarne,
                     &ementas[totalEmentas].caloriasCarne,
                  ementas[totalEmentas].pratoDieta,
                     &ementas[totalEmentas].caloriasDieta,
                  ementas[totalEmentas].pratoVegetariano,
                     &ementas[totalEmentas].caloriasVegetariano)
                     == 10) {
            totalEmentas++;
        if (totalEmentas >= 5) {
            break;
        }
    }

    fclose(ficheiro);
    printf("Carregamento de ementa semanal foi realizado com
       sucesso!\n");
    return totalEmentas;
}
```

Listing 2.3: carregarEscolhas

```
/**
@brief  Fun  o para carregar as escolhas dos utentes a partir
   de um ficheiro
@param nomeFicheiro  Nome do ficheiro
@param escolhas  Array de escolhas
@return int  Retorna o total de escolhas carregadas*/
int carregarEscolhas(char* nomeFicheiro, Escolha escolhas[]) {
    FILE* ficheiro = fopen(nomeFicheiro, "r");
    if (ficheiro == NULL) {
```

```
9           printf("Erro ao abrir o ficheiro!");
10          return 0;
11      }
12
13      int totalEscolhas = 0;
14      while (fscanf(ficheiro, "%19[^;];%d;%19[^\n]\n",
15                    escolhas[totalEscolhas].diaSemana,
16                    &escolhas[totalEscolhas].numeroFuncionario,
17                    escolhas[totalEscolhas].tipoPrato) == 3) {
18          totalEscolhas++;
19          if (totalEscolhas >= MAX_ESCOLHAS) { // Limita ao
                n mero m ximo de escolhas
20              printf("Limite m ximo de refei  es atingido %d
                    \n", MAX_ESCOLHAS);
21              break;
22          }
23      }
24      fclose(ficheiro);
25
26      printf("Carregamento das escolhas realizado com sucesso!\n");
27      return totalEscolhas;
28  }
```

Listing 2.4: listarRefeicoesDia

```
1   /**
2   @brief  Fun  o para listar as refei  es de um dia espec fico
3   @param escolhas  Array de escolhas
4   @param totalEscolhas  Total de escolhas
5   @param funcionarios  Array de funcion rios
6   @param totalFuncionarios  Total de funcion rios
7   @param diaSemana  Dia da semana*/
8   void listarRefeicoesDia(Escolha* escolhas, int totalEscolhas,
        Funcionario* funcionarios, int totalFuncionarios, const char*
        diaSemana) {
9       printf("Refei  es para o dia %s:\n", diaSemana);
10      printf("-------------------------------------------------\n");
11      printf("| N  Funcion rio | Nome           | Prato
            |\n");
12      printf("-------------------------------------------------\n");
13
14      int encontrado = 0; // Flag para indicar se alguma
            refei  o foi encontrada
15
16      for (int i = 0; i < totalEscolhas; i++) {
17          if (strcmp(escolhas[i].diaSemana, diaSemana) == 0) {
18              // Encontra o nome do funcion rio correspondente
19              for (int j = 0; j < totalFuncionarios; j++) {
20                  if (funcionarios[j].numero ==
                        escolhas[i].numeroFuncionario) {
21                      printf(" |%15d|%-15s|%-15s| \n",
22                             funcionarios[j].numero,
```

```
23                              funcionarios[j].nome,
24                              escolhas[i].tipoPrato);
25                      encontrado = 1; // Marca que encontramos
                           pelo menos uma refei  o
26                      break; // Para de procurar o funcion rio
                           correspondente
27                  }
28              }
29          }
30      }
31
32      if (!encontrado) {
33          printf("Nenhuma refei  o foi registrada para o dia de
               %s.\n", diaSemana);
34      }
35  }
```

Listing 2.5: listarUtentesOrdemDecrescente

