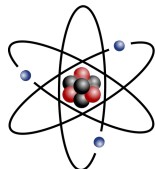




2ª Avaliação



O átomo é a unidade fundamental da matéria e a menor fração capaz de identificar um elemento químico. Em nível microscópico, pode-se afirmar que um elemento químico é o conjunto de átomos que possui o mesmo número atômico. Já uma molécula é um conjunto de átomos – iguais ou diferentes – unidos por ligações covalentes.

Com base nesses conceitos elementares da Química, desenvolva uma aplicação em Dart que implemente os seguintes requisitos e funcionalidades:

REQUISITOS E FUNCIONALIDADES:

1. Implemente a classe “*Element*” para representar um único elemento químico da Tabela Periódica.
 - A classe deverá possuir os seguintes atributos e respectivos tipos: *symbol* (String), *name* (String), *latinName* (String) e *weight* (int).
 - Os atributos da classe não poderão ser alterados depois de inicializados.
 - O construtor deverá levantar uma exceção do tipo *Exception* caso o símbolo seja inválido.
2. Implemente a classe “*PeriodicTable*” que deverá armazenar os dados de todos os elementos químicos da Tabela Periódica. Os dados dos elementos deverão ser carregados a partir de um arquivo no momento da instanciação da classe.
 - Na seção “Recursos do Projeto” apresentada adiante, será disponibilizado um repositório que contém um arquivo JSON com os dados de todos os elementos químicos da Tabela Periódica.
 - A classe “*PeriodicTable*” deverá ser um singleton utilizado internamente pelas classes do projeto. O padrão “singleton” determina que uma classe tenha apenas uma única instância alocada em toda a aplicação.
3. Implemente a classe “*Atom*” que representa um único átomo, sabendo que este é definido pelo seu símbolo (*symbol*) na Tabela Periódica. Por exemplo: “H”, “O”, “C”, “Cl”, “Ag”, etc.
 - O símbolo do átomo deverá ser o único parâmetro do construtor da classe “*Atom*”. Se o símbolo for inválido, a classe deverá levantar uma exceção.
 - Os atributos da classe não poderão ser alterados depois de inicializados.
4. O símbolo do elemento deverá ser a string de representação de uma instância da classe “*Atom*”.
5. Defina a classe “*Molecule*” que representa uma molécula, sabendo que esta é definida por sua fórmula molecular. Por exemplo, a fórmula molecular da água é “H₂O”, composta por 2 átomos de hidrogênio (“H”) e 1 átomo de oxigênio (“O”).
 - A fórmula molecular e seu nome deverão ser os únicos parâmetros do construtor da classe “*Molecule*”. No caso de uma fórmula inválida, uma exceção deverá ser levantada.
 - Os atributos fórmula e nome deverão ter acesso público.
 - Para simplificar o problema, serão consideradas apenas fórmulas com letras e dígitos numéricos. Não serão avaliadas fórmulas complexas como “(NH₄)₂SO₄” ou “Na₂[B₄O₅(OH)₄]·8H₂O”.
 - Exemplos de fórmulas: “O₂”, “H₂O”, “NaCl”, “H₂SO₄”, “C₆H₁₂O₆”, etc.

6. A classe “*Molecule*” deverá implementar um *getter* “*weight*” que retorna o peso atômico da molécula. O peso atômico da molécula é dado pela soma dos pesos atômicos de todos os átomos que compõem a molécula.
7. A classe “*Molecule*” deverá implementar a interface genérica “*Comparable<T>*”, que permitirá que instâncias de “*Molecule*” possam ser comparadas umas com as outras. O parâmetro de comparação utilizado deverá ser o peso atômico da molécula representada pela instância.
 - A interface *Comparable<T>* é uma classe abstrata já existente no SDK do Dart. Veja a documentação da interface *Comparable<T>* em api.dart.dev.
8. Seu código será testado com a função *main()* existente no arquivo “*main.dart*”, que poderá ser baixado do repositório GitHub disponibilizado na seção “Recursos do Projeto” apresentada adiante.
 - O código do arquivo “*main.dart*” não deverá ser alterado. Seu código deve se adaptar às chamadas lá existentes.
 - Seu código deverá se adaptar às instruções existentes no arquivo “*main.dart*”.
 - São permitidas alterações apenas em instruções relativas à importação de módulos.
9. Qualidade e publicação do código.
 - Toda a codificação deverá ser realizada em língua inglesa, incluindo identificadores e strings de mensagens.
 - A codificação do projeto deverá observar os princípios de estilo *Effective Dart* (<https://dart.dev/effective-dart>).
 - O projeto deverá ser disponibilizado em um repositório com acesso público do GitHub, com “*main*” como seu branch principal (default do GitHub). Qualquer outro branch não será considerado.

RECURSOS DO PROJETO:

1. Todos os arquivos necessários para o projeto estão disponíveis no repositório <https://github.com/uespi-phb/chemical.git>
2. Os arquivos disponibilizados não deverão ser alterados.

ARQUIVO	DESCRIÇÃO
main.dart	Arquivo principal para teste do código submetido.
elements.json	Arquivo em formato JSON que contém os dados dos elementos químicos.
output.txt	Saída esperada ao executar o programa a partir do arquivo “main.dart”

COMPOSIÇÃO DA EQUIPE:

1. A implementação poderá ser realizada em equipes de ATÉ 2 (dois) membros;
2. Os autores de cada implementação poderão ser questionados sobre o código implementado, com o objetivo de comprovar a participação de cada membro na execução do projeto;

INTRUÇÕES PARA REMESSA DO PROJETO:

1. Ao finalizar o projeto, remeter o *link* do repositório GitHub no sistema SIGAA.
2. Prazo de entrega: **24/11/2023** (este prazo não será prorrogado).
3. O projeto deverá ser compatível com a versão 3 do Dart.

CRITÉRIOS DE AVALIAÇÃO:

- Cada requisito/funcionalidade será avaliado individualmente e receberá uma das seguintes notas:

Nota	Grau de atendimento ao requisito/funcionalidade
0	Não atende
1 a 4	Atende parcialmente
5	Atende completamente

- A nota final será determinada pela média ponderada das notas de cada requisito/funcionalidade.