Exercício 1

Implemente um programa no qual o usuário deverá informar o nome e a idade de três pessoas. O programa deverá informar o nome da pessoa que possuir a maior idade.

Regras que deverão ser seguidas para a implementação do algoritmo:

- É obrigatório o uso de classe para representar uma pessoa e a mesma deverá possuir como propriedades (características) um nome e uma idade.
- A classe também deverá possuir um método chamado ExibirDados. Esse método deverá exibir o nome e a idade da pessoa em questão.
- Ao implementar a classe é obrigatório implementar dois construtores (Sobrecarga), um que não recebe parâmetro algum e outro que irá receber o nome e a idade de uma pessoa.

Exercício 2

Crie um programa que exiba para o usuário qual é a área de um retângulo. Para implementar esse programa você deverá seguir as seguintes regras:

- O usuário deverá informar para o programa o valor da base e altura do retângulo/quadrado.
- É obrigatório criar/utilizar uma classe para representar o retângulo/quadrado.
- A base e a altura informada pelo usuário deveram ser representadas na classe como propriedades.
- O objeto deverá possuir uma propriedade para exibir o valor da área do retângulo/quadrado
- O objeto deverá possuir um método que exiba os dados de todas as suas propriedades.

Exercício 3

Crie um algoritmo/programa no qual o usuário deverá informar o nome e o tipo de cinco animais de estimação. O programa deverá exibir na tela para o usuário quantos Cachorros, Gatos e peixes foram informados.

Regras que deverão ser seguidas para a implementação do algoritmo:

- Os únicos tipos de animais válidos são Cachorro, Gato, Peixe.
- Caso seja informado um tipo diferente o programa deverá definir o tipo do animal como Peixe.
- É obrigatório criar uma classe para representar o Animal.

- A classe deverá possuir dois dados privados (propriedades) para representar as características do animal.
- A classe deverá possuir métodos de acesso para permitir que o usuário armazene/leia os dados dos dois dados privados (propriedades).

Exercício 4

Crie a classe Jogador descrita no “diagrama” abaixo:

a)Cadastrar/Instanciar pelo menos um time completo 11 jogadores.

b) Criar um método que verifica a condição de jogo, ou seja, um método booleano que retornará true se o jogador está apto a jogar e false se o jogador está suspenso. Note que um jogador está suspenso pelo 3 cartão amarelo ou quando recebe uma punição cartão vermelho ou em uma decisão do tribunal.

c) Em uma outra classe, crie o método main, o qual cadastra os jogadores e ao final imprimirá a lista do “plantel” juntamente com a informação de quem está apto a jogar, conforme a figura abaixo.

```
JOGADORES CADASTRADOS
Goleiro: 1 - Marcelo Grohe (Muralha Tricolor) - 13/jan/1987 CONDIÇÃO: SUSPENSO
Lateral Direito: 2 - Marcelo Hermes (Hermes) - 01/fev/1995 CONDIÇÃO: TÁ PRA JOGO
```

Crie novos métodos na classe Jogador:

• aplicarCartao(int quantidade): void - Aplica a quantidade de cartões informada ao jogador, adicionalmente pode tornar um jogador suspenso.

• cumprirSuspencao(): void – Esse método vai zerar a quantidade de cartões e tornar o jogador apto a jogar

• sofrerLesao(): void – Este método vai gerar aleatoriamente um lesão no jogador. A gravidade da lesão irá se refletir em uma redução da qualidade do jogador, quanto mais grave maior a redução da qualidade. Crie uma escala de redução de no mínimo 1 ponto até o máximo de 15% da qualidade total do jogador. Note que a qualidade jamais pode ficar negativa. A tabela abaixo pode ser utilizada como referência:

Probabilidade	Qualidade decrementada
5%	15% do total da qualidade
10%	10% do total da qualidade
15%	5% do total da qualidade
30%	2 pontos
40%	1 ponto

• executarTreinamento(): void – A exemplo do método anterior, este método aleatoriamente vai aumentar a qualidade do jogador. Note que só pode ser executado 1 treinamento antes de cada partida (você deve adicionar um atributo na classe para poder controlar essa informação). Além disso, deve existir uma maior probabilidade para pequenos incrementos na qualidade conforme a seguinte tabela:

Probabilidade	Qualidade incrementada
5%	5 pontos
10%	4 pontos
15%	3 pontos
30%	2 pontos
40%	1 ponto

Observe que a qualidade nunca pode superar o máximo de 100 pontos. Além disso, você deverá adicionar mais um atributo na classe jogador para poder controlar os jogadores que já efetuaram o treinamento e que só poderão treinar após o próximo jogo.

Exercício 5

Crie uma nova classe chamada Time conforme abaixo:

a) Cadastrar/Instanciar pelo menos 2 times e em cada time pelo menos 23 jogadores.

b) Observe que o método relacionarJogadores deve ser criado. Este método irá selecionar os 11 melhores jogadores de acordo com a qualidade (os 11 jogadores com a maior qualidade devem ser relacionados).

c) Em uma outra classe, crie o método main, o qual cadastra os times e os jogadores e ao final imprimirá as escalações de ambos os times.

d) Altere o método relacionarJogadores para selecionar 18 jogadores, 11 titulares e 7 reservas ordenados pela qualidade, ou seja, os 11 titulares devem estar alocados nas primeiras posições do array e os 7 reservas nas últimas posições do array.

e) Altere o método relacionarJogadores para considerar não apenas a qualidade dos jogadores, mas também a sua posição, assim os 18 relacionados devem ser os melhores em sua posição, mas não necessariamente os jogadores com as maiores qualidades no elenco.

f) Em uma outra classe, crie o método main, o qual cadastra os times e os jogadores e ao final imprimirá as escalações de ambos os times, porém, agora também devem ser informados

Exercício 6

Construa a classe Livros que obedeça à descrição abaixo:

- A classe possui os atributos titulo, qtdPaginas e paginasLidas. Esses atributos devem ser marcados com o modificador de acesso private.
- Crie os métodos get e set para cada um dos atributos.
- Crie ainda o método verificarProgresso que deverá calcular a porcentagem de leitura do livro até o momento. Para isso, deverá usar os valores dos atributos qtdPaginas e paginasLidas, através da formula: porcentagem = paginasLidas * 100 / qtdPaginas. O valor da porcentagem deverá ser mostrado na tela conforme a mensagem "Você já leu X por cento do livro", onde o valor de X é o valor calculado pela fórmula apresentada anteriormente.

