Pontifícia Universidade Católica do Rio Grande do Sul
Programação Orientada a Objetos
Prof. Marcelo H. Yamaguti
2022/2

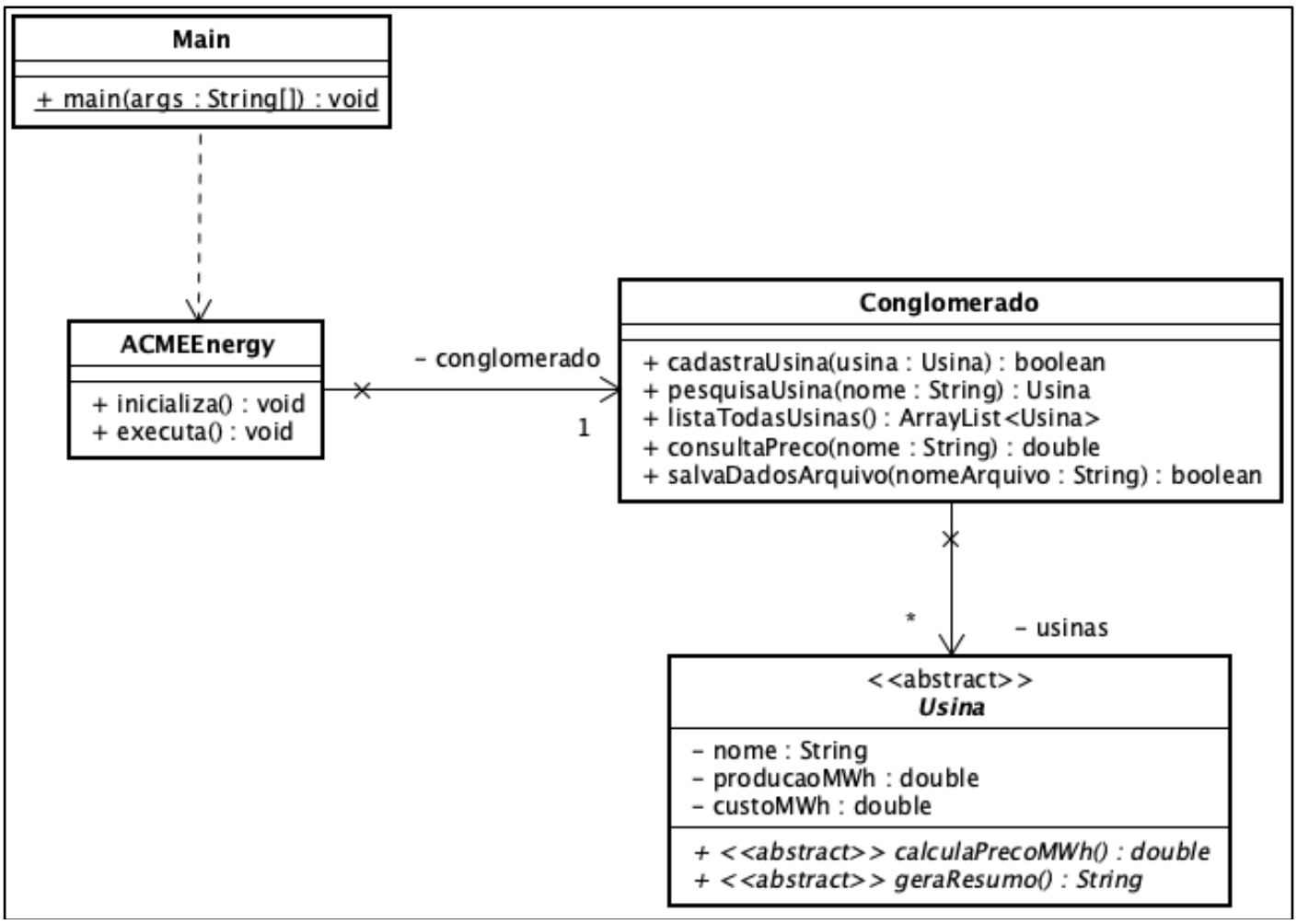
Exercício de Avaliação 2

1. **Enunciado geral:**

A ACMEEnergy gerencia um conglomerado de diversas usinas geradoras de energia elétrica e quer automatizar o controle de usinas.

Você será responsável pelo desenvolvimento da aplicação.

O analista de sistemas gerou um diagrama de classes inicial, com alguns atributos, operações e relacionamentos apresentados a seguir.



O analista identificou operações básicas da classe **Conglomerado**, que gerencia uma coleção de usinas:

- **cadastraUsina(Usina)**: adiciona uma nova usina no cadastro (não pode haver duas usinas com mesmo nome). Retorna *true* se a usina foi adicionada, ou *false* caso contrário.
- **pesquisaUsina(String)**: retorna a usina com o nome indicado, ou *null* se nenhuma usina foi encontrada.
- **listaTodasUsinas()**: retorna uma coleção de todas as usinas cadastradas, ou *null* se não há usinas cadastradas.
- **consultaPreco(String)**: retorna o valor do preço do MWh (MegaWatt-Hora) da usina com o nome indicado, ou -1.0 se nenhuma usina foi encontrada.