```
1  /**
2  @brief Fun  o para listar os utentes em ordem decrescente de
       n mero de funcion rio
3  @param escolhas Array de escolhas
4  @param totalEscolhas Total de Escolhas Possivel
5  @param funcionarios  Array de funcion rios
6  @param totalFuncionarios  Total de Funcion rios*/
7  void listarUtentesOrdemDecrescente(Escolha* escolhas, int
       totalEscolhas, Funcionario* funcionarios, int
       totalFuncionarios) {
8      // Arrays para armazenar refei  es e despesas por
           funcion rio
9      int refeicoes[totalFuncionarios];
10     float despesas[totalFuncionarios];
11
12     // Inicializa os contadores
13     for (int i = 0; i < totalFuncionarios; i++) {
14         refeicoes[i] = 0;
15         despesas[i] = 0.0f;
16     }
17
18     // Calcula o n mero de refei  es e a despesa total para
           cada funcion rio
19     for (int i = 0; i < totalEscolhas; i++) {
20         for (int j = 0; j < totalFuncionarios; j++) {
21             if (escolhas[i].numeroFuncionario ==
                   funcionarios[j].numero) {
22                 refeicoes[j]++;
23                 despesas[j] += 6.00;
24             }
25         }
26     }
27
```

```
28      // Ordena os funcion rios em ordem decrescente de n mero
            de funcion rio
29      for (int i = 0; i < totalFuncionarios - 1; i++) {
30          for (int j = i + 1; j < totalFuncionarios; j++) {
31              if (funcionarios[i].numero< funcionarios[j].numero) {
32                  // Troca os funcion rios
33                  Funcionario tempFuncionario = funcionarios[i];
34                  funcionarios[i] = funcionarios[j];
35                  funcionarios[j] = tempFuncionario;
36
37                  // Troca os dados correspondentes
38                  int tempRefeicoes = refeicoes[i];
39                  refeicoes[i] = refeicoes[j];
40                  refeicoes[j] = tempRefeicoes;
41
42                  float tempDespesas = despesas[i];
43                  despesas[i] = despesas[j];
44                  despesas[j] = tempDespesas;
45              }
46          }
47      }
48
49      // Exibe os dados ordenados
50      printf("Relat rio de Utentes - Recursos Humanos\n");
51      printf("-------------------------------------------------------------\n"
52      printf("| N   Funcion rio | Nome            | Refei   es |
            Total (   ) |\n");
53      printf("-------------------------------------------------------------\n"
54      for (int i = 0; i < totalFuncionarios; i++) {
55          printf("| %-15d | %-15s | %-10d | %-10.2f |\n",
56                 funcionarios[i].numero,
57                 funcionarios[i].nome,
58                 refeicoes[i],
59                 despesas[i]);
60      }
61  }
```

Listing 2.6: listarRefeicoesUtente

```
1   /**
2   @brief  Fun   o para listar as refei   es e calorias de um
        utente durante um per odo
3   @param escolhas  Array de escolhas
4   @param totalEscolhas  Total de escolhas
5   @param ementas  Array de ementas
6   @param totalFuncionarios  Total de funcion rios
7   @param totalEmentas  Total de ementas*/
8   void listarRefeicoesUtente(Escolha* escolhas, int totalEscolhas,
        Ementa* ementas, int totalFuncionarios, int totalEmentas) {
9       int numeroFuncionario;
10      printf("Escreva o n mero do funcion rio: ");
11      scanf("%d", &numeroFuncionario);
```

```
12
13      // Verifica se o n mero do funcion rio    v lido antes de
            pedir os dias
14      if (numeroFuncionario <= 0  numeroFuncionario >
            totalFuncionarios) {
15          printf("Erro: N mero de funcion rio %d n o
                existe.\n", numeroFuncionario);
16          return;  // Sai da fun   o e retorna ao menu
17      }
18
19      char DiaInicio[20], DiaFim[20];
20      printf("Escreva o dia de in cio (ex: Segunda): ");
21      scanf("%s", DiaInicio);
22      printf("Escreva o dia de fim (ex: Sexta): ");
23      scanf("%s", DiaFim);
24
25      int numeroInicio = diaSemanaParaNumero(DiaInicio);
26      int numeroFim = diaSemanaParaNumero(DiaFim);
27
28
29      if (numeroInicio == -1  numeroFim == -1) {
30          printf("Erro: Dia de in cio ou fim inv lido.\n");
31          return;
32      }
33
34      printf("Refei  es e calorias do funcion rio %d durante o
            per odo de %s a %s:\n", numeroFuncionario, DiaInicio,
            DiaFim);
35      printf("-------------------------------------------------------\n");
36      printf("|  Dia da Semana  |   Tipo de Prato    |   Calorias
             |\n");
37      printf("-------------------------------------------------------\n");
38
39      int numeroRefeicoes = 0;
40      int totalCalorias = 0;
41      // Passa pelas escolhas e buscar correspond ncias na ementa
42      for (int i = 0; i < totalEscolhas; i++) {
43          if (escolhas[i].numeroFuncionario == numeroFuncionario) {
44              for (int j = 0; j < totalEmentas; j++) {
45                  if (strcmp(escolhas[i].diaSemana,
                        ementas[j].diaSemana) == 0) {
46                      int indiceAtual =
                            diaSemanaParaNumero(ementas[j].diaSemana);
47
48                      if (estaNoIntervalo(indiceAtual,
                            numeroInicio, numeroFim)) {
49                          // Determinar as calorias do prato
                                espec fico
50                          int calorias = 0;
51
52                          if (strcmp(escolhas[i].tipoPrato,
```

