Programação II

Manipulação de Arquivos em Java

RAFAEL VIEIRA COELHO

Relembrando…

Tarefas

1) No ProgramaPrincipal, altere todos os dados fixos para atributos estáticos e finais.

2) Crie os métodos equals e toString das classes Professor e SetorEnsino.

3) Modifique o ProgramaPrincipal para que use os métodos criados no exercício 2 e faça testes para corrigir possíveis problemas (um erro comum é não testar se o objeto ou o array contém valor nulo, gerando assim a exceção NullPointerException).

```
    public static void main(String[] args) throws IOException {
        Aluno[] alunos = new Aluno[TOTAL_ALUNOS];
        SetorEnsino ensino = new SetorEnsino(DIRETOR_ENSINO, COORDENADOR_ENSINO);
        int opcao = 4;

        do {
            opcao = menu("MENU 1: \n "
                    + "[" + OP_ALUNO + "] Aluno \n "
                    + "[" + OP_PROFESSOR + "] Professor \n "
                    + "[" + OP_ENSINO + "] Setor de Ensino \n "
                    + "[" + OP_SAIR + "] Sair \n",
                    br);
            switch (opcao) {
                case OP_ALUNO:
                    menu_alunos("MENU 2: \n "
                            + "[" + OP_ALUNO_VER_CURSOS + "] Ver Cursos \n"
                            + "[" + OP_ALUNO_VER_NOTAS + "] Ver notas \n"
                            + "[" + OP_ALUNO_VOLTAR + "] Voltar",
                            ensino,
                            alunos,
                            br);

                    break;
                case OP_PROFESSOR:
                    System.out.println("Qual o seu número de siape, professor?");
                    int siape = Integer.parseInt(br.readLine());
                    int posicao_professor = login_professor(siape, ensino, br);

                    if (posicao_professor != POSICAO_INVALIDA) {
                        menu_professor("MENU 2: \n "
```

```
                int posicao_professor = login_professor(siape, ensino, br);

                if (posicao_professor != POSICAO_INVALIDA) {
                    menu_professor("MENU 2: \n "
                            + "[" + OP_PROFESSOR_DAR_NOTAS + "] Dar Notas de uma di
                            + "[" + OP_PROFESSOR_ALTERAR_NOTA + "] Alterar uma nota
                            + "[" + OP_PROFESSOR_ADICIONAR_AREA + "] Adicionar Área
                            + "[" + OP_PROFESSOR_REMOVER_AREA + "] Remover Área \n"
                            + "[" + OP_PROFESSOR_VOLTAR + "] Voltar",
                            posicao_professor,
                            ensino,
                            br
                    );
                } else {
                    System.out.println("SIAPE inválido.");
                }
                break;
            case OP_ENSINO:
                menu_ensino("MENU 2: \n "
                        + "[" + OP_ENSINO_NOVO_ALUNO + "] Cadastrar Aluno \n "
                        + "[" + OP_ENSINO_NOVO_CURSO + "] Cadastrar Curso \n "
                        + "[" + OP_ENSINO_NOVA_DISCIPLINA + "] Adicionar Disciplina
                        + "[" + OP_ENSINO_NOVO_PROFESSOR + "] Cadastrar Professor \
                        + "[" + OP_ENSINO_VOLTAR + "] Voltar",
                        ensino,
                        alunos,
                        br);

        }
    } while (opcao != OP_SAIR);
```

Relembrando…

```
////////////////// ALUNO ////////////////////////////////////////
private static void menu_alunos(String opcoes, SetorEnsino ensino, Aluno alunos[],
    int opcao = menu(opcoes, br);

    switch (opcao) {
        case OP_ALUNO_VER_CURSOS:
            ver_cursos(ensino);
            break;
        case OP_ALUNO_VER_NOTAS:
            System.out.println("Qual a sua matrícula, caro discente?");
            long matricula = Long.parseLong(br.readLine());

            ver_notas(ensino, alunos, matricula);
            break;
    }
}
```

