Aula prática nº 12 – Exemplos de Exames práticos

A

Pretende-se desenvolver um programa que possibilite a gestão de cabines num cruzeiro. Devem ser suportadas as seguintes entidades principais, bem como os seus atributos:

- uma cabine tem número, capacidade máxima de pessoas que pode acomodar (*inteiros*) e lista de ocupantes (*strings*);
- um cruzeiro tem nome de navio, lista de cidades que visita, data quando começa (*strings*), um conjunto de cabines (ordenadas pelo seu número) e duração em dias;
- uma cabine com janela tem indicação se a janela dá para o interior do navio ou para o mar;
- uma cabine com varanda tem indicação se é de vista livre para o mar, parcialmente obstruída ou totalmente obstruída;
- uma suite, que é uma cabine com vários quartos (*inteiro*);
- os ocupantes de cabines com varanda e de cabines com janela podem escolher um dos pacotes de extras (dança, programação em java, internet, desporto);

1. Comece por identificar todas as entidades envolvidas neste sistema de informação, assim como as tarefas associadas a cada uma delas. Defina relações entre essas entidades e efetue o desenho do sistema. Implemente todas as classes. Inclua todos os métodos que sejam necessários para suportar as funcionalidades explicitamente pedidas e também os métodos que podem vir a ser necessários para reutilizar estas classes em projetos futuros.

Teste o seu código com a função *main* apresentada na página seguinte.

2. Adicione a possibilidade de guardar num ficheiro de texto (“SeaPrincess-2017.txt”) a seguinte informação:

1) a primeira linha com a percentagem de cabines disponíveis para venda;
2) a segunda linha com a média de pessoas por quarto nas Suites ocupadas;
3) as restantes com os dados das cabines ocupadas, ordenadas por número.

```
public class POO_P_S2 {
public static void main(String[] args) {
        POO_P_S2 ap = new POO_P_S2();

        ap.alinea1();
        //      ap.alinea2();
}

public void alinea1() {
        Cruzeiro cruz = criarCruzeiro();

        // imprime os produtos ordenados por data
        System.out.println(cruz);
}

public void alinea2() {
        Cruzeiro cruz = criarCruzeiro();

        // coloque o código da alínea 2 aqui

}

public Cruzeiro criarCruzeiro() {
        String[] aux={"Lisboa","Barcelona","Rodes","Southampton"};
        Cruzeiro cr2 = new Cruzeiro("Sea Princess", aux , "22/01/2017");
        cr2.setDuracao(11);

        CabineComJanela ccj=new CabineComJanela(17, 2, TipoDeJanela.INTERIOR);
        ccj.setPassageiros(new String("Maria Luz;Manuel Luz").split(";"));
        ccj.pacoteExtra(Extra.Spa); cr2.add(ccj);
        cr2.add(new CabineComJanela(15, 4, TipoDeJanela.INTERIOR,
                        new String("António Campos;Maria Campos;Marina Mota").split(";")));
        cr2.add(new CabineComJanela(25, 2,
TipoDeJanela.INTERIOR,"Anonymous1;Anonymous2".split(";")));
        cr2.add(new CabineComJanela(4, 4, TipoDeJanela.MAR, new String("Ursula Magnusson and
Matts Magnusson and Miki Rosberg and Charles Sean").split(" and ")));
        Suite suite1=new Suite(100, 2); suite1.setNumQuartos(3);
        suite1.setMaxOcupantes(2*3); cr2.add(suite1);
        Suite s=new Suite(102, 6); s.setNumQuartos(3); cr2.add(s);
        s.setPassageiros(new String("A. Jolie:B. Pitt:Shiloh:Knox Leon").split(":"));
        CabineComVaranda cab=new CabineComVaranda(21, 1, TipoDeVaranda.VISTA_LIVRE);
        cab.setTipoVaranda(TipoDeVaranda.OBSTR_PARCIAL);
        cab.pacoteExtra(Extra.Desporto);
        try {
                cab.setPassageiros(new String("Paulo Portas;Júlia Portas").split(";"));
                cr2.add(cab);
        } catch (IllegalArgumentException e){
                System.out.println("Não adicionado devido a excesso de ocupantes !!");
        }
        Cabine eo = new CabineComJanela(1,4,TipoDeJanela.MAR,"Marcelo R. de Sousa".split(";"));
        cr2.add(eo);
        cr2.add(new CabineComJanela(130,4,TipoDeJanela.MAR));
        cr2.add(new CabineComJanela(131,4,TipoDeJanela.INTERIOR));

        return cr2;
}
```

