CEFET-MG
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS

Projeto – Parte II

Laboratório de Algoritmos e Estruturas de Dados I

Profª. Natália Batista

Nomes: _____Total: 15 pontos

Instruções

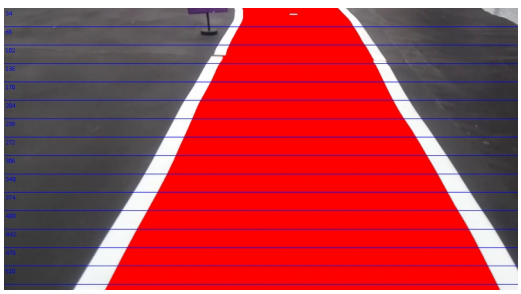
A Parte II do Projeto de LAEDI será realizada na aula prática de 14/01/22 e poderá ser realizado em duplas (deverá ser mantida a mesma dupla da Parte I). A entrega poderá ser realizada até 21/01/22, pelo sistema run.codes.

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 alteração em padrão da sequência – Impedimentos na pista

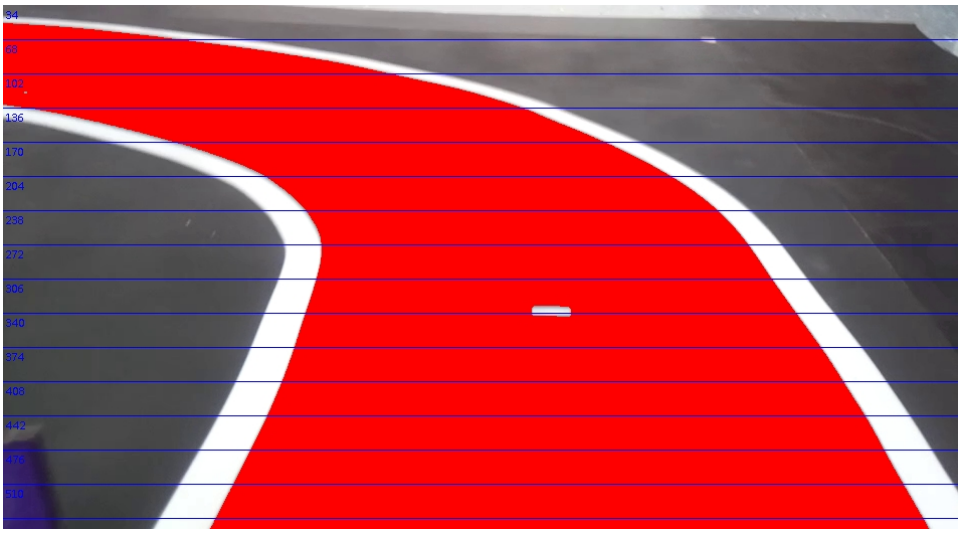
Na primeira etapa do projeto, foi implementada a busca pelo padrão da pista em uma linha da imagem (desenhada em amarelo na imagem de exemplo abaixo). Esse padrão é uma sequência de cores que indica que a pista está sendo vista por completo (da borda esquerda até a borda direita). A sequência de cores que aparecem na altura da linha amarela é: preto, branco, vermelho, branco e preto.

Considere que agora não há apenas um, mas **L** linhas selecionadas na imagem, como mostrado na figura abaixo, em que cada linha está colorida de azul.



O objetivo da segunda etapa do projeto é realizar a detecção de impedimentos (obstáculos) em uma imagem da pista de forma simples, considerado as informações de vários perfis (linhas) da imagem. Dessa forma, cada perfil da imagem deverá ser analisado e o resultado (Pista normal, Pista impedida ou Padrão não identificado) deverá ser impresso.

Quando há algum impedimento na pista, o padrão da pista (1 3 2 3 1) é alterado, aparecendo outros valores na região 2. Por exemplo, na imagem abaixo, há um pequeno obstáculo no centro da pista. O perfil (linha) que corta esse obstáculo, apresenta a seguinte sequência dos tipos dos segmentos: 1 3 2 3 2 3 1.

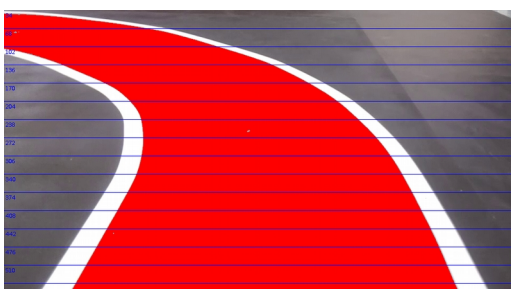


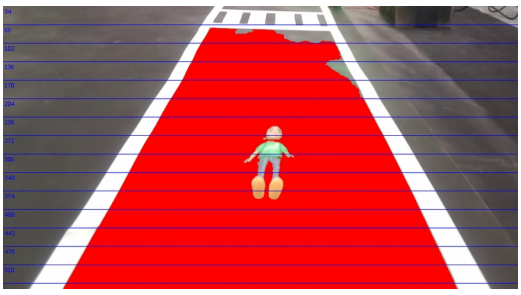
Observe que o padrão da pista foi alterado pelo obstáculo, pois surgiu uma nova região no meio do segmento 2 (que corresponde à pista), dividindo-o. Outros tipos de obstáculos poderão gerar diferentes segmentos na região da pista, como ilustrado na região azul da figura a seguir. As regiões brancas e azul podem conter vários segmentos de tipos distintos e a região amarela corresponde ao padrão original da pista.

...	**1**	**3**	**2**	**...**	**2**	**3**	**1**	**...**

Para a implementação da Parte II, o código da Parte I poderá ser utilizado, mantendo a utilização de uma das implementações do livro texto do Ziviani para o tipo abstrato de dados Lista (por meio de arranjo ou apontadores).

Cada grupo deverá escolher qual método será utilizado para realizar a detecção de obstáculos (por exemplo, pode ser considerado que há obstáculos se em pelo menos um perfil for detectado o padrão da faixa alterado por obstáculo). Lembrem-se de levar em consideração que há perfis diversos na imagem, alguns somente com o padrão da pista completa, outros com falhas devido à iluminação ou movimento do robô, etc.





Formato de entrada dos dados

Os valores dos **N** pixels (elementos) dos **L** perfis da imagem serão fornecidos em um arquivo texto com o seguinte formato: na primeira linha o valor de L e nas linhas seguintes: o valor de N e na próxima linha os N valores dos elementos do perfil separados por um espaço em branco. Todos valores são números inteiros.

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: Pista sem impedimento.
```

ou

```
Digite o nome do arquivo: teste.txt
Resultado: Pista com impedimento.
```

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. Os horários e link para a monitoria estão divulgados no SIGAA.
- O exercício poderá ser realizado em dupla (a mesma da parte anterior do projeto).
- 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 quaisquer valores de L, de N e dos valores dos elementos).
- O programa deverá ser bem organizado, identado e conter comentários explicativos relevantes.
- É obrigatório utilizar as funções da implementação do livro texto do Nivio Ziviani (Projeto de Algoritmos).
- 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.