Relembrando…

Tarefas

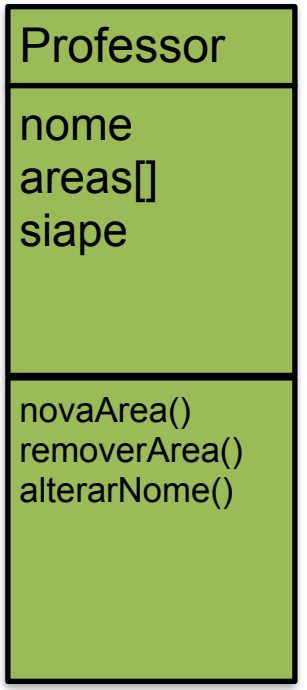
1) No ProgramaPrincipal, altere todos os dados fixos para atributos estáticos e finais.

2) Crie os métodos equals e toString das classes Professor e SetorEnsino.

3) Modifique o ProgramaPrincipal para que use os métodos criados no exercício 2 e faça testes para corrigir possíveis problemas (um erro comum é não testar se o objeto ou o array contém valor nulo, gerando assim a exceção NullPointerException).

Relembrando…

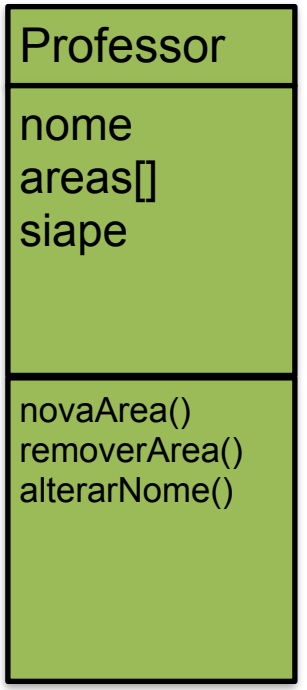
```
@Override
public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (obj == null) {
        return false;
    }
    if (getClass() != obj.getClass()) {
        return false;
    }
    final Professor other = (Professor) obj;
    if (this.siape != other.siape) {
        return false;
    }
    return Objects.equals(this.nome, other.nome);
}
```



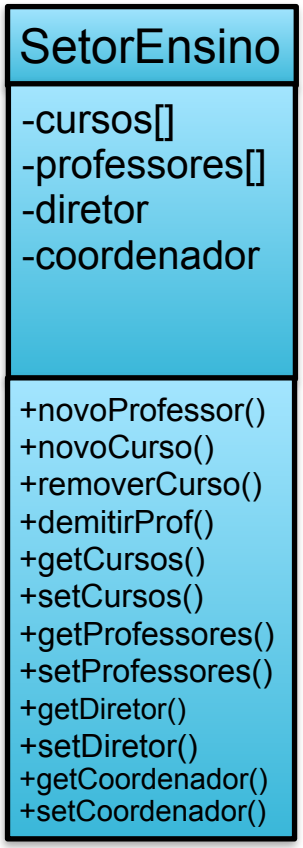
Relembrando…

```
@Override
public String toString() {
    String nomes_areas = "";

    for (String d : areas) {
        nomes_areas += d + "\n"
    }
    return "\n nome: "
            + nome
            + "\n siape: "
            + siape
            + "\n areas: "
            + nomes_areas;
}
```



