



Pontifícia Universidade Católica de Minas Gerais  
Instituto de Ciências Exatas e Informática  
Departamento de Ciência da Computação  
Disciplina: Algoritmos e Estruturas de Dados II

# Trabalho Prático II






(Data de Entrega: 04/09)

---

## Regras Básicas

1. Todos os programas serão desenvolvidos nas linguagens Java ou C. Quando não especificada a linguagem, considere que a mesma é Java.
2. Como o combinado não sai caro: As cópias de trabalho serão encaminhadas para o colegiado; O aluno perderá 1 ponto para cada método não comentado ou com comentário inútil.
3. As exceções devem ser perguntadas/discutidas/negociadas com o professor.
4. Na primeira parte deste trabalho, nas questões em Java, você não pode usar os métodos das classes `String`, `Integer`, `Double`, ... Os únicos métodos permitidos são `char charAt(int)` e `int length()` da classe `String`.
5. Para cada questão, devemos submeter apenas um arquivo (.java ou .c). Essa regra será necessária para a submissão de trabalhos no Verde e no identificador de plágio utilizado na disciplina.
6. Na leitura de um número inteiro, se seu valor for vazio ou não número, ele recebe zero.
7. Nos exercícios de ordenação ou estruturas de dados, se dois registros tiverem a mesma chave de pesquisa, eles serão ordenados pelo atributo nome.
8. Para contar as letras, consoantes e vogais desconsidere acentos e cedilha, ou seja, consideramos somente os caracteres cujos códigos ASCII estiverem entre 'A' e 'Z' ou 'a' e 'z'.
9. Para cada exercício: faça (entende-se análise, implementação e comentários), teste (várias vezes) e submeta no Verde. Os exercícios não testados/comentados serão penalizados.
10. A correção será automática através do Verde e de entrevista com o aluno nas aulas de laboratório.
11. A correção sempre será feita na versão do Linux disponível nos laboratórios. Qualquer problema devido ao uso de outras arquiteturas será de responsabilidade do aluno.
12. Fique atento ao **charset** dos arquivos de entrada e saída.

## Questões

1.  **Álgebra Booleana** - Crie um método **iterativo** que recebe uma string contendo uma expressão booleana e o valor de suas entradas e retorna um booleano indicando se a expressão é verdadeira ou falsa. Cada string de entrada é composta por um número inteiro  $n$  indicando o número de entradas da expressão booleana corrente. Em seguida, a string contém  $n$  valores binários (um para cada entrada) e a expressão booleana. Na saída padrão, para cada linha de entrada, escreva uma linha de saída com SIM / NÃO indicando se a expressão corrente é verdadeira ou falsa.
2.  **Is** - Crie um método **iterativo** que recebe uma string e retorna true se a mesma é composta somente por vogais. Crie outro método **iterativo** que recebe uma string e retorna true se a mesma é composta somente por consoantes. Crie um terceiro método **iterativo** que recebe uma string e retorna true se a mesma corresponde a um número inteiro. Crie um quarto método **iterativo** que recebe uma string e retorna true se a mesma corresponde a um número real. Na saída padrão, para cada linha de entrada, escreva outra de saída da seguinte forma X1 X2 X3 X4 onde cada  $X_i$  é um booleano indicando se a entrada é: composta somente por vogais (X1); composta somente por consoantes (X2); um número inteiro (X3); um número real (X4). Se  $X_i$  for verdadeiro, seu valor será SIM, caso contrário, NÃO.
3.  **Inversão de String** - Crie um método **iterativo** que recebe uma string como parâmetro e retorna a string invertida. Na saída padrão, para cada linha de entrada, escreva uma linha de saída com a string invertida. Por exemplo, se a entrada for “abcde”, a saída deve ser “edcba”.
4.  **Soma de Dígitos** - Crie um método **recursivo** que recebe um número inteiro como parâmetro e retorna a soma de seus dígitos. Na saída padrão, para cada linha de entrada, escreva uma linha de saída com o resultado da soma dos dígitos. Por exemplo, se a entrada for 12345, a saída deve ser 15.
5.  **Verificação de Anagrama** - Crie um método **iterativo** que recebe duas strings como parâmetros e retorna true se as strings são anagramas uma da outra, ou false caso contrário. Na saída padrão, para cada par de strings de entrada, escreva uma linha de saída com SIM/NÃO indicando se as strings são anagramas. Por exemplo, se as entradas forem “listen” e “silent”, a saída deve ser SIM.