Exercício 7

Crie uma classe TestarLivros no mesmo pacote da classe Livros da questão anterior. Essa classe possuirá apenas o método main que servirá para testar a classe Livros. As seguintes ações devem ser realizadas:

- Crie um objeto livrofavorito do tipo Livro.

- Altere o atributo titulo para "O Pequeno Príncipe". Utilize, para isso, o método setTitulo;

- Altere o atributo qtdPaginas para 98. Utilize, para isso, o método setQtdPaginas; Escreva na tela a mensagem: "O livro X possui Y páginas", onde no lugar de X deverá aparecer o valor do atributo titulo e, no lugar de Y deverá aparecer o valor do atributo qtdPaginas. Utilize, para tanto, os métodos getTitulo e getQtdPaginas.

- Altere a quantidade de paginasLidas para 20;

- Chame o método verificarProgresso.

- Altere a quantidade de paginasLidas para 50; • Chame o método verificarProgresso. • Instancie outros 10 livros no método main e chame os métodos desenvolvidos, conforme o exemplo anterior.

Exercício 8

Dada as classes a seguir:

Representa o total de despesas de um mês

```
public class DespesaMes {

private int mes; // mes da despesa

private float valor; // valor da despesa
```

```
 public DespesaMes(int mes, float valor) {
        this.mes = mes;
         this.valor = valor; }
public int getMes() {
        return mes; }
public float getValor() {
        return valor; }
}
```

Representa o total de despesas de um dia

```
public class DespesaDia extends DespesaMes {
private int dia; // dia da despesa
public DespesaDia(int dia, int mes, float valor) {
        super(mes, valor);
        this.dia = dia; }
public int getDia() {
        return dia; }
}
```

Escreva uma classe que representa todas as despesas de um indivíduo. Ela mantém um vetor onde podem ser registradas tanto despesas de um dia (DespesaDia), quanto despesas de um mês (DepesaMes). A classe implementa os seguintes métodos:

Construtor	Recebe como parâmetro o CPF e um veotr com as despesas de um indivíduo e as guarda.
getCPF	Retorna o CPF do indivíduo.
totalixaMes	Recebe um parâmetro com um mês (int) e retorna um objeto da classe DespesaMes onde estará registrada a soma de todas as despesas que o indivíduo fez naquele mês.

Exercício 9

Escreva uma classe que represente um país. Um país é representado através dos atributos: código ISO 3166-1 (ex.: BRA), nome (ex.: Brasil), população (ex.: 193.946.886) e a sua dimensão em Km2 (ex.: 8.515.767,049). Além disso, cada país mantém uma lista de outros países com os quais ele faz fronteira. Escreva a classe em Java e forneça os seus membros a seguir:

a) Construtor que inicialize o código ISO, o nome e a dimensão do país;

b) Métodos de acesso (getter/setter) para as propriedades código ISSO, nome, população e dimensão do país;

c) Um método que permita verificar se dois objetos representam o mesmo país (igualdade semântica). Dois países são iguais se tiverem o mesmo código ISO;

d) Um método que informe se outro país é limítrofe do país que recebeu a mensagem;

e) Um método que retorne a densidade populacional do país;

f) Um método que receba um país como parâmetro e retorne a lista de vizinhos comuns aos dois países. Considere que um país tem no máximo 40 outros países com os quais ele faz fronteira.

Exercício 10

Crie uma classe Matriz que represente uma matriz matemática. Forneça um construtor que permita a inicialização das dimensões da Matriz Forneça métodos para acesso (leitura/escrita) de cada elemento da matriz. Forneça os métodos adequados para as seguintes operações com matriz:

a) Comparação semântica da matriz;

b) Retornar a transposta (é aquela onde as linhas se transformam em colunas e as colunas em linhas) da matriz.

c) Retornar a oposta (é aquela onde todos os elementos possuem sinais trocados) da matriz;

d) Gere uma matriz nula (é aqueles onde todos os elementos são iguais a 0);

e) Informe se a matriz é identidade (matriz quadrada onde os elementos da diagonal principal são todos iguais a 1 e os demais 0);

f) Informe se a matriz é diagonal (matriz quadrada onde os elementos fora da diagonal principal são todos iguais a 0).

g) Informe se a matriz é singular (matriz diagonal onde os elementos da diagonal principal são todos iguais);

h) Informe se a matriz é simétrica (uma matriz quadrada é dita simétrica se ela é igual a sua transposta);

i) Informe se a matriz é anti-simétrica (uma matriz quadrada é dita anti-simétrica se sua oposta é igual a sua transposta)

j) Adicionar duas matrizes (alterando o valor da que recebeu a mensagem);

k) Subtrair duas matrizes(alterando o valor da que recebeu a mensagem);

l) Multiplicar duas matrizes(alterando o valor da que recebeu a mensagem);

m) Gere uma cópia da matriz.