CEFET-MG
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS

Projeto – Parte I

Laboratório de Algoritmos e Estruturas de Dados I

Profª. Natália Batista

Nomes: _____Total: 15 pontos

Instruções

A Parte I do Projeto de LAEDI será realizado nas aulas práticas de 19 e 26/05/23 e poderá ser realizado em duplas (enviar pelo SIGAA os nomes dos integrantes da dupla). A entrega poderá ser realizada até 01/06/23, pelo sistema run.codes.

A solução deverá ser implementada pelos(as) próprios(as) alunos(as) em sistema Linux na linguagem C ou C++ e não será permitido a utilização de trechos de códigos de outras pessoas ou retirados da internet.

Busca por padrão em sequência

Uma imagem digital é uma matriz de elementos, podendo ter duas ou mais dimensões, em que cada elemento é um pixel com uma cor ou tom de cinza. Cada cor pode ser representada por um número (tom de cinza) ou por uma tupla (por exemplo, quando utiliza-se o sistema de cores RGB, que é a sigla para *Red Green Blue*, a representação da cor será uma tripla com três valores).

Nesta primeira etapa do projeto, não será necessário trabalhar diretamente com imagens digitais, mas vamos tomá-las como exemplo para explicar o problema a ser resolvido. Considere as seguintes imagens de uma pista em miniatura:

Quando a pista é percorrida com uma câmera e as imagens de cada instante são obtidas, surgem visões diferentes da pista, principalmente nas curvas, em que pode ocorrer cortes na visão da pista. Observe, por exemplo, a sequência de imagens a seguir:





Nessa sequência de imagens, a partir da primeira, é possível ver que há uma curva próxima e, à medida que se caminha na pista e a curva se aproxima, a câmera captura cada vez menos as bordas da pista (que são as linhas brancas) devido ao seu campo de visão e ao seu movimento. Na última imagem da sequência, é possível ver somente a borda direita, ou seja, parte da pista foi cortada.

Imagine agora que a pista foi colorida de vermelho. Se fizermos um corte na imagem e tormarmos apenas uma linha (desenhada em amarelo nas imagens abaixo), veremos que, quando a imagem da pista aparece sem cortes (primeira imagem), as cores que aparecem na altura da linha são preto, branco, vermelho, branco e preto, da esquerda para a direita.


Quando a imagem da pista aparece com cortes, outras possibilidades são possíveis, por exemplo, na segunda imagem temos a seguinte sequência de cores na altura da linha amarela: vermelho, branco e preto, da esquerda para a direita.

O objetivo da primeira parte do projeto é, dada uma linha da imagem, buscar pelo padrão de sequência de cores que indica que a pista está sendo vista por completo (da borda esquerda até a borda direita).

No exercício de programação “Análise de segmentos com elementos iguais”, foi fornecido um arquivo com os valores (“cores”) dos elementos de uma linha e foi realizada a análise dos tipos dos segmentos e a contagem do número de elementos do respectivo segmento.

Se considerarmos o seguinte mapeamento dos valores do vetor para os números inteiros (o mapeamento será fixo e sempre o mesmo em todas as etapas do projeto, independentemente dos valores dos elementos fornecidos):

0 (preto) -> 1
128 (vermelho*) -> 2
255 (branco) -> 3

*Observação: na verdade o valor 128 representa um tom de cinza mas que no exemplo será entendido como a cor vermelha para simplificar.

Então o padrão da pista completa que estamos buscando é a seguinte sequência dos tipos dos segmentos:

1 3 2 3 1

Que representam as cores: preto, branco, vermelho, branco e preto. A sequência dos tipos dos segmentos foi parte da saída do exercício de programação citado e, portanto, o código entregue poderá ser aproveitado na solução desta parte do projeto.

Se eu já tenho um programa que imprime a sequência dos tipos dos segmentos, o que devo fazer então?

Primeiramente, deverá ser criada uma lista que irá armazenar em cada elemento as informações de cada segmento. Utilize uma das implementações do livro texto do Ziviani para o tipo abstrato de dados Lista (por meio de arranjo ou apontadores), disponíveis em:

http://www2.dcc.ufmg.br/livros/algoritmos/implementacoes-03.php

Cada item da lista, que representa um segmento, deverá conter: um campo **Chave** (número inteiro que será atribuído sequencialmente a cada semento na ordem em que é lido do arquivo), um campo **Tipo** (tipo do segmento), **NumElementos** (número de elementos do segmento) e **PontoMedio** (número inteiro com a posição do ponto médio do segmento). Se uma linha possuir N elementos (pixels), de 0 a N-1, então o ponto médio do segmento será (p+q)/2, onde p e q são as posições do início e do fim do segmento. Outros campos poderão ser adicionados, se necessário.

É obrigatório utilizar as funções da implementação do Ziviani e não é permitido alterá-las, com exceção da função main e das definições solicitadas neste enunciado. Funções adicionais poderão ser criadas, se necessário.

Após criar a lista, o programa deverá verificar se o padrão “1 3 2 3 1” está presente em alguma parte dessa sequência de elementos. Por exemplo, considere a sequência dos tipos dos segmentos:

1	3	1	3	2	3	1	3	1

Nesse exemplo há o padrão da pista completa em uma parte da sequência (colorido em amarelo). No exemplo a seguir não há o padrão de pista completa:

1	3	1	3	1	3	1

Há várias outras possibilidades de sequências em que o padrão pode ou não estar presente. O padrão será considerado presente na sequência somente se a sub-sequência "1 3 2 3 1" for encontrada nesta ordem.

Formato de entrada dos dados

Os valores dos pixels (elementos) da linha da imagem serão fornecidos em um arquivo texto com o seguinte formato: na primeira linha o valor de N e na linha seguinte os N valores dos elementos separados por um espaço em branco. Todos valores são números inteiros. Esse formato é igual ao do exercício de programação "Análise de segmentos com elementos iguais".

O programa deverá solicitar o nome do arquivo de entrada.

Formato de saída dos dados

O programa deverá produzir a impressão na tela conforme o modelo a seguir, de acordo com o resultado encontrado (não utilize acentuação nem cedilha na saída para esse programa):

```
Digite o nome do arquivo: teste.txt
Resultado: Padrao encontrado.
```

ou

```
Digite o nome do arquivo: teste.txt
Resultado: Padrao nao encontrado.
```

Observações importantes:

- Por favor, leia **todas** as informações do enunciado antes de enviar o programa para o run.codes.
- Caso tenha alguma dúvida ou dificuldade, entre em contato com antecedência, evitando deixar para a véspera da entrega.
- O exercício poderá ser realizado em duplas.
- O programa deverá solicitar ao usuário o nome do arquivo a ser testado. O programa deverá permitir testes com quaisquer arquivos no formato especificado (com nomes distintos, com qualquer valor de N e com quaisquer valores dos elementos).
- O programa deverá ser bem organizado, identado e conter comentários explicativos relevantes.
- Os testes do programa entregue serão realizados pelo sistema run.codes, conforme as

instruções de utilização do sistema que podem ser consultadas pelo SIGAA.

- O programa deverá compilar sem erros ou avisos (*warnings*) com o compilador *gcc/g++*. Programas que não compilarem ou que apresentarem erros de execução (falhas de segmentação, etc) não serão corrigidos. Não serão aceitos arquivos enviados por e-mail e nem *'prints'* da execução do programa como prova de seu funcionamento.
- Se houver indícios de plágio no código fonte do programa, a nota final da atividade será zero e serão aplicadas as penalidades previstas no Regime Disciplinar Discente.