Resultado obtido na saída:

```
Não adicionado devido a excesso de ocupantes !!
Navio Sea Princess, partida em 22/01/2017
Itinerário:[Lisboa, Barcelona, Rodes, Southampton]
Cabine com Janela MAR [ N°1( max 4 pessoas ) : [Marcelo R. de Sousa]]
Cabine com Janela MAR [ N°4( max 4 pessoas ) : [Charles Sean, Matts Magnusson, Miki Rosberg,
Ursula Magnusson]]
Cabine com Janela INTERIOR [ N°15( max 4 pessoas ) : [António Campos, Maria Campos, Marina
Mota]]
Cabine com Janela INTERIOR Spa Extra [ N°17( max 2 pessoas ) : [Manuel Luz, Maria Luz]]
Cabine com Janela INTERIOR [ N°25( max 2 pessoas ) : [Anonymous1, Anonymous2]]
Suite c/3 quartos [ N°100( max 6 pessoas ) : Disponível para venda!]
Suite c/3 quartos [ N°102( max 6 pessoas ) : [A. Jolie, B. Pitt, Knox Leon, Shiloh]]
Cabine com Janela MAR [ N°130( max 4 pessoas ) : Disponível para venda!]
Cabine com Janela INTERIOR [ N°131( max 4 pessoas ) : Disponível para venda!]
```

B

Pretende-se desenvolver um programa que possibilite a gestão de publicações numa biblioteca. As entidades principais neste sistema de informação são livros, revistas e jornais. Devem ser suportadas as caraterísticas seguintes:

- a biblioteca, para além de conter um conjunto variável dos elementos seguintes, tem um nome e endereço (strings);
- um livro tem um título, editora, ISBN e uma lista de autores (strings);
- uma revista tem título, editora e ISSN (strings);
- um jornal tem título, editora e indicação se sai na versão preto/branco ou a cores;
- para revistas e jornais deve-se poder definir a sua periodicidade, que pode ser diária, quinzenal, mensal ou bimestral.

1. Comece por identificar todas as entidades envolvidas neste sistema de informação, assim como as tarefas associadas a cada uma delas. Defina relações entre essas entidades. Implemente todas as classes. Inclua todos os métodos que sejam necessários para suportar as funcionalidades explicitamente pedidas e também os métodos que podem vir a ser necessários para reutilizar estas classes em projetos futuros.

Teste o seu código com a função main apresentada na página seguinte.

2. Adicione a possibilidade de se poder ordenar todas as publicações por título. Imprima o resultado na consola.

3. Adicione a possibilidade de se poder ler dados de livros (tipo 1) e revistas (tipo 2) de um ficheiro de texto com a estrutura seguinte (os campos são separados por tab):

Tipo	Título	Editora	Autor(es)	ISBN/ISSN
1	Papillon	Amadora: Bertrand	Henri Charrière	
2	Revista municipal	Aveiro: C.M.A.		
1	História da Europa	Lisboa: Dom Quixote	Jean-Baptiste Duroselle	972-20-0824-2

