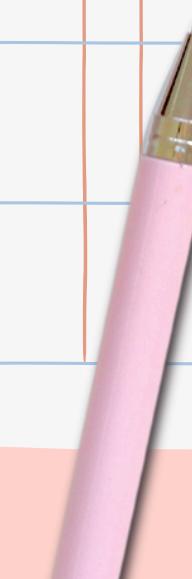


ORGANIZANDO AS TURMAS DO PROFESSOR CHARLES



ALUNAS

BEATRIZ DI PALMA
CARVALHO
RA: 10439477

LUIZA MARINHO DE
MESQUITA
RA: 10438045

MARINA CANTARELLI
BARROCA
RA: 10740412

```
c main.cpp > ...
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #define MAX 1000
5 // Define o número máximo de alunos que podem ser armazenados no nosso array
6
7 struct estudantes{    // armazena informações dos estudantes
8     int semestre;
9     char turma;
10    char periodo;
11    char nome[200];
12    char disciplina[200];
13    float media;
14 };
15
16 // Função para carregar dados de um arquivo CSV para o array dos estudantes
17 int carregar_csv(const char *nomeArquivo, estudantes alunos[]) {
18     FILE *arquivo = fopen(nomeArquivo, "r");    // Abre um arquivo de leitura
19
20     if (!arquivo) {    // Verifica se o arquivo foi aberto sem erros
21         printf("Erro ao abrir o arquivo\n");
22         return 0;
23     }
24
25     int num_alunos = 0;
26     char linha[MAX];    // Buffer que armazena cada linha por linha do arquivo
27
28     while (fgets(linha, sizeof(linha), arquivo)) {
29         estudantes aluno;
30
31         // Lê os valores do CSV e o armazena na estrutura
32         if (sscanf(linha, "%d,%c,%c,%199[^,],%199[^,],%f", //le um int, caractere, 199 caracteres ate virgula e float
33                    &aluno.semestre, &aluno.turma, &aluno.periodo,
34                    aluno.nome, aluno.disciplina, &aluno.media) == 6) {
35             alunos[num_alunos++] = aluno;    // Adiciona o aluno ao array
36         }
37     }
38     fclose(arquivo);    // Fecha o arquivo
39     return num_alunos; // Retorna o número de alunos lidos
40 }
41
```



```
42 // Função que salva os dados dos alunos em um arquivo CSV com o que foi pedido
43 void salvar_csv(const char *nomeArquivo, estudantes alunos[], int num_alunos, int opcao) {
44     FILE *arquivo = fopen(nomeArquivo, "w");      // Abre um arquivo de escrita
45
46     if (!arquivo) {
47         printf("Erro ao criar o arquivo de saída: %s\n", nomeArquivo);
48         exit(1);
49     }
50
51     // Escreve os dados dos alunos no arquivo CSV dependendo da opção que foi selecionada
52     for (int i = 0; i < num_alunos; i++) {
53         switch (opcao) {
54             case 1: // Ordenar por nome
55                 fprintf(arquivo, "%s,%d,%c,%c,%s,.2f\n", alunos[i].nome, alunos[i].semestre, alunos[i].turma, alunos[i].periodo, alunos[i].disciplina, alunos[i].media);
56                 break;
57
58             case 2: // Ordenar por semestre
59                 fprintf(arquivo, "%s,%d,%c,%c,%s,.2f\n", alunos[i].nome, alunos[i].semestre, alunos[i].turma, alunos[i].periodo, alunos[i].disciplina, alunos[i].media);
60                 break;
61
62             case 3: // Ordenar por semestre, turma, período, disciplina e nome
63                 fprintf(arquivo, "%d,%c,%c,%s,%s\n", alunos[i].semestre, alunos[i].turma, alunos[i].periodo, alunos[i].nome, alunos[i].disciplina);
64                 break;
65
66             case 4: // Ordenar por disciplina e média final
67                 fprintf(arquivo, "%s,.2f,%s\n", alunos[i].disciplina, alunos[i].media, alunos[i].nome);
68                 break;
69
70             case 5: // Ordenar por período, semestre, turma, disciplina e nome
71                 fprintf(arquivo, "%c,%d,%c,%s,%s\n", alunos[i].periodo, alunos[i].semestre, alunos[i].turma, alunos[i].nome, alunos[i].disciplina);
72                 break;
73
74             default: // else..
75                 printf("Opção inválida para salvar!\n");
76                 break;
77         }
78     }
79     fclose(arquivo);
80     printf("Arquivo '%s' gerado com sucesso!\n", nomeArquivo);
81 }
82 }
```

```
83 // Função que troca dois elementos no array para que auxilia a ordenação (bubble sort)
84 void troca(estudantes *a, estudantes *b) {
85     estudantes temp = *a;
86     *a = *b;
87     *b = temp;
88 }
89
90 // Implementação do algoritmo Bubble Sort para que ocorra a ordenação dos alunos
