Trabalho Prático 2
Buscador de maior subvetor - Implementação em C++

Eduardo Pinto Esteves Farias
Departamento de Ciência da Computação – Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, MG – Brasil

eduardofarias@ufmg.br

1.Introdução

O programa tem como objetivo principal identificar o subvetor de maior soma contido dentro de um vetor principal. É possível que no nosso vetor haja notas negativas, por isso, apesar do vetor inteiro ser um subvetor dele mesmo, não necessariamente ele terá a maior soma.

2.Implementação

O desenvolvimento foi realizado na linguagem C++ e compilado pelo G++ da GNU Compiler Collection. O sistema operacional usado foi o Windows 10 - build 1909. O processador utilizado foi o Ryzen 5 1600 em um sistema com 16GB de RAM disponíveis.

Foi necessária a criação de uma struct “soma”. A função desse struct é armazenar os índices e o valor da soma dos subvetores computados pelo programa.

O programa necessitou de 2 funções além da função main para a implementação. Cada uma delas foi modularizada individualmente. Essas funções correspondem às funções contidas no livro “Introduction to Algorithms”, de Thomas Cormen, nas páginas 71 e 72. A escolha deste algoritmo se deu por conta da restrição do tipo *divide and conquer* imposta, logo, a escolha do algoritmo do Cormen foi vista como a melhor possível.

A seguir, as funções são explicadas individualmente:

Arquivo “maxRegSub.hpp” - Função “findMaxSubarray”

Briefing: Primeira parte do algoritmo do Cormen. Recursivamente, divide o vetor em vetores menores e a partir deles faz a soma dos elementos. Porém, para ficar completo, necessita da função “findMaxCrossingSubarray”, que será explicada a

seguir. Ao final, a função compara qual o maior subvetor encontrado e o retorna, com os índices que demarcam o subvetor e o valor da soma encontrado.

Arquivo “maxCrossSub.hpp” - Função “findMaxCrossingSubarray”

Briefing: Segunda parte do algoritmo do Cormen. Determina o valor da soma da maior subarray que “atravessa” entre as duas divisões do vetor criadas pela função anterior. Retorna o tipo “soma”, com os índices do subvetor encontrado e o valor da soma.

Arquivo “msgassert.h” - Função “erroAssert”

Briefing: Garante que a entrada, como número de amigos, shows e notas, estejam dentro dos padrões estipulados.

Função “main”

Briefing: Realiza a leitura dos dados iniciais de entrada, e caso sejam válidos, entra no loop onde a execução do programa em si acontece. Os dados são armazenados em um vetor que é passado para a função “findMaxSubarray” tratar. Ao final, imprime os índices correspondentes ao maior subvetor encontrado

3.Bibliografia:

Cormen, Thomas H.- Introduction to Algorithms, Third Edition

4.Instruções para execução:

Para executar o projeto, dentro da pasta TP, é necessário executar o comando make. Após isso, o executável “tp02” estará na pasta bin. Basta executá-lo passando as entradas corretas que a saída deseja será impressa no terminal.