PROVA DE EXAME
de
Programação Orientada por Objetos

INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Tecnologia e Gestão

Engenharia Informática
Informática de Gestão

Época Normal – 29 de janeiro de 2020

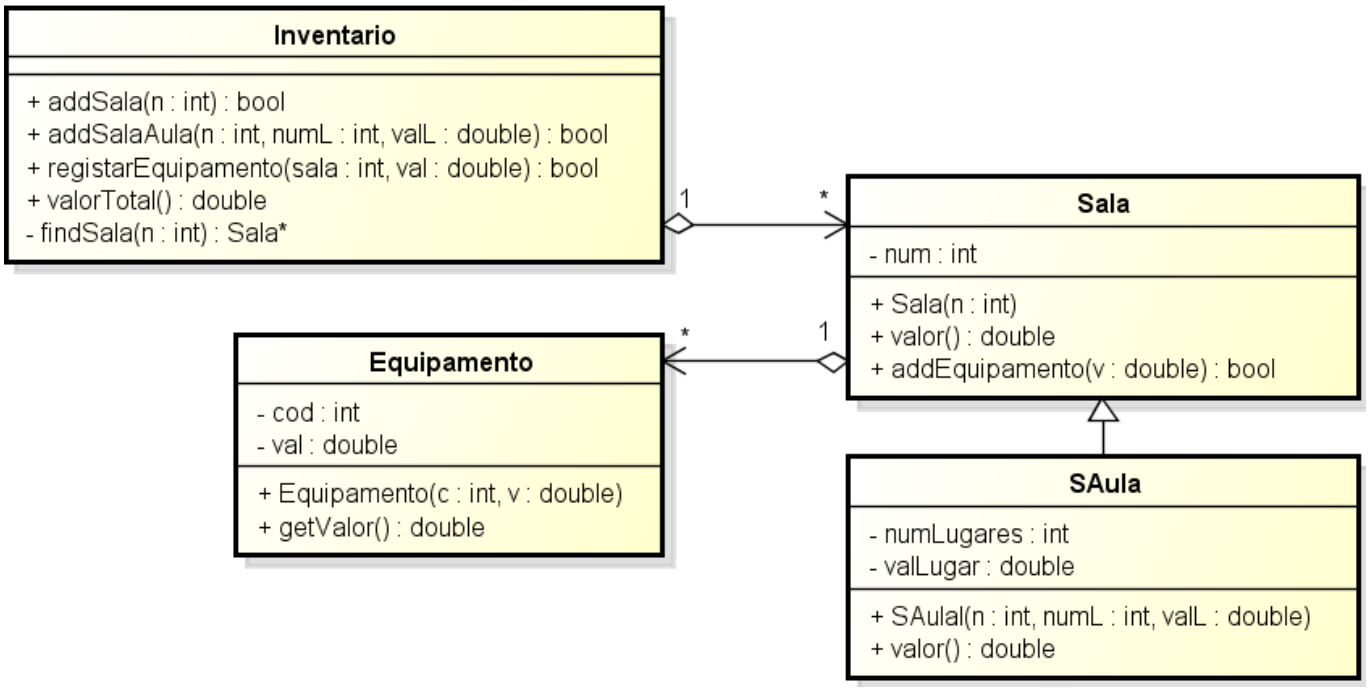
Notas importantes:

1. *A duração da prova é de 1 hora e 30 minutos.*
2. *Comece por preencher a zona reservada à sua identificação na folha de exame.*

Grupo I

[15 valores]

Com o diagrama UML que se segue pretende-se modelar um sistema simplificado de gestão do inventário de uma escola. O sistema de gestão em causa é responsável por manter o registo dos diferentes bens (equipamentos) alocados às salas, cada um com um valor financeiro estimando. Tratando-se de uma sala normal, o valor financeiro que representa será simplesmente o somatório do valor dos bens que integra; mas caso se trate de uma sala de aula, a esse montante deverá ser ainda adicionado o valor estimado para o mobiliário que equipa todos os lugares de que dispõe. Cada sala de aula tem um determinado número de lugares e um valor financeiro médio estimado para equipar cada lugar. A aplicação terá, como principal funcionalidade, o cálculo automático do valor financeiro de todo o inventário da escola.



a) Defina em C++ todas as classes do problema, respeitando integralmente os métodos e atributos descritos no diagrama, e tendo ainda em conta todas as seguintes considerações: [11.4 val.]

- *Não defina quaisquer outros métodos ou atributos, a não ser, nas classes em que se justifique, o operador que permita que os objetos sejam colecionáveis;*

- *O código do equipamento deverá ser automaticamente atribuído pelo sistema e ser único dentro de cada sala, usando o código 1 para o 1º equipamento, o 2 para o 2º, e assim sucessivamente;*
- *As coleções deverão ser implementadas com base nos templates de classes Colecao ou ColecaoHibrida que utilizou nas aulas de POO. Para efeitos de consulta, disponibilizam-se os protótipos dos seus principais métodos no Anexo B deste enunciado;*
- *Defina os métodos no momento em que está a declarar as classes (não separe a declaração da implementação) e considere que todo o código reside num único ficheiro;*
- *Nesta e nas restantes alíneas, não precisa de indicar as diretivas #include.*

b) Acrescente ao problema os métodos que permitam listar