TP2 - Avaliação do evento

Algoritmos 1
Data de entrega: 11/11/2022

1 Objetivos do trabalho

O objetivo deste trabalho é modelar o problema computacional descrito a seguir utilizando uma estrutura de dados que permita resolvê-lo de forma eficiente com os algoritmos estudados nesta disciplina.

Serão fornecidos alguns casos de teste bem como a resposta esperada para que o aluno possa verificar a corretude de seu algoritmo. Não obstante, recomenda-se que o aluno crie casos de teste adicionais a fim de validar sua própria implementação.

A sua solução deve obrigatoriamente ser desenvolvida utilizando algoritmos de divisão e conquista.

O código-fonte da solução e uma documentação sucinta (relatório contendo não mais do que 5 páginas) deverão ser submetidos via moodle até a data limite de **11/11/2022**. A especificação do conteúdo do relatório e linguagens de programação aceitas serão detalhadas nas seções subsequentes.

2 Definição do problema

A cada 4 anos acontece o festival Rock In Rio, que é um evento que reúne várias pessoas que gostam de música em um único lugar. A programação do evento conta com um conjunto de shows que acontece de forma sequencial. Como o evento ainda é bem pequeno, eles não podem bancar mais de um palco ao mesmo tempo. Outro detalhe que torna esse evento interessante é que todo ano o festival convida as mesmas bandas, e garante que elas vão tocar na mesma ordem do ano anterior.

Neste ano, um grupo de amigos decidiu ir ao festival e escutar todas as bandas disponíveis. Esses amigos, por acaso, fazem parte de uma turma de computação e fizeram um aplicativo para cada um avaliar sua experiência ao longo do festival. Com esse aplicativo cada membro do grupo de amigos pode dar uma nota de -5 até +5 para cada show. Como requisito do aplicativo, todos os amigos do grupo devem avaliar todos os shows que aconteceram no evento. O objetivo deles com esse aplicativo é **somar** todas as avaliações de um mesmo show e descobrir qual será o intervalo de shows que mais agradou o grupo para que eles possam planejar o retorno no próximo festival, uma vez que, se eles saírem do festival, não podem voltar. Vale lembrar que na hora de montar o planejamento, sempre que existir um empate na decisão de assistir ou não um show, os amigos sempre optam por assistir ao show.

3 Exemplo do Problema

Em um caso hipotético, 3 amigos foram ao evento que teve 6 bandas diferentes tocando.

- A avaliação do primeiro amigo foi: 0, 3.5, -4, 2, 4 e 1.
- A avaliação do segundo amigo foi: 0, 2.7, -5, 0, 4 e 1.
- A avaliação do terceiro amigo foi: 1, 3, 5, 3 , 0 e -4.

De acordo com as avaliações desse grupo de amigos, no próximo evento eles assistirão apenas aos shows das bandas 1, 2, 3, 4 e 5. Note que, mesmo com o show 3 não sendo muito bem avaliado, vale a pena continuar no festival para assistir os shows 4 e 5.

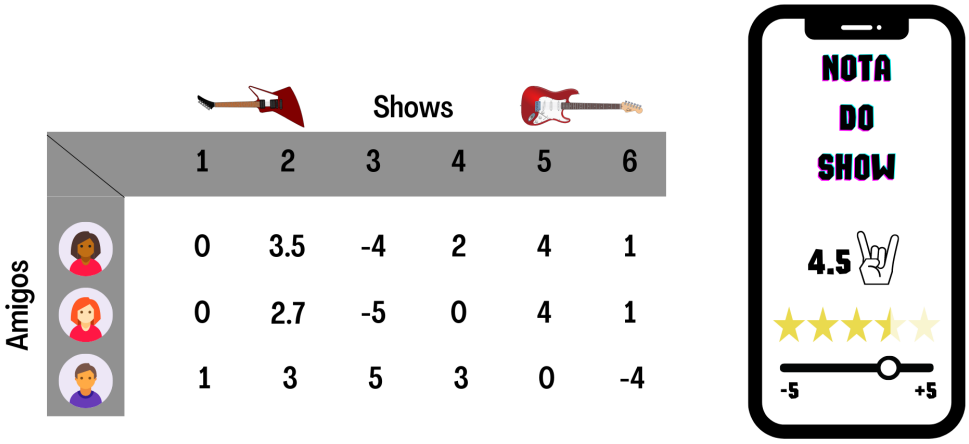

Figure 1: Exemplo de notas para 3 amigos e 5 shows

3.1 Modelagem do Problema

Este trabalho prático aborda a parte de divisão e conquista da ementa desta disciplina. Para a resolução do problema a sua modelagem **precisa** usar algoritmos de divisão e conquista. Caso contrário ele não vai rodar a tempo.

3.2 Formato da Entrada Esperada

O seu programa deverá processar diversos casos de teste em uma única execução. A primeira linha de um cenário de teste é composta de dois número inteiros A e S, representando respectivamente o a quantidade de amigos que assistiram os shows ($1 \leq A \leq 50$) e a quantidade de shows que aconteceram no festival ($1 \leq S \leq 100000$). Cada uma das próximas A linhas descreve a avaliação de um integrante do grupo. A linha de avaliações é uma sequencia de S números reais separados por espaço. Uma linha com $A = 0$ e $S = 0$ indica o final do arquivo.