Teste com o ficheiro fornecido (samples.txt), calcule o total de publicações por editora e apresente no ecrã uma tabela do tipo (Editora, #publicações).

```
public static void main(String[] args) {
      POO15Ap ap = new POO15Ap();
      ap.alinea1();
//    ap.alinea2();
//    ap.alinea3();
}

public void alinea1() {
      Biblioteca bAveiro = new Biblioteca("Biblioteca Municipal de Aveiro",
"Largo Dr. Jaime Magalhães Lima, 3800 - 156 Aveiro, Portugal");

    bAveiro.add(new Livro("História da Europa", "Lisboa: Dom Quixote",
            "Jean-Baptiste Duroselle", "972-20-0824-2"));
    bAveiro.add(new Livro("Papillon", "Amadora: Bertrand", "Henri Charrière"));
    bAveiro.add(new Livro("Branca de neve e os sete anões", "Abril Morumbi",
            "Jacob Grimm; Wilhelm Grimm"));
    bAveiro.add(new Revista("Revista municipal", "Aveiro: C.M.A.", "0874-727X"));
      bAveiro.add(new Jornal("Diário de notícias"), Ver.CORES);
      Journal gaf = new Jornal ("O gafanhoto");
      gaf.setPeriodicidade(Period.BIMESTRAL);
      bAveiro.add(gaf);

      System.out.println(bAveiro);   // imprime todas as publicações da
biblioteca
    try {
            PrintWriter fl = new PrintWriter(new File("output.txt"));
            fl.println(bAveiro);
            fl.close();
      } catch (FileNotFoundException e) {
            e.printStackTrace();
      }
}
```

C

Pretende-se desenvolver um programa que possibilite a gestão de testes numa escola de línguas. As entidades principais neste sistema de informação são 1) exame escrito, 2) prova oral e 3) teste no computador. Devem ser suportadas as caraterísticas seguintes:

- a escola, para além de conter um conjunto variável dos elementos seguintes, tem um nome e endereço (strings);
- cada teste tem associados o nome da disciplina, o professor responsável, a data/hora e a lista de salas utilizadas (strings);
- o exame escrito tem o número de versões que se deve produzir (por omissão, 1) e pode ser realizado com ou sem consulta (por omissão, sem consulta);
- a prova oral é realizada numa das seguintes línguas (UK por omissão): PT, UK, ES, DE.
- o teste nos computadores tem associado um link (string) para submissão da prova;
- para exames escritos (1) e testes nos computadores (3) deve-se poder definir e consultar o prazo limite de correção (setPrazo e getPrazo).

1. Comece por identificar todas as entidades envolvidas neste sistema de informação, assim como as tarefas associadas a cada uma delas. Defina relações entre essas entidades. Inclua todos os métodos que sejam necessários para suportar as funcionalidades explicitamente pedidas e também os métodos que podem vir a ser necessários para reutilizar estas classes em projetos futuros.
Teste o seu código com a função main apresentada na página seguinte.
2. Adicione a possibilidade de se poder ordenar todas as provas com prazo limite de correção (i.e. exames escritos e testes nos computadores) por nome do curso. Imprima o resultado na consola.
3. Adicione a possibilidade de se poder ler dados das provas dum ficheiro de texto (samples_s2.txt) com a estrutura seguinte (os campos são separados por tab):

Tipo	Título	Professor	Data/hora	Sala(s)
1	Inglês para Empresas	Anthony Laurel	12.06.2015 15h	15; 22
3	Italiano Intermédio	Luca Benini	01.06.2015 10h	25
1	Alemão	Manfred Glesner	12.06.2015 10h	13
2	Inglês Avançado	Anthony Laurel	05.06.2015 10h	3

o tipo indica: 1 – um exame escrito; 2 – uma prova oral; 3 – um teste nos computadores.
Calcule o total de provas por professor e apresente no ecrã uma tabela (Professor, #provas).

```
public class POO15Ap {
	public static void main(String[] args) {
		POO15Ap ap = new POO15Ap();
		ap.alinea1();
		// ap.alinea2();
		// ap.alinea3();

	}

	public void alinea1() {
		System.out.println("\nA1)");
		Escola ih = new Escola("International House Aveiro",
				"Rua Domingos Carrancho, 1, 3800-145 Aveiro");
		ExameEscrito e1 = new ExameEscrito("Inglês para Empresas",
				"Anthony Laurel", "12.06.2015 15h", "15; 22", 4); // 4
versões
		e1.setConsulta(true);
		e1.setPrazo(6);
		ih.add(e1);
		ih.add(new ExameEscrito("Alemão", "Manfred Glesner", "12.06.2015
10h", "13"));
		TesteComputador t1 = new TesteComputador("Italiano Intermédio",
				"Luca Benini", "01.06.2015 10h", "25");
		t1.setLink("http://www.ihaveiro.com/outras-linguas/tii");
		ih.add(t1);
		ih.add(new ProvaOral("Inglês Avançado", "Anthony Laurel",
				"05.06.2015 10h", "3", Lingua.UK));
		System.out.println(ih); // imprime todas as provas da escola

		System.out.println(ih);
		try {
			PrintWriter fl = new PrintWriter(new File("output2.txt"));
			fl.println(ih);
			fl.close();
		} catch (FileNotFoundException e) {
			e.printStackTrace();
		}
	}
}
```