```
                            "Peixe") == 0) {
53                              calorias = ementas[j].caloriasPeixe;
54                          } else if (strcmp(escolhas[i].tipoPrato,
                            "Carne") == 0) {
55                              calorias = ementas[j].caloriasCarne;
56                          } else if (strcmp(escolhas[i].tipoPrato,
                            "Dieta") == 0) {
57                              calorias = ementas[j].caloriasDieta;
58                          } else if (strcmp(escolhas[i].tipoPrato,
                            "Vegetariano") == 0) {
59                              calorias =
                                    ementas[j].caloriasVegetariano;
60                          }
61
62                          // Atualizar valores e exibir o prato
                            espec fico
63                          totalCalorias += calorias; // Acumular
                            calorias
64                          printf("| %16s | %15s | %5d |\n",
                            ementas[j].diaSemana,
                            escolhas[i].tipoPrato, calorias);
65                          numeroRefeicoes++;
66                      }
67                  }
68              }
69          }
70      }
71
72      printf("-------------------------------------------------------\n");
73      if (numeroRefeicoes > 0) {
74          printf("Total de refei  es: %d\n", numeroRefeicoes);
75          printf("Total de calorias consumidas no per odo: %d\n",
                totalCalorias);
76      } else {
77          printf("Nenhuma refei  o registrada para o
                funcion rio %d durante o per odo de %s a %s.\n",
                numeroFuncionario, DiaInicio, DiaFim);
78      }
79  }
```

Listing 2.7: calcularMediaCaloriasEspaco

```
1   /**
2   @brief  Fun  o para calcular a m dia de calorias consumidas
        no per odo
3   @param ementas  Array de ementas
4   @param totalEmentas  Total de ementas
5   @param escolhas  Array de escolhas
6   @param totalEscolhas  Total de escolhas
7   @param diainicio  Dia de in cio
8   @param diafim  Dia de fim*/
9   // Fun  o para calcular a m dia de calorias consumidas no
```