3.3 Formato da Saída Esperada

Para cada caso de teste seu programa deve imprimir uma linha contendo dois números inteiros ,X_i e X_f, separados por espaço, sendo que $1 \leq X_i < X_f \leq S$. X_i é o índice do primeiro show que eles devem assistir e X_f é o índice do último show que eles devem assistir. Repare que o intervalo $[X_i, X_f]$ é inclusivo nas duas pontas.

3.4 Caso de teste

Entrada

```
3 6
0 3.5 −4 2 4 1
0 2.7 −5 0 4 1
1 3 5 3 0 −4
5 5
5 −2 3 5 −2
3 −5 2 5 0
2 0 3 5 −2
4 1 1 5 1
3 2 3 5 3
2 7
−2 −4 3 1 −5 −1 3
−3 −5 2 4 −5 −2 3
10 4
4.1 −2 3 1
3 −5 2 3.7
4.4 0 2.8 0.88
3.8 1 1 −5
3 2 3 2
5 5 5 5
1.3 −2.2 0 −4
4 3 0 0
−5 −5 −5 −5
−2 −1 0.5 1
0 0
```

Saída

```
1 3
1 4
3 4
1 3
```

.

4 Implementação

O seu programa deverá ser implementado na linguagem C++ e deverá fazer uso apenas de funções da biblioteca padrão da linguagem. Trabalhos que utilizem qualquer outra linguagem de programação e/ou que façam uso de bibliotecas que não a padrão não serão aceitos.

O aluno pode implementar seu programa em qualquer ambiente (Windows, Linux, MacOS, etc...), no entanto, deve garantir que seu código compile e rode nas máquinas do DCC (tigre.dcc.ufmg.br ou jaguar.dcc.ufmg.br ou login.dcc.ufmg.br), pois será neste ambiente que o TP será corrigido. Note que essas máquinas são acessíveis a todos os alunos do DCC com seu login e senha, podendo inclusive ser realizado acesso remoto via ssh. O aluno pode buscar informações no site do CRC (Centro de Recursos Computacionais) do DCC (https://www.crc.dcc.ufmg.br/).

Para facilitar o desenvolvimento vamos fornecer uma estrutura base de arquivos com Makefile já configurado. A pasta `TP02-Template-CPP.zip`, disponível para download na tarefa do Moodle, contem 4 arquivos: `main.cpp`, `rock.cpp`, `rock.hpp` e `Makefile`. O ponto de entrada do seu programa está no arquivo `main.cpp`. Para compilar seu programa basta executar o comando "`make`" no mesmo diretório que o Makefile está. Ao final deste comando, se a compilação for bem sucedida, será criado um arquivo executável chamado "`tp02`". Esse arquivo pode ser executado pela linha de comando usando "`./tp02`".

O arquivo da entrada deve ser passado ao seu programa como entrada padrão, através da linha de comando (e.g., `$ ./tp02 < casoTeste02.txt`) e gerar o resultado também na saída padrão (não gerar saída em arquivo).

Por favor não altere o nome da pasta nem o nome do arquivo executável. Parte da correção é automática e é extremamente importante manter o padrão de execução.

5 O que deve ser entregue

Deverá ser submetido um arquivo .zip contendo apenas uma pasta chamada tp2, esta pasta deverá conter: (i) Documentação em formato PDF e (ii) Implementação.

5.1 Documentação

A documentação deve ser sucinta e não ultrapassar 5 páginas. Você deve descrever cada solução do problema de maneira clara e precisa, detalhando e justificando os algoritmos e estruturas de dados utilizados. Para tal, artifícios como pseudo-códigos, exemplos ou diagramas podem ser úteis. Note que documentar uma solução não é o mesmo que documentar seu código. Não é necessário incluir trechos de código em sua documentação nem mostrar detalhes de sua implementação, exceto quando estes influenciem o seu algoritmo principal, o que se torna interessante. É essencial que a documentação contenha ao menos:

1. **Identificação**: Nome e Matrícula.
2. **Introdução**: Breve resumo do problema com suas palavras.
3. **Modelagem**: Detalhamento e justificativa dos algoritmos e estruturas de dados escolhidos.

5.2 Implementação

O código fonte submetido deve conter todos os arquivos fonte e o Makefile usado para compilar o projeto. Lembre que seu código deve ser **legível**, então **evite variáveis com nomes não descritivos** (`int a, aa, aaa;`) e lembre-se de **comentar seu código**. Já estamos fornecendo uma implementação base com os arquivos necessários, então indicamos que você só o altere se for necessário.

5.3 Atrasos

Trabalhos poderão ser entregues após o prazo estabelecido, porém sujeitos a uma penalização regida pela seguinte fórmula:

$$\Delta_p = \frac{2^{d-1}}{0.32}\% \tag{1}$$

Nesta fórmula d representa dias de atraso. Por exemplo, se a nota dada pelo corretor for 70 e você entregou o TP com 4 dias corridos de atraso, sua penalização será de $\Delta_p = 25\%$ e, portanto, a sua nota final será: Nf$= 70 * (1 - \Delta_p) = 52.2$. Note que a penalização é exponencial e 6 dias de atraso resultam em uma penalização de 100%.

6 Considerações Finais

Assim como em todos os trabalhos dessa disciplina é estritamente proibida a cópia parcial ou integral de códigos, seja da internet ou de colegas. Utilizaremos o algoritmo MOSS para detecção de plágio em trabalhos, seja honesto. Você não aprende nada copiando código de terceiros nem pedindo a outra pessoa que faça o trabalho por você. Se a cópia for detectada, sua nota será zerada e os professores serão informados para que as devidas providências sejam tomadas.