D

Pretende-se desenvolver um programa que possibilite a gestão de produtos numa pequena loja de alimentos. As entidades principais neste sistema de informação são: produtos lácteos (1), cereais (2) e refrigerantes (3). Devem ser suportadas as caraterísticas seguintes:

- a loja, para além de conter um conjunto variável dos elementos seguintes, tem um nome e endereço (strings);
- todos os alimentos têm associados um nome (string), preço, calorias por 100 gramas e a data de validade (string);
- os produtos lácteos registam o teor de gordura;
- os cereais registam a composição que pode ser de {trigo}, {arroz e trigo} ou {milho};
- os lácteos (1) e refrigerantes (3) são guardados no frigorífico e deve-se poder definir e consultar as temperaturas mínima e máxima de conservação (setTemp, getTempMin e getTempMax). Por omissão, as temperaturas são de 3.0 ºC (min) a 5.0 ºC (max).

1. Comece por identificar todas as entidades envolvidas neste sistema de informação, assim como as tarefas associadas a cada uma delas. Defina relações entre essas entidades. Inclua todos os métodos que sejam necessários para suportar as funcionalidades explicitamente pedidas e também os métodos que podem vir a ser necessários para reutilizar estas classes em projetos futuros.

Teste o seu código com a função main apresentada na página seguinte.

2. Adicione a possibilidade de se poder ordenar todos os alimentos que são guardados no frigorífico (i.e. produtos lácteos e refrigerantes) por nome. Imprima o resultado na consola.

3. Adicione a possibilidade de se poder ler dados dos alimentos dum ficheiro de texto (alimentos.txt) com a estrutura seguinte (os campos são separados por tab):

Tipo	Nome	Preço	Calorias	Data	Gordura
1	Iogurte Natural Danone	1.48	56	07.06.2015	4.0
3	Coca Cola	1.12	300	15.09.2015	
1	Leite UHT Meio Gordo Mimosa	0.64	46	20.06.2015	1.7
2	Nestlé Fitness	1.59	200	15.09.2015	

o tipo indica: 1 – lácteo; 2 – cereal; 3 – refrigerante.

Calcule o total de alimentos por data de validade e apresente no ecrã uma tabela (data_de_validade, #alimentos).

```
public class POO15AP {
   public static void main(String[] args) {
      POO15AP ap = new POO15AP();
      ap.alinea1();
//    ap.alinea2();
//    ap.alinea3();
   }

   public void alinea1() {
      System.out.println("\nA1)");
      LojaAlimentar lZe = new LojaAlimentar("Loja do Zé",
"Rua Domingos Carrancho, 15, 3800-145 Aveiro");

      Lacteo lact1 = new Lacteo("Iogurte Natural Danone", 1.48, 56,
"07.06.2015");
 // nome, preço, calorias, validade
      lact1.setGordura(4.0);
      lact1.setTemp(4, 6); // temperaturas mínima e máxima
      lZe.add(lact1);

      lZe.add(new Lacteo("Leite UHT Meio Gordo Mimosa", 0.64, 46, "20.06.2015",
1.7));
// o último valor é a gordura

      Cereal c1 = new Cereal("Nestlé Fitness", 1.59, 200, "15.09.2015");
      c1.setComp(TIPO_CEREAL.ARROZ_TRIGO);
      lZe.add(c1);

      lZe.add(new Refrigerante("Cola Cola", 1.12, 300, "15.09.2015"));

      System.out.println(lZe); // imprime todos os produtos da loja

      try {
            PrintWriter fl = new PrintWriter(new File("output3.txt"));
            fl.println(lZe);
            fl.close();
      } catch (FileNotFoundException e) {
            e.printStackTrace();
      }
   }
}
```