91 void bubble_sort(estudantes alunos[], int num_alunos, int (*compara)(estudantes, estudantes)) {
92     for (int i = 0; i < num_alunos - 1; i++) {
93         for (int j = 0; j < num_alunos - i - 1; j++) {
94             if (compara(alunos[j], alunos[j + 1]) > 0) {
95                 troca(&alunos[j], &alunos[j + 1]);
96             }
97         }
98     }
99 }
100
101 // Funções de comparação para os diferentes critérios de ordenação
102 int compara_nome(estudantes a, estudantes b)
103 {
104     for (int i = 0; a.nome[i] != '\0' && b.nome[i] != '\0'; i++) {
105         if (a.nome[i] != b.nome[i]) {
106             return a.nome[i] - b.nome[i];
107         }
108     }
109     return 0;
110 }
111
112 int compara_semestre(estudantes a, estudantes b)
113 {
114     return a.semestre - b.semestre;
115 }
116
```

```
117 int compara_item3(estudantes a, estudantes b) {
118     if (a.semestre != b.semestre) return a.semestre - b.semestre;
119     if (a.turma != b.turma) return a.turma - b.turma;
120     if (a.periodo != b.periodo) return a.periodo - b.periodo;
121     if (strcmp(a.disciplina, b.disciplina) != 0) return strcmp(a.disciplina, b.disciplina);
122     return strcmp(a.nome, b.nome);
123 }
124
125 // Função de comparação para ordenação DECRESCENTE por media final e disciplina
126 int compara_disciplina_media(estudantes a, estudantes b) {
127     int compara = strcmp(b.disciplina, a.disciplina);
128
129     if (compara == 0) {
130         if (a.media < b.media) return 1;
131         if (a.media > b.media) return -1;
132         return 0;
133     }
134     return compara;
135 }
136
137 int compara_item5(estudantes a, estudantes b) {
138     if (a.periodo != b.periodo) return a.periodo - b.periodo;
139     if (a.semestre != b.semestre) return a.semestre - b.semestre;
140     if (a.turma != b.turma) return a.turma - b.turma;
141     int disciplina_cmp = strcmp(a.disciplina, b.disciplina);
142     if (disciplina_cmp != 0) return disciplina_cmp;
143     return strcmp(a.nome, b.nome);
144 }
145
```

```
146 // Função que exibe o menu e executa a opção escolhida pelo usuário
147 void menu(estudantes alunos[], int num_alunos){
148     int opcao;
149     printf("\nMENU: \n");
150     printf("1. Ordenar por nome\n");
151     printf("2. Ordenar por semestre\n");
152     printf("3. Ordenar por semestre, turma, período, disciplina e nome\n");
153     printf("4. Ordenar por disciplina e média final (decrescente)\n");
154     printf("5. Ordenar por período, semestre, turma, disciplina e nome\n");
155     printf("Opção desejada: ");
156     (void)scanf("%d", &opcao);
157     // Armazena a opção em uma variável e realiza os seus cases respectivos
158
159     switch (opcao) {
160         case 1:
161             bubble_sort(alunos, num_alunos, compara_nome);
162             salvar_csv("saída_nomes.csv", alunos, num_alunos, opcao);
163             break;
164
165         case 2:
166             bubble_sort(alunos, num_alunos, compara_semestre);
167             salvar_csv("saída_semestre.csv", alunos, num_alunos, opcao);
168             break;
169
170         case 3:
171             bubble_sort(alunos, num_alunos, compara_item3);
172             salvar_csv("saída_item_tres.csv", alunos, num_alunos, opcao);
173             break;
174
175         case 4:
176             bubble_sort(alunos, num_alunos, compara_disciplina_media);
177             salvar_csv("saída_disciplinaMedia.csv", alunos, num_alunos, opcao);
178             break;
179
180         case 5:
181             bubble_sort(alunos, num_alunos, compara_item5);
182             salvar_csv("saída_item5.csv", alunos, num_alunos, opcao);
183             break;
184
185         default:
186             printf("Opção inválida! Tente novamente...\n");
187     }
188 }
```



```
189
190 int main() {
191     estudantes alunos[MAX];           // Declara um array para armazenar os alunos
192     int num_alunos = carregar_csv("entrada.csv", alunos); // Carrega o CSV e finaliza o programa
193     menu(alunos, num_alunos);
194     return 0;
195 }
```

OBRIGADA!

