# Documentação Trabalho Prático 2

**Lucas Roberto Santos Avelar**

Departamento de Ciência da Computação - Universidade Federal de Minas Gerais (UFMG) - Belo Horizonte – MG – Brasil

lucasrsavelar@ufmg.br

## 1. Introdução

Esta documentação lida com o problema da simulação de um aplicativo utilizado por amigos com a finalidade de somar a nota de cada um deles para cada show por eles presenciado em um festival e, ao final, descobrir qual o intervalo de shows cuja soma das notas é a maior possível. Percebe-se, então, que temos um problema ampliado de cálculo de subvetor de soma máxima, em que além de calcular o maior intervalo existente dentro do vetor de notas, também é necessário manter o registro dos índices inicial e final desse subvetor.

## 2. Implementação

O código organiza-se principalmente em 2 blocos. Dentro dos arquivos, encontram-se alguns comentários sobre as operações de cada função. A seguir, há uma análise mais detalhada.

O primeiro bloco corresponde ao arquivo “main.cpp”, onde está localizada a função main do programa. Trata de um bloco mais trivial, que apenas declara algumas variáveis necessárias e lê da entrada padrão os valores da quantidade de amigos e da quantidade de shows. Após isso, o programa entra em loop enquanto os valores de amigos e shows não são inválidos (ou seja, ambos 0). Dentro desse loop, cria-se um vetor que armazena as notas, esse vetor é preenchido com os dados advindos da entrada padrão e passado como parâmetro para a função SSM (Subvetor de Soma Máxima, será discutida posteriormente), em conjunto com suas posições de início e fim. A função retorna uma estrutura que contém a solução (soma do maior subvetor) e seus índices, que são acrescentados de 1 (visto que o vetor inicia na posição 0 e os shows começam do 1). Finalmente, os índices

são impressos na saída padrão e as próximas entradas de número de amigos e de shows são lidas.

Já o segundo bloco consiste nos arquivos “rock.hpp” e “rock.cpp”, que reúnem respectivamente a declaração e a implementação das funções fundamentais para o funcionamento do programa, bem como das estruturas necessárias e suas variáveis.

As estruturas consistem em Indice, que armazena o valor (soma) de uma determinada porção do vetor, em qual índice essa porção se inicia e em qual ela termina; e em Solucao, que armazena instâncias de Indice para o prefixo do vetor, para o sufixo, para o vetor como um todo e para o ssm.

Nas funções, SSM recebe o vetor de notas, seu início e seu final e retorna uma instância de Solucao contendo um Indice ssm que possui o subvetor com a maior soma possível e os índices (start e end) desse subvetor. Para isso, primeiro a função verifica se o início e o fim são iguais. Se sim, significa que foi passado apenas uma posição do vetor e, portanto, apenas um elemento, então cria-se um Indice auxiliar que armazena o valor desse elemento e suas posições (iguais), e esse Indice auxiliar é atribuído a todas as instâncias da solução. Caso sejam diferentes, primeiro define-se o meio do vetor e então duas chamadas recursivas são realizadas: uma para a esquerda do vetor (passando o vetor, o início e o meio como parâmetros) e uma para a direita (passando o vetor, a posição seguinte ao meio e o final). As instâncias de Solucao advindas dessas chamadas recursivas, além da instância que será efetivamente retornada, são passadas como parâmetro (por referência) para a função RecuperaIndices, que calcula o subvetor de soma máxima e seus índices. Finalmente, retorna-se a instância de Solucao com as respostas.

A função RecuperaIndices consiste no principal núcleo do programa, já que é responsável por calcular os subvetores, encontrar o maior e determinar seus índices. Primeiramente, ela verifica qual o maior prefixo para a solução, comparando se a soma total da esquerda mais o prefixo da direita é maior do que o sufixo da esquerda. Se for, então o prefixo da solução recebe a soma desses valores, o índice de início do prefixo da esquerda e o índice final do prefixo da direita. Caso contrário, o prefixo da solução recebe o prefixo da esquerda. As mesmas ideias são repetidas logo a seguir, porém para calcular o sufixo da solução, o que

demanda comparar a soma entre o total da direita e o sufixo da esquerda com o sufixo da direita.

A função então calcula a soma total da solução simplesmente somando o total da esquerda e o total da direita. O início da soma total é o início da esquerda e o fim é o final da direita.

Ocorre, então, uma série de comparações para efetivamente determinar qual a soma máxima da solução. Para isso, o sufixo da solução, o prefixo da solução, a soma máxima da direita e a soma máxima da esquerda são comparados entre si (e com a soma total da solução) a fim de verificar qual possui o maior valor, que é atribuído à solução. Vale ressaltar a presença do “maior ou igual” para auxiliar nos casos de desempate (o vetor {1, 2, 3, 0} apresenta soma máxima 6 tanto do primeiro ao terceiro elemento quanto do primeiro ao quarto. Como a decisão é sempre por assistir um show que não altera a nota, deve-se considerar o subvetor de soma máxima com maior tamanho).

Finalmente, ocorre uma última comparação entre a melhor solução encontrada até agora com o prefixo da esquerda e o sufixo da direita, visando verificar se a maior soma está cruzando o meio do vetor e, portanto, está contida tanto na parte esquerda quanto na direita (o que ainda não havia sido testado; já que foram comparadas apenas a parte esquerda/direita ou sufixos/prefixos contidos inteiramente em uma das partes). Se o subvetor que cruza as partes for a maior soma, então atualiza-se o valor e os índices da soma máxima da solução. Como a passagem de parâmetros foi por referência, não é necessário retornar nada e, portanto, está terminado o cálculo. Saindo da RecuperaIndices, o programa volta para a SSM, que retorna para o main a solução atualizada com os valores e índices.