Instituto Federal de Minas Gerais - Campus Ouro Branco
Curso: Bac. em Sistemas de Informação
Disciplina: Arquitetura e Organização de Computadores
Professor: Saulo Henrique Cabral Silva (saulo.cabral@ifmg.edu.br)

Trabalho Prático 1
Técnicas de Detecção e Correção

Valor: 10 pontos **Data da entrega: 27/04/2025**

Objetivos: Revisar e aplicar conceitos de orientação a objetos utilizando Java, além de compreender e implementar técnicas clássicas de detecção e correção de erros em sistemas de comunicação digital: Código de Redundância Cíclica (CRC) e Código de Hamming.

Descrição:

Neste trabalho vocês deverão simular um sistema de comunicação digital composto por três elementos: **Transmissor**, **Canal de Comunicação** e **Receptor**. O sistema será responsável por transmitir uma sequência de caracteres, podendo ser afetado por **ruídos** introduzidos pelo canal.

A proposta é comparar, na prática, o funcionamento e a eficiência das técnicas de **CRC** e **Hamming**. Para isso, os alunos devem implementar duas abordagens distintas, utilizando as mesmas estruturas, mas alterando o comportamento do transmissor e receptor conforme a técnica adotada.

O sistema deve ser capaz de lidar com erros de 1 bit ou múltiplos bits introduzidos durante a transmissão, conforme a configuração de probabilidade definida pelo usuário e a técnicas escolhida para detecção e correção entre transmissor e receptor.

Neste trabalho você receberá um projeto com algumas funcionalidades desenvolvidas, e outras que você deverá desenvolver para que as técnicas de detecção e/ou correção possam resolver o problema dos ruídos durante a transmissão. Abaixo a descrição do projeto que você receberá.

Componentes (classes) do Sistema:

- O **Transmissor** codifica a mensagem original utilizando uma das técnicas (*CRC* ou *Hamming*).
- O **Canal de Comunicação** transmite os bits codificados e, com base nas *probabilidades informadas*, pode gerar ruídos.
    - Probabilidade de ocorrer um erro em um único bit.
    - Probabilidade de ocorrerem múltiplos erros na mensagem.
- O **Receptor** deve ser capaz de decodificar a mensagem recebida, detectar e/ou corrigir erros, e fornecer um relatório da transmissão:
    - Na técnica **CRC**, detectar erros e solicitar o reenvio da mensagem em caso de falha.
    - Na técnica **Hamming**, identificar e corrigir automaticamente um erro de 1 bit, e notificar falhas irrecuperáveis em caso de múltiplos erros.

Testes e resultados:

Após concluir a implementação das técnicas de CRC e Hamming, você deverá realizar testes com diferentes configurações de ruído e analisar os resultados obtidos. Para isso, utilize como mensagem de entrada o conteúdo completo do livro Moby Dick, em formato .txt.

O objetivo desta etapa é comparar o comportamento das duas técnicas em relação à eficiência (tempo de execução) e à robustez diante de erros, variando os seguintes parâmetros:

- Probabilidade de erro em 1 bit: 0%, 10%, 20%, 30%, 40%, 50%;
- Probabilidade de erro em múltiplos bits: 0%, 5%, 10%.

Para cada combinação dessas probabilidades, registre o tempo total de execução da transmissão para ambas as técnicas. Organize os dados em duas tabela similares à abaixo (uma para os tempos da técnica CRC e outra para Hamming), substituindo os campos "Tempo x" pelos valores reais obtidos:

Prob1 / Prob2	0	10	20	30	40	50
0	Tempo a	Tempo b	Tempo c	Tempo d	Tempo e	Tempo f
5	Tempo g	Tempo h	Tempo i	Tempo j	Tempo k	Tempo l
10	Tempo m	Tempo n	Tempo o	Tempo p	Tempo q	Tempo r

Com base nos dados coletados, elabore uma análise crítica dos resultados, respondendo às seguintes questões:

- Como o tempo de execução varia conforme as probabilidades de erro aumentam?
- Qual técnica se mostrou mais eficiente em ambientes com pouco ruído?
- Qual técnica se comportou melhor em canais com alta taxa de erro?
- Quais são as vantagens e limitações de cada abordagem, conforme os testes realizados?

O que deve ser entregue:

1. Código fonte do programa em Java (bem *indentado* e comentado).
2. Documentação do trabalho. Entre outras coisas, a documentação deve conter:
   2.1. Introdução: descrição do problema a ser resolvido e visão geral sobre o funcionamento do programa.
   2.2. Implementação: descrição sobre a implementação do programa. Deve ser detalhada a estrutura de dados utilizada (de preferência com diagramas ilustrativos), o funcionamento das principais funções e procedimentos utilizados, o formato de entrada e saída de dados, bem como decisões tomadas relativas aos casos e detalhes de especificação que porventura estejam omissos no enunciado.
   2.3. Conclusão: comentários gerais sobre o trabalho e as principais dificuldades encontradas em sua implementação.
   2.4. Bibliografia: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet se for o caso
3. Formato: mandatoriamente em PDF.

Obs1: Apesar desse trabalho ser bem simples, a documentação pedida segue o formato da documentação que deverá ser entregue nos próximos trabalhos.

Obs2: Consulte as dicas do Prof. Nívio Ziviani de como deve ser feita uma boa implementação e documentação de um trabalho prático: https://saulocabral.pagekite.me/roteirotp.pdf

Como deve ser feita a entrega:

A entrega **DEVE** ser feita por email na forma de um único arquivo *zipado*, contendo o código, os arquivos, o executável e a documentação.

Comentários Gerais:

- Comece a fazer este trabalho logo, enquanto o problema está fresco na memória e o prazo para terminá-lo está tão longe quanto jamais poderá estar;
- Clareza, *indentação* e comentários no programa também vão valer pontos;
- O trabalho pode ser feito em duplas (**grupos de MÁXIMO 2 alunos**);
- Todos os grupos e integrantes DEVEM apresentar o trabalho em uma data a ser definida;
- Trabalhos copiados (e **FONTE**) terão nota ZERO;
- Trabalhos entregue em atraso serão aceitos, todavia a nota atribuída ao trabalho será zero
- Evite discussões inócuas com o professor em tentar postergar a data de entrega do referido trabalho.