- **salvaDadosArquivo(String)**: salva os dados de todas as usinas cadastradas em um arquivo-texto em formato CSV com o nome de arquivo indicado.

Sabe-se que será necessário haver subclasses de ***Usina***. Cada subclasse possui informações adicionais específicas:

- **Usina de energia renovável**: possui uma fonte de energia (que pode ser: solar, eólica ou hídrica).
- **Usina de energia não-renovável**: possui um combustível (que pode ser: petróleo, carvão ou nuclear) e uma durabilidade do combustível (que é medida em anos).

O método ***calculaPrecoMWh()*** retorna o valor do preço do MWh que depende da subclasse:

- **Usina de energia renovável**: conforme a fonte de energia: custo MWh acrescido em 25% se for solar, 15% de for eólica ou 5% se for hídrica.
- **Usina de energia não-renovável**: conforme o combustível: custo MWh acrescido em 30% se for petróleo, 20% ser for carvão ou 10% se for nuclear.

O método ***geraResumo()*** gera uma String no formato CSV, que inicia com "1" se for Usina de energia renovável ou "2" se for Usina de energia não-renovável, seguido dos valores dos atributos do objeto separados por ";". Exemplo: um objeto da subclasse Usina de energia renovável, com nome "Lapa", com produção de 158,7 MWh, custo por MWh de R$ 2,00 e fonte solar, geraria a String: "1;Lapa;158.7;2.00;solar".

O método **inicializa()** da classe ACMEEnergy cadastra alguns objetos no início da execução da aplicação, que deve ter cadastramento correto de, pelo menos:

- 1 objeto diferente de cada subclasse.
- 2 objetos com mesma produção e custo MWh mas com nomes diferentes.

O método **executa()** da classe ACMEEnergy deve apresentar e executar uma tela cíclica com o usuário com as opções de:

0. **Sair**: termina a execução do sistema.
1. **Cadastrar nova usina**: solicita ao usuário os dados de uma nova usina e adiciona-a no cadastro. Apresenta a mensagem "Usina repetida." se o nome já existe.
2. **Pesquisar uma usina**: solicita ao usuário o nome de uma usina e apresenta os seus dados completos (incluindo o preço do MWh); se não encontrar apresenta a mensagem de erro: "Nenhuma usina localizada com este nome."
3. **Listas todas as usinas**: mostra a descrição de todas as usinas cadastradas, com os respectivos preços de MWh. Se não houver usinas cadastradas apresenta a mensagem de erro: "Nenhuma usina cadastrada."
4. **Consulta o preço do MWh**: solicita ao usuário o nome de uma usina, e apresenta o preço do MWh; se não encontrar uma usina apresenta a mensagem de erro: "Nenhuma usina foi localizada com este nome."
5. **Salvar usinas em arquivo**: solicita ao usuário o nome do arquivo (sem extensão) para salvamento de dados. Os dados das usinas do cadastro são guardados em um arquivo de extensão .CSV, conforme descrito no método geraResumo().

2. Definição do exercício:

O objetivo do exercício é implementar um sistema capaz de atender as necessidades da empresa descrita no enunciado geral, e que atenda as restrições que a seguir:

- É permitida a criação de novas classes, atributos e métodos; mas os relacionamentos, atributos e métodos definidos no diagrama de classes original não podem ser alterados.

- Toda entrada e saída de dados com o usuário deve ocorrer apenas na classe ACMEEnergy. Deve haver tratamento de exceções para que a aplicação não falhe na execução.
- O diagrama de classes deve ser atualizado conforme as alterações realizadas e deve ser entregue em arquivo Astah ou PDF.

3. Critérios de avaliação

O exercício será avaliado conforme os seguintes critérios:

- Diagrama de classes atualizado: 1 ponto.
- Uso de herança: 2 pontos.
- Uso de polimorfismo: 1 ponto.
- Tratamento de exceções: 1 ponto.
- Salvamento de dados em arquivo-texto: 1 ponto
- Implementação correta conforme a descrição do exercício e o diagrama de classes: 2 pontos.
- Execução correta das opções previstas na tela do usuário: 2 pontos.

- *Ponto extra (opcional): implementar uma opção adicional para que o usuário possa 'Carregar usinas de arquivo – o usuário digita o nome do arquivo de dados e o sistema carregar os dados do arquivo no cadastro: 1 ponto (máximo de 10 pontos).*

4. **Entrega:**

Data de entrega: 3 / 10 / 2022.

A entrega do exercício envolverá:

- arquivos dos códigos-fonte do sistema (e demais arquivos necessários para a compilação do sistema).
- diagrama de classes atualizado.

Deverá ser gerado um arquivo compactado (.zip ou .rar), com os itens acima, e entregue na tarefa da área Moodle da disciplina.

5. **Considerações finais:**

- O exercício deve ser desenvolvido individualmente.
- A implementação deve seguir o Java Code Conventions para nomes de identificadores e estruturas das classes.
- Não será aceito exercício com erros de compilação. Programas que não compilarem corretamente terão nota zerada.
- A cópia parcial ou completa do exercício terá como consequência a atribuição de nota 0 (zero) aos exercícios dos alunos envolvidos. Para análise de similaridade será utilizado o MOSS (https://theory.stanford.edu/~aiken/moss/).