

ilp_grafico.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <time.h>
5
6  #define TAMANHO_VETOR 100000000
7
8  // Função para registrar os dados no arquivo CSV
9  void registrar_csv(FILE *arquivo, const char *otimizacao, const char *laco, double tempo) {
10     if (arquivo) {
11         fprintf(arquivo, "%s;%s;%f\n", otimizacao, laco, tempo);
12     }
13 }
14
15 int main(int argc, char *argv[]) {
16     if (argc < 2) {
17         fprintf(stderr, "Uso: %s <Nivel_Otimizacao>\n", argv[0]);
18         return 1;
19     }
20     char *nivel_otimizacao = argv[1];
21
22     FILE *arquivo_csv = fopen("dados_desempenho.csv", "a");
23     if (arquivo_csv == NULL) {
24         perror("Erro ao abrir o arquivo CSV");
25         return 1;
26     }
27
28     int *vetor = (int *)malloc(TAMANHO_VETOR * sizeof(int));
29     if (vetor == NULL) {
30         fprintf(stderr, "Falha na alocao de memoria\n");
31         fclose(arquivo_csv);
32         return 1;
33     }
34
35     struct timespec inicio, fim;
36     double tempo_gasto;
37
38     // --- Laço 1: Inicialização do Vetor ---
39     clock_gettime(CLOCK_MONOTONIC, &inicio);
40     for (int i = 0; i < TAMANHO_VETOR; i++) {
41         vetor[i] = (i % 10) + 1;
42     }
43     clock_gettime(CLOCK_MONOTONIC, &fim);
44     tempo_gasto = (fim.tv_sec - inicio.tv_sec) + (fim.tv_nsec - inicio.tv_nsec) / 1e9;
45     printf("Laço 1 (Inicializacao).....: %f segundos\n", tempo_gasto);
46     registrar_csv(arquivo_csv, nivel_otimizacao, "Laco_1_Inicializacao", tempo_gasto);
47
48     // --- Laço 2: Soma Acumulativa ---
49     long long soma_dependente = 0;
50     clock_gettime(CLOCK_MONOTONIC, &inicio);
51     for (int i = 0; i < TAMANHO_VETOR; i++) {
```

```
52     soma_dependente += vetor[i];
53 }
54 clock_gettime(CLOCK_MONOTONIC, &fim);
55 tempo_gasto = (fim.tv_sec - inicio.tv_sec) + (fim.tv_nsec - inicio.tv_nsec) / 1e9;
56 printf("Laço 2 (Soma com Dependencia).....: %f segundos\n", tempo_gasto);
57 registrar_csv(arquivo_csv, nivel_otimizacao, "Laco_2_Dependente", tempo_gasto);
58
59 // --- Laço 3: Quebra de Dependências (Fator 4) ---
60 long long s1=0, s2=0, s3=0, s4=0;
61 clock_gettime(CLOCK_MONOTONIC, &inicio);
62 for (int i = 0; i < TAMANHO_VETOR; i += 4) {
63     s1 += vetor[i]; s2 += vetor[i+1]; s3 += vetor[i+2]; s4 += vetor[i+3];
64 }
65 long long soma_total_indep4 = s1 + s2 + s3 + s4;
66 clock_gettime(CLOCK_MONOTONIC, &fim);
67 tempo_gasto = (fim.tv_sec - inicio.tv_sec) + (fim.tv_nsec - inicio.tv_nsec) / 1e9;
68 printf("Laço 3 (Quebra de Dependencia, Fator 4): %f segundos\n", tempo_gasto);
69 registrar_csv(arquivo_csv, nivel_otimizacao, "Laco_3_Fator_4", tempo_gasto);
70
71 // --- Laço 4: Quebra de Dependências (Fator 8) ---
72 long long a1=0, a2=0, a3=0, a4=0, a5=0, a6=0, a7=0, a8=0;
73 clock_gettime(CLOCK_MONOTONIC, &inicio);
74 for (int i = 0; i < TAMANHO_VETOR; i += 8) {
75     a1 += vetor[i]; a2 += vetor[i+1]; a3 += vetor[i+2]; a4 += vetor[i+3];
76     a5 += vetor[i+4]; a6 += vetor[i+5]; a7 += vetor[i+6]; a8 += vetor[i+7];
77 }
78 long long soma_total_indep8 = a1+a2+a3+a4+a5+a6+a7+a8;
79 clock_gettime(CLOCK_MONOTONIC, &fim);
80 tempo_gasto = (fim.tv_sec - inicio.tv_sec) + (fim.tv_nsec - inicio.tv_nsec) / 1e9;
81 printf("Laço 4 (Quebra de Dependencia, Fator 8): %f segundos\n", tempo_gasto);
82 registrar_csv(arquivo_csv, nivel_otimizacao, "Laco_4_Fator_8", tempo_gasto);
83
84
85
86 //Um resultado final para garantir que todas as somas são essenciais.
87 long long total_geral = soma_dependente + soma_total_indep4 + soma_total_indep8;
88 printf("\nVerificacao de Somas (Dependente: %lld, Fator 4: %lld, Fator 8: %lld)\n",
89     soma_dependente, soma_total_indep4, soma_total_indep8);
90
91 free(vetor);
92 fclose(arquivo_csv);
93
94 // O valor de retorno do programa agora depende dos resultados.
95 return (int)(total_geral % 256);
96 }
```