Pontifícia Universidade Católica de Minas Gerais
Curso de Ciência da Computação
Disciplina: Algoritmos e Estruturas de Dados II
Prof. Felipe Domingos da Cunha / Prof. Max do Val Machado

Trabalho Prático I - Aquecimento

1 Regras Básicas

1. Todos os programas devem ser desenvolvidos na linguagem Java (exceto quando dito o contrário).
2. Todas as apresentações de trabalho devem ser feitas no Linux usando o editor VIM.
3. 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.
4. As exceções devem ser perguntadas/discutidas/negociadas com o professor.
5. Os únicos métodos da classe String permitidos neste trabalho são *char charAt(int)* e *int length()*.
6. O arquivo **readme.txt** contém dicas sobre Linux, VIM e redirecionamento de entrada/saída.
7. 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'.
8. Não utilize as classes IO nem Scanner.
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.

2 Aquecimento

1. Crie um método **iterativo** que receba como parâmetro uma string e retorne seu número de caracteres maiúsculos.

2. Teste o método anterior usando redirecionamento de entrada e saída. A entrada padrão é composta por várias linhas sendo que a última contém a palavra FIM[1]. A saída padrão contém um número inteiro para cada linha de entrada.

3 Descrição

1. **Palíndromo** - Crie um método **iterativo** que recebe uma string como parâmetro e retorna true se essa é um palíndromo. Na saída padrão, para cada linha de entrada, escreva uma linha de saída com SIM / NÃO indicando se a linha é um palíndromo.

2. **Ciframento de César** - O Imperador Júlio César foi um dos principais nomes do Império Romano. Entre suas contribuições, temos um algoritmo de criptografia chamado "Ciframento de César". Segundo os historiadores, César utilizava esse algoritmo para criptografar as mensagens que enviava aos seus generais durante as batalhas. A ideia básica é um simples deslocamento de caracteres. Assim, por exemplo, se a chave utilizada para criptografar as mensagens for 3, todas as ocorrências do caractere 'a' são substituídas pelo caractere 'd', as do 'b' por 'e', e assim sucessivamente. Crie um método **iterativo** que recebe uma string como parâmetro e retorna outra contendo a entrada de forma cifrada. Neste exercício, suponha a chave de ciframento três. Na saída padrão, para cada linha de entrada, escreva uma linha com a mensagem criptografada.

3. **Decifrar mensagem de César** - Dada uma mensagem criptografa com Ciframento de César, o algoritmo para decifrá-la efetua o deslocamento invertido. Assim, por exemplo, se a chave utilizada para criptografar as mensagens tiver sido três, todas as ocorrências do caractere 'd' são substituídas pelo caractere 'a', as do 'e' por 'b' e, assim, sucessivamente. Crie um método **iterativo** que recebe uma string contendo uma mensagem cifrada como parâmetro e retorna outra contendo a entrada de forma decifrada. Neste exercício, suponha a chave de ciframento três. Na saída padrão, para cada linha de entrada (contendo uma mensagem cifrada), escreva uma linha com a mensagem descriptografada.

4. **Inverte** - Crie um método **iterativo** que recebe uma string como parâmetro e retorna outra string correspondendo a entrada invertida. Por exemplo, se a entrada for "Ciência da Computação", a saída será "oãçatupmoC ad aicnêiC". Na saída padrão, para cada linha de entrada, escreva uma linha de saída com a entrada invertida.

[1] A entrada padrão dos demais exercícios é da mesma forma.

5. **Alteração Aleatória** - Crie um método **iterativo** que recebe uma string, sorteia duas letras minúsculas aleatórias (código ASCII $\geq$ 'a' e $\leq$ 'z'), substitui todas as ocorrências da primeira letra na string pela segunda e retorna a string com as alterações efetuadas. Na saída padrão, para cada linha de entrada, execute o método desenvolvido nesta questão e mostre a string retornada como uma linha de saída. Abaixo, observamos um exemplo de entrada supondo que para a primeira linha as letras sorteados foram o 'a' e o 'q'. Para a segunda linha, foram o 'e' e o 'k'.