```
    per odo
10 void calcularMediaCaloriasEspaco(Ementa* ementas, int
    totalEmentas, Escolha* escolhas, int totalEscolhas, char*
    diainicio, char* diafim) {
11     // Converte os dias de in cio e fim em n meros
12     int numeroDiaInicio = diaSemanaParaNumero(diainicio);
13     int numeroDiaFim = diaSemanaParaNumero(diafim);
14
15     if (numeroDiaInicio == -1 || numeroDiaFim == -1) {
16         printf("Erro: Dia de in cio ou fim inv lido.\n");
17         return;
18     }
19     // Inicializa as vari veis para somar as calorias por dia
20     int caloriasDia[5] = {0};  // 0: Segunda, 1: Ter a , 2:
       Quarta, 3: Quinta, 4: Sexta
21     int numeroRefeicoesPorDia[5] = {0};
22     int CaloriasPrato = 0;
23
24     // Percorre todas as escolhas de refei   o
25     for (int i = 0; i < totalEscolhas; i++) {
26         // Obt m o n mero do dia da escolha
27         int numeroDiaEscolha =
           diaSemanaParaNumero(escolhas[i].diaSemana);
28
29         // Verifica se o dia da escolha est  dentro do
           intervalo de dias
30         if (numeroDiaEscolha >= numeroDiaInicio &&
           numeroDiaEscolha <= numeroDiaFim) {
31             // Percorre as ementas para verificar a
               correspond ncia dos pratos
32             for (int y = 0; y < totalEmentas; y++) {
33                 if (numeroDiaEscolha ==
                   diaSemanaParaNumero(ementas[y].diaSemana)) {
34                     // Verifica o tipo de prato e calcula as
                       calorias
35                     if (strcmp(escolhas[i].tipoPrato, "Peixe")
                       == 0) {
36                         CaloriasPrato = ementas[y].caloriasPeixe;
37                     } else if (strcmp(escolhas[i].tipoPrato,
                       "Carne") == 0) {
38                         CaloriasPrato = ementas[y].caloriasCarne;
39                     } else if (strcmp(escolhas[i].tipoPrato,
                       "Dieta") == 0) {
40                         CaloriasPrato = ementas[y].caloriasDieta;
41                     } else if (strcmp(escolhas[i].tipoPrato,
                       "Vegetariano") == 0) {
42                         CaloriasPrato =
                               ementas[y].caloriasVegetariano;
43                     }
44
45                     // Adiciona as calorias do prato escolhido
```

```
                        ao total de calorias do dia correspondente
46                      caloriasDia[numeroDiaEscolha] +=
                            CaloriasPrato;
47                      numeroRefeicoesPorDia[numeroDiaEscolha]++;
48                      break; // Interrompe o loop de ementas, pois
                            encontramos o prato para o dia
49                  }
50              }
51          }
52      }
53      / Exibe as m dias de calorias consumidas por refei  o
            para cada dia da semana
54      printf("M dia de calorias consumidas por refei  o de cada
            dia da semana no per odo de %s a %s:\n", diainicio,
            diafim);
55      printf("-------------------------------------------------------\n");
56
57      // Array com os dias da semana
58      const char* diasSemana[] = {"Segunda", "Ter a", "Quarta",
            "Quinta", "Sexta"};
59
60      for (int k = 0; k < 5; k++) {
61          if (numeroRefeicoesPorDia[k] > 0) {
62              float media = (float)caloriasDia[k] /
                    numeroRefeicoesPorDia[k];
63              printf("%s:  %d refei  es , m dia: %.2f calorias
                    por refei  o.\n",
64                   diasSemana[k], numeroRefeicoesPorDia[k], media);
65          }
66      }
67  }
```

Listing 2.8: gerarTabelaEmentaUtente

```
1   /**
2   @brief  Fun  o para gerar uma tabela de ementa para um utente
3   @param escolhas  Array de escolhas
4   @param totalEscolhas  Total de escolhas
5   @param ementas  Array de ementas
6   @param totalEmentas  Total de ementas
7   @param numeroFuncionario  N mero do funcion rio*/
8   void gerarTabelaEmentaUtente(Escolha* escolhas, int
        totalEscolhas, Ementa* ementas, int totalEmentas, int
        numeroFuncionario) {
9       printf("| Dia Semana | Prato Escolhido | Calorias |\n");
10      printf("=============================================\n");
11      // Percorre todos os dias da semana para verificar se o
            funcion rio fez escolha
12      for (int i = 0; i < totalEmentas; i++) {
13          char* refeicaoEscolhida = 0;
14          int calorias = 0;
15
```

