Programação Orientada a Objetos

Exercício 2 – Ranking de Pontuação

O seguinte exercício pode ser realizado em grupos de no máximo 4 integrantes.

A entrega de todo o código-fonte do projeto deve ser realizado até o dia 15/05/2023 no Moodle.

A atividade consiste em realizar o exercício descrito a seguir e entregar todo o código-fonte do projeto em um arquivo zipado.

Todos os jogos desenvolvidos por uma determinada empresa de jogos de computador são capazes de armazenar um determinado número de melhores scores. Para tanto a empresa definiu a interface *Scores* e a classe *GameEntry*.

A classe *GameEntry* armazena uma entrada de lista de scores, ou seja, o nome do jogador, o score que ele atingiu e a data que aquele score foi obtido.

A interface *Scores* define os métodos que uma classe que armazena os melhores scores deve implementar. A coleção de scores deve ser sempre mantida ordenada de forma decrescente de valores do score; assim, os métodos que alteram essa coleção devem manter a ordem de forma coerente. A partir do momento em que a coleção estiver cheia, sempre que um novo "melhor score" tiver de ser inserido o menor score armazenado terá de ser removido. Isso implica, também, que a partir do momento em que a coleção estiver cheia, apenas scores maiores que os já armazenados poderão ser armazenados.

```
public class GameEntry {
  public GameEntry(String n, int s, LocalDate d)…
  public String getName()…
  public int getScore()…
  public LocalDate getDate()…
  /** Retorna uma string representando o objeto
   *  Formato: (<name>, <score>, <date>)
   *  Exemplo: (John, 10, 10/04/2023)
   */
  public String toString()…
}
```

```
public interface Scores {
 /** Retorna uma string representando o objeto
  * Formato: [(<name>, <score>, <date>), (<name>, <score>, <date>) …]
  * Exemplo: [(John, 10, 10/04/2023), (Carol, 5, 01/02/2022)]
  */
 String toString();
 /** Adiciona um novo score se ele for grande o suficiente
```

```
   * Retorna verdadeiro se foi adicionado, falso caso contrário
   */
  boolean add(GameEntry e);
  /** Retorna o score na posição i */
  GameEntry get(int i);
  /** Retorna a capacidade máxima da coleção */
  int getCapacity();
  /** Retorna o número de scores armazenados */
  int getNumScores();
  /** Retorna a média dos scores armazenados */
  double getAvgScores();
}
```

A partir das definições apresentadas, realize as seguintes tarefas:

a) Implemente a classe *GameEntry*.

b) Construa uma classe que implementa a interface *Scores* de maneira adequada. Lembre-se que a coleção deve possuir um número máximo de scores armazenados e tal informação deve ser indicada via o construtor da classe.

c) Implemente uma classe utilitária *ScoresSerializacao* com métodos para ler e escrever os dados da coleção de scores em arquivos com o seguinte formato:

- Arquivo texto CSV (“comma separated values”, https://pt.wikipedia.org/wiki/Comma-separated_values).

Para todos os casos defina corretamente um mecanismo de tratamento das exceções e escreva um pequeno programa para testar a implementação de suas classes.