Relembrando…



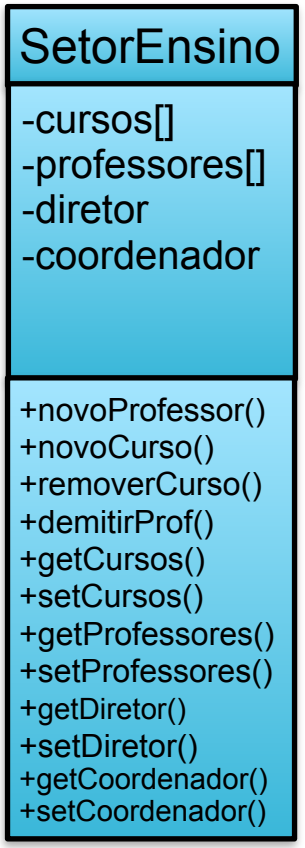

```
@Override
public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (obj == null) {
        return false;
    }
    if (getClass() != obj.getClass()) {
        return false;
    }
    final SetorEnsino other = (SetorEnsino) obj;
    if (!Objects.equals(this.diretor, other.diretor)) {
        return false;
    }
    return (Objects.equals(this.coordenador, other.coordenador));
}
```

Relembrando…

```
@Override
public String toString() {
    String nomes_cursos = "";

    for (Curso d : cursos) {
        nomes_cursos += d.toString() + "\n";
    }
    String nomes_professores = "";

    for (Professor e : professores) {
        nomes_professores += e.toString() + "\n";
    }
    return "\n cursos: "
            + nomes_cursos
            + "\n professores: "
            + nomes_professores
            + "\n diretor: "
            + diretor
            + "\n coordenador: " + coordenador;
}
```



Relembrando…

Tarefas

1) No ProgramaPrincipal, altere todos os dados fixos para atributos estáticos e finais.

2) Crie os métodos equals e toString das classes Professor e SetorEnsino.

3) Modifique o ProgramaPrincipal para que use os métodos criados no exercício 2 e faça testes para corrigir possíveis problemas (um erro comum é não testar se o objeto ou o array contém valor nulo, gerando assim a exceção NullPointerException).

Relembrando…por exemplo

```
if (disciplina.getNotas() != null) {
    float nota = disciplina.getNotas()[i];

    System.out.println("A nota do aluno "
            + a[i].toString()
            + " é de "
            + nota
            + " na disciplina "
            + disciplina.toString());
}
```

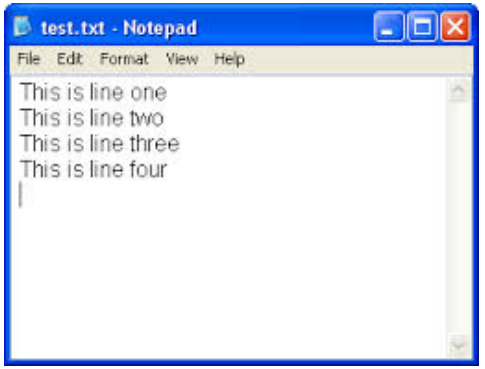
```
private static void ver_cursos(SetorEnsino ensino) {
    Curso cursos[] = ensino.getCursos();

    if (cursos != null) {
        for (Curso curso : cursos) {
            if (curso != null) {
                System.out.println(curso.toString());
            }
        }
    }
}
```

Tipos de Arquivos

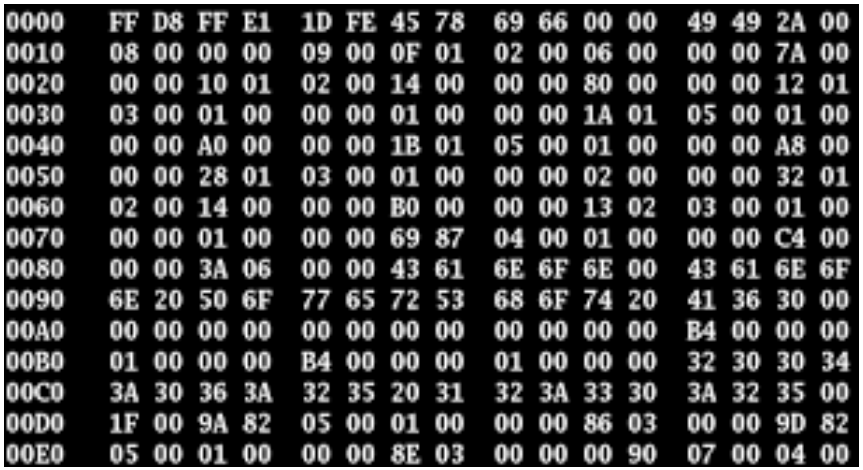
Os arquivos podem ser classificados em arquivos de texto ou arquivos binários

• **Arquivos de texto:** são compostos por uma série de caracteres ascii agrupados em uma ou mais linhas. São compreendidos pelos seres humanos.