EXEMPLO DE ENTRADA:	EXEMPLO DE SAÍDA:
o rato roeu a roupa do rei de roma	*o rqto roeu q roupq do rei de romq*
e qwe qwe qwe ewq ewq ewq	*k qwk qwk qwk kwq kwq kwq*
FIM	

A classe Random do Java gera números (ou letras) aleatórios e o exemplo abaixo mostra uma letra minúscula na tela. Em especial, destacamos que: i) *seed* é a semente para geração de números aleatórios; ii) nesta questão, por causa da correção automática, a *seed* será quatro; iii) a disciplina de Estatística e Probabilidade faz uma discussão sobre "aleatório".

```
Random gerador = new Random();
gerador.setSeed(4);
System.out.println((char)('a' + (Math.abs(gerador.nextInt()) % 26)));
```

6. **Equals** - Crie um método **iterativo** que recebe duas strings e retorna true se elas forem iguais e false, caso contrário. Em seguida, crie outro método **iterativo** que recebe duas strings, ignora letras maiúsculas e minúsculas para retornar true se elas forem iguais e false, caso contrário. Na saída padrão, para cada par de linhas de entrada (primeira e segunda, terceira e quarta, quinta e sexta, ...), escreva uma linha de saída da seguinte forma X1 X2, onde $X1, X2 \in \{SIM, NÃO\}$ e ambos indicam se as strings de entrada são iguais. X2 ignora letras maiúsculas e minúsculas. Se o número de linhas de entrada for ímpar, compare a última linha com ela mesma.

7. **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.

8. **Contar** - Crie um método **iterativo** que recebe uma string e um caractere e retorna quantas vezes esse ocorre na string. Crie outro método **iterativo** que recebe uma string e retorna quantas letras existem na string. Crie um terceiro método **iterativo** que recebe uma string e retorna

quantos caracteres diferentes de letra existem na string. Crie um quarto método **iterativo** que recebe uma string e retorna quantos caracteres doidões existem na string. Consideramos que um caractere é doidão quando ele é um dígito ímpar ou consoante cujo código ASCII é múltiplo de cinco e não múltiplo de dois ou vogal cujo código ASCII não é múltiplo de cinco nem de oito. Na saída padrão, para cada linha de entrada, escreva outra de saída da seguinte forma X1 X2 X3 X4, onde X1 é o número de ocorrências do primeiro caractere da linha na mesma, X2 é o número de letras, X3 é o número de caracteres não letra, X4 é o número de caracteres doidões.

9. **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 Xi é um booleano indicando se a é entrada é: composta somente por vogais (X1); composta somente somente por consoantes (X2); um número inteiro (X3); um número real (X4). Se Xi for verdadeiro, seu valor será SIM, caso contrário, NÃO.

10. **Conversão** - Crie um método **iterativo** que recebe uma string e retorna a mesma convertendo todos os caracteres minúsculas em maiúsculos. Crie outro método **iterativo** que recebe uma string e retorna a mesma convertendo todos os caracteres maiúsculos em minúsculas. Crie um terceiro método **iterativo** que recebe uma string e retorna a mesma removendo todos seus espaços. Crie um quarto método **iterativo** que recebe uma string e um caractere e retorna a string removendo todas as ocorrências do caractere. Na saída padrão, para cada linha de entrada, escreva quatro linhas de saída. A primeira terá todos os caracteres minúsculos convertidos em maiúsculos. A segunda terá os maiúsculos convertidos em minúsculos. A terceira terá os mesmos caracteres da entrada removidos os espaços. A quarta terá os mesmos caracteres da entrada removidos os caracteres iguais ao primeiro (inclusive).

11. **Palíndromo em C** - Na linguagem C, crie um método **iterativo** que recebe uma string como parâmetro e retorna true se essa é um palíndromo. Na saída padrão, para cada linha de entrada, escreva uma linha de saída com SIM / NÃO indicando se a linha é um palíndromo.

    Observação: Uma linha de entrada pode ter caracteres não letras.