```
16          // Verifica se o funcion rio fez uma escolha para
                aquele dia
17          for (int j = 0; j < totalEscolhas; j++) {
18              if (escolhas[j].numeroFuncionario ==
                    numeroFuncionario &&
19                  strcmp(escolhas[j].diaSemana,
                        ementas[i].diaSemana) == 0) {
20
21
22                  // Atribui as calorias conforme o prato escolhido
23                  if (strcmp(escolhas[j].tipoPrato, "Peixe") == 0)
                        {
24                      calorias = ementas[i].caloriasPeixe;
25                  } else if (strcmp(escolhas[j].tipoPrato,
                        "Carne") == 0) {
26                      calorias = ementas[i].caloriasCarne;
27                  } else if (strcmp(escolhas[j].tipoPrato,
                        "Dieta") == 0) {
28                      calorias = ementas[i].caloriasDieta;
29                  } else if (strcmp(escolhas[j].tipoPrato,
                        "Vegetariano") == 0) {
30                      calorias = ementas[i].caloriasVegetariano;
31                  }
32
33                  // Imprime os dados na tabela
34                  printf("| %15s | %15s | %-5d |\n",
35                         ementas[i].diaSemana,
                               escolhas[j].tipoPrato , calorias);
36                  break;
37              }
38          }
39      }
40  }
```

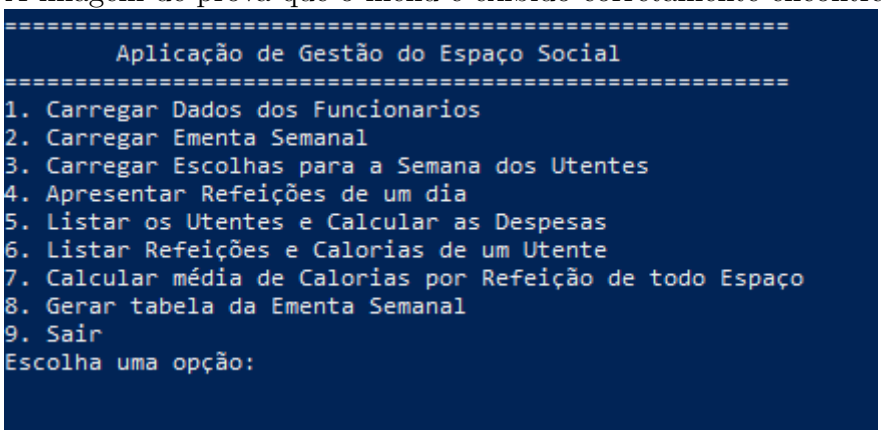
Capítulo 3

Resultados

A funcionalidade de mostrar o menu foi implementada corretamente, conforme esperado. Ao executar o programa, o menu é exibido de forma clara e interativa, permitindo ao utilizador selecionar diversas opções de acordo com o que é solicitado no enunciado do problema. As funções presentes no menu contemplam todas as funcionalidades exigidas pelo enunciado, incluindo a manipulação e processamento de dados, bem como a interação com o sistema de forma eficiente.

As opções do menu foram testadas com entradas válidas e inválidas, e o sistema comportou-se corretamente em todas as situações. O menu também apresenta mensagens informativas e de erro apropriadas, o que garante uma experiência para o utilizador intuitiva.

A imagem de prova que o menu é exibido corretamente encontra-se abaixo:



Após testar todas as operações presentes no menu (1-9) os resultados apresentam-se na tabela abaixo:

Tabela 3.1: Resultados obtidos nos testes.

Teste	Resultado Esperado	Resultado Obtido
Teste 1	Sucesso	Sucesso
Teste 2	Sucesso	Sucesso
Teste 3	Sucesso	Sucesso
Teste 4	Sucesso	Sucesso
Teste 5	Sucesso	Sucesso
Teste 6	Sucesso	Sucesso
Teste 7	Sucesso	Sucesso
Teste 8	Sucesso	Sucesso
Teste 9	Sucesso	Sucesso

Capítulo 4

Conclusão

O desenvolvimento deste trabalho permitiu consolidar competências em programação imperativa, gestão de dados e boas práticas de desenvolvimento de trabalho de grupo. A solução proposta responde de forma eficiente às necessidades de gestão do Espaço Social, o que permite o processamento de dados de funcionários, ementas e escolhas, e gerar relatórios úteis para a administração. A estrutura do programa, com separação de responsabilidades em diferentes ficheiros e funções, facilita a manutenção e escalabilidade da aplicação. A utilização de técnicas como leitura e validação de ficheiros, organização de dados e cálculo de métricas reforçou a compreensão dos conceitos aprendidos ao longo da unidade curricular. Por fim, o uso de ferramentas como Makefile, Git e Doxygen assegurou um desenvolvimento estruturado e documentado, alinhado às boas práticas de engenharia de software. Este trabalho destacou a importância da colaboração em equipa, da organização no desenvolvimento e da clareza na documentação para garantir o sucesso de projetos de programação.