•**Arquivos binários:** composto por uma série de bytes representados por caracteres não compreendidos pelo ser humano.

Ex.: imagens, vídeo, áudio, etc.



SUMÁRIO

Classe File: Métodos

- Representa um arquivo ou diretório
  File f = new File("log.txt");

- boolean canRead() -> retorna true se for possível ler o arquivo, falso o contrário
- boolean canWrite() -> retorna true se for possível escrever no arquivo, falso o contrário
- boolean exists() -> retorna true se o diretório ou arquivo se o objeto File existe, falso o contrário
- boolean isFile() -> retorna true se é um arquivo, falso o contrário
- boolean isDirectory() -> retorna true se é um diretório, falso o contrário
- String getAbsolutePath() -> retorna uma String com o caminho absoluto do diretório ou arquivo
- String getName() -> retorna uma String com o nome do arquivo ou do diretório
- String getPath() -> retorna uma Sting com o caminho do arquivo ou diretório
- String getParent() -> retorna uma String com o caminho do diretório pai (acima;anterior)
- long length() -> retorna um tamanho, em bytes, do arquivo ou inexistente, caso seja diretório
- long lastModified() -> retorna o tempo em que o arquivo/diretório foi modificado pela última vez
- String[] list() -> retorna um array de Strings com o conteúdo do diretório, ou null se for arquivo

Classe File: Exemplo

```
//criando um arquivo em um diretório específico
File folder = new File("C:/");
File fileInFolder = new File(folder, "io.txt");
fileInFolder.createNewFile();

//obtendo o caminho de um arquivo
Path pathFromFile = new File("io.txt").toPath();

//verificando se um arquivo existe
if (file.exists()) {
    //apagando um arquivo
    boolean deleted = file.delete();

    if (deleted) {
          System.out.println(“Arquivo apagado.”);
   }
}
```

SUMÁRIO

Formas de Escrita em um Arquivo

1) ObjectOutputStream (binário)

2) PrintWriter (texto)

Classe utilizada nos exemplos

```
public class Pessoa implements Serializable {
    private String nome;
    private int idade;
    private String sexo;

    public Pessoa(String nome, int idade, String sexo) {
        this.nome = nome;
        this.idade = idade;
        this.sexo = sexo;
    }

    public Pessoa() {
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public int getIdade() {
        return idade;
    }

    public void setIdade(int idade) {
        this.idade = idade;
    }
```

Forma 1) ObjectOutputStream

```
public class EscrevendoArquivoBinario {

    public static void main(String[] args) {
        Pessoa ze = new Pessoa("Zé", 29, "Masculino");
        Pessoa maria = new Pessoa("Maria", 34, "Feminino");

        try {
            File f = new File("arquivo.bin");
            if (!f.exists()) {
                f.createNewFile();
            }
            try (FileOutputStream fos = new FileOutputStream(f);
                    ObjectOutputStream oos = new ObjectOutputStream(fos)) {
                oos.writeObject(maria);
                oos.writeObject(ze);
            }
            System.out.println("Dados salvos com sucesso: ");
            System.out.println(maria.toString());
            System.out.println(ze.toString());
        } catch (IOException e) {
            System.err.println("Erro ao criar o arquivo arquivo.bin.");
        }
    }
}
```

Forma 2) PrintWriter

```
public class EscrevendoArquivoTexto {

    public static void main(String[] args) {
        Pessoa ze = new Pessoa("Zé", 29, "Masculino");
        Pessoa maria = new Pessoa("Maria", 34, "Feminino");

        try {
            File f = new File("arquivo.txt");
            if (!f.exists()) {
                f.createNewFile();
            }
            try (FileWriter fw = new FileWriter(f);
                    BufferedWriter bw = new BufferedWriter(fw);
                    PrintWriter pw = new PrintWriter(bw)) {
                pw.println(maria.toString());
                pw.println(ze.toString());
            }
            System.out.println("Dados salvos com sucesso: ");
            System.out.println(maria.toString());
            System.out.println(ze.toString());
        } catch (IOException e) {
            System.err.println("Erro ao criar o arquivo arquivo.txt.");
        }
    }
}
```

SUMÁRIO

Formas de Leitura de um Arquivo

1. ObjectInputStream (binário)

2. BufferedReader (texto)

Forma 1) ObjectInputStream

```
public class LendoArquivoBinario {

    public static void main(String[] args) {
        Pessoa maria, ze;

        try {
            File f = new File("arquivo.bin");
            if (f.exists()) {

                try (FileInputStream fis = new FileInputStream(f);
                        ObjectInputStream ois = new ObjectInputStream(fis)) {
                    maria = (Pessoa) ois.readObject();
                    ze = (Pessoa) ois.readObject();
                    System.out.println("Dados lidos: ");
                    System.out.println(maria.toString());
                    System.out.println(ze.toString());
                } catch (ClassNotFoundException ex) {
                    System.err.println("Erro ao ler objeto de arquivo.bin.");
                }
            }
        } catch (IOException e) {
            System.err.println("Erro ao abrir o arquivo arquivo.bin.");
        }
    }
}
```

Forma 2) BufferedReader

```
16
17      public class LendoArquivoTexto {
18
19          public static void main(String[] args) {
20              Pessoa maria, ze;
21
22              try {
23                  File f = new File("arquivo.txt");
24                  if (f.exists()) {
25
26                      try (FileReader fr = new FileReader(f);
27                              BufferedReader br = new BufferedReader(fr)) {
28                          String texto_lido = "",
29                                  linha_lida = "";
30
31                          do {
32                              linha_lida = br.readLine();
33                              if (linha_lida != null) {
34                                  texto_lido += linha_lida + System.lineSeparator();
35                              }
36                          } while (linha_lida != null);
37                          System.out.println("Dados lidos: ");
38                          System.out.println(texto_lido);
39                      }
40                  }
41              } catch (IOException e) {
42                  System.err.println("Erro ao abrir o arquivo arquivo.bin.");
43              }
          }
      }
```

Observação Aleatória

Podemos utilizar o BufferedReader para ler do teclado:

```
    BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
    String line = br.readLine();
```

Adeus Scanner!!

Observação Aleatória

Como gerar um número randômico?

```
Random random = new Random();
int number = random.nextInt(100);//0 - 99
```

Dica Aleatório

Como obter a data e o horário corrente do sistema operacional?

```
String var = String.valueOf(Calendar.getInstance().getTime());
System.out.println(var);
//Fri Jun 23 14:41:15 BRT 2017

System.out.printf("Today is: %1$tm/%1$td/%1$tY", LocalDate.now());
//Today is: 06/23/2017
```

Tarefas

1) Modifique o projeto no qual estamos trabalhando para que seja escrito em um arquivo de *log* (log.txt) sempre que ocorrer um erro no programa. Ou seja, sempre que entrar em um *catch*.

2) Modifique o programa principal para que sejam abertos no início e salvos no final de cada execução os dados acadêmicos. Para isto, precisam ser salvos a lista de alunos (dados do *array* alunos) e os dados do setor de ensino (encapsulados no objeto ensino). Use um arquivo binário chamado ifrs.bin para armazenar os dados dos objetos.