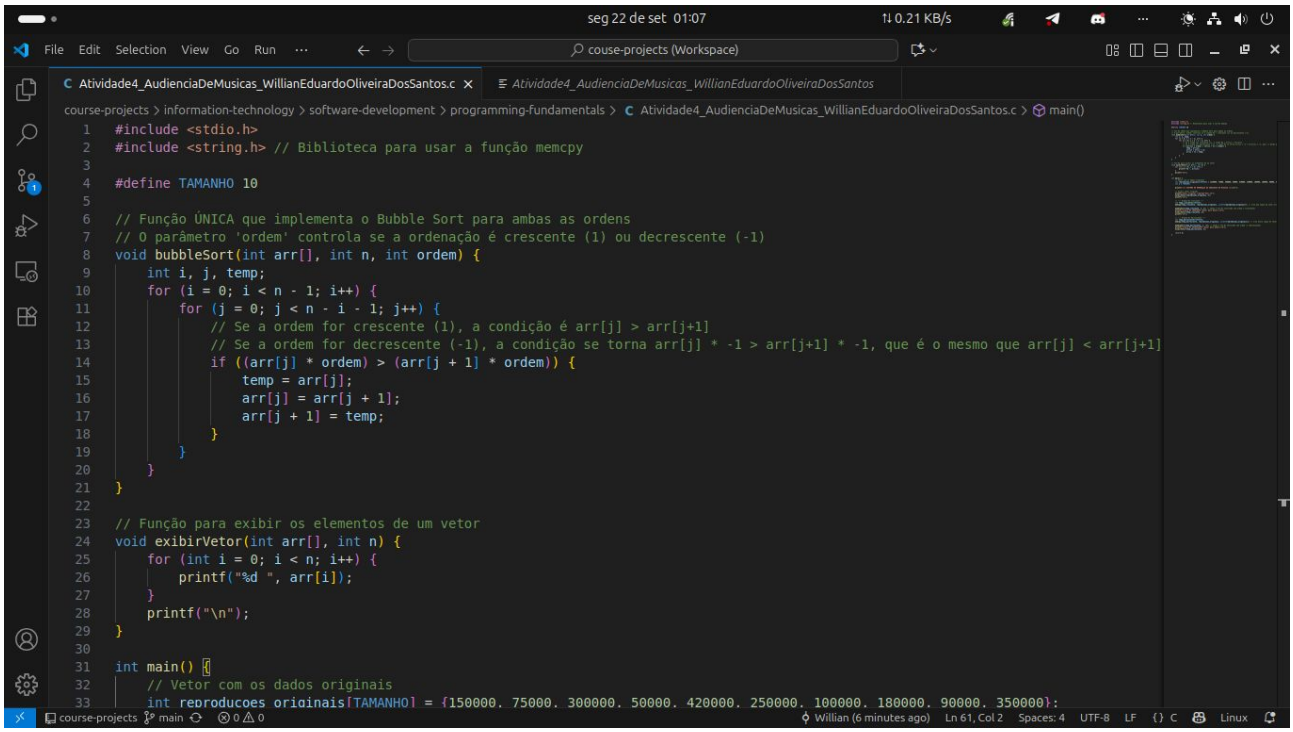
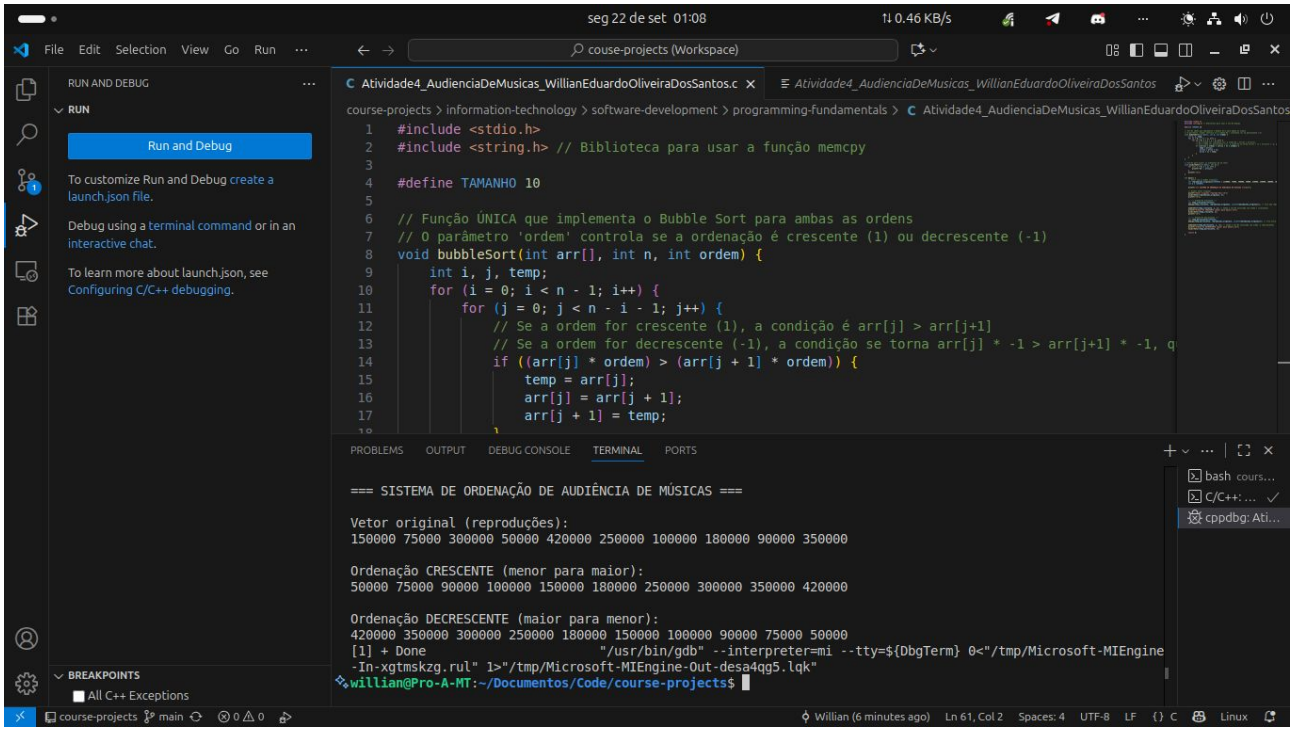
```c
#include <stdio.h>
#include <string.h> // Biblioteca para usar a função memcpy

#define TAMANHO 10

// Função ÚNICA que implementa o Bubble Sort para ambas as ordens
// O parâmetro 'ordem' controla se a ordenação é crescente (1) ou decrescente (-1)
void bubbleSort(int arr[], int n, int ordem) {
    int i, j, temp;
    for (i = 0; i < n - 1; i++) {
        for (j = 0; j < n - i - 1; j++) {
            // Se a ordem for crescente (1), a condição é arr[j] > arr[j+1]
            // Se a ordem for decrescente (-1), a condição se torna arr[j] * -1 >
arr[j+1] * -1, que é o mesmo que arr[j] < arr[j+1]
            if ((arr[j] * ordem) > (arr[j + 1] * ordem)) {
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

// Função para exibir os elementos de um vetor
void exibirVetor(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

int main() {
    // Vetor com os dados originais
    int reproducoes_originais[TAMANHO] = {150000, 75000, 300000, 50000, 420000,
250000, 100000, 180000, 90000, 350000};
    int n = TAMANHO;

    printf("=== SISTEMA DE ORDENAÇÃO DE AUDIÊNCIA DE MÚSICAS ===\n\n");

    // Exibir vetor original
    printf("Vetor original (reproduções):\n");
    exibirVetor(reproducoes_originais, n);
    printf("\n");

    // --- Ordenação Crescente ---
    int temp_crescente[TAMANHO];
    memcpy(temp_crescente, reproducoes_originais, sizeof(reproducoes_originais)); //
Cria uma cópia do vetor original

    bubbleSort(temp_crescente, n, 1); // Chama a função unificada com ordem 1
(crescente)
    printf("Ordenação CRESCENTE (menor para maior):\n");
    exibirVetor(temp_crescente, n);
    printf("\n");

    // --- Ordenação Decrescente ---
    int temp_decrescente[TAMANHO];
    memcpy(temp_decrescente, reproducoes_originais, sizeof(reproducoes_originais));
// Cria outra cópia do vetor original

    bubbleSort(temp_decrescente, n, -1); // Chama a função unificada com ordem -1
(decrescente)
    printf("Ordenação DECRESCENTE (maior para menor):\n");
    exibirVetor(temp_decrescente, n);

    return 0;
}
```


seg 22 de set 01:07
0.21 KB/s
File Edit Selection View Go Run
couse-projects (Workspace)
Atividade4_AudienciaDeMusicas_WillianEduardoOliveiraDosSantos.c
Atividade4_AudienciaDeMusicas_WillianEduardoOliveiraDosSantos
course-projects > information-technology > software-development > programming-fundamentals > Atividade4_AudienciaDeMusicas_WillianEduardoOliveiraDosSantos.c > main()
#include <stdio.h>
#include <string.h> // Biblioteca para usar a função memcpy
#define TAMANHO 10
// Função ÚNICA que implementa o Bubble Sort para ambas as ordens
// O parâmetro 'ordem' controla se a ordenação é crescente (1) ou decrescente (-1)
void bubbleSort(int arr[], int n, int ordem) {
int i, j, temp;
for (i = 0; i < n - 1; i++) {
for (j = 0; j < n - i - 1; j++) {
// Se a ordem for crescente (1), a condição é arr[j] > arr[j+1]
// Se a ordem for decrescente (-1), a condição se torna arr[j] * -1 > arr[j+1] * -1, que é o mesmo que arr[j] < arr[j+1]
if ((arr[j] * ordem) > (arr[j + 1] * ordem)) {
temp = arr[j];
arr[j] = arr[j + 1];
arr[j + 1] = temp;
}
}
}
}
// Função para exibir os elementos de um vetor
void exibirVetor(int arr[], int n) {
for (int i = 0; i < n; i++) {
printf("%d ", arr[i]);
}
printf("\n");
}
int main() {
// Vetor com os dados originais
int reproducoes_originais[TAMANHO] = {150000, 75000, 300000, 50000, 420000, 250000, 100000, 180000, 90000, 350000};
course-projects main
Willian (6 minutes ago) Ln 61, Col 2 Spaces: 4 UTF-8 LF C Linux


seg 22 de set 01:08
0.46 KB/s
File Edit Selection View Go Run
couse-projects (Workspace)
RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug create a launch.json file.
Debug using a terminal command or in an interactive chat.
To learn more about launch.json, see Configuring C/C++ debugging.
BREAKPOINTS
All C++ Exceptions
Atividade4_AudienciaDeMusicas_WillianEduardoOliveiraDosSantos.c
Atividade4_AudienciaDeMusicas_WillianEduardoOliveiraDosSantos
course-projects > information-technology > software-development > programming-fundamentals > Atividade4_AudienciaDeMusicas_WillianEduardoOliveiraDosSantos
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash cours...
C/C++: ...
cppdbg: Ati...
=== SISTEMA DE ORDENAÇÃO DE AUDIÊNCIA DE MÚSICAS ===
Vetor original (reproduções):
150000 75000 300000 50000 420000 250000 100000 180000 90000 350000
Ordenação CRESCENTE (menor para maior):
50000 75000 90000 100000 150000 180000 250000 300000 350000 420000
Ordenação DECRESCENTE (maior para menor):
420000 350000 300000 250000 180000 150000 100000 90000 75000 50000
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-xgtmskzg.rul" 1>"/tmp/Microsoft-MIEngine-Out-desa4qg5.lqk"
willian@Pro-A-MT:~/Documentos/Code/course-projects$
course-projects main
Willian (6 minutes ago) Ln 61, Col 2 Spaces: 4 UTF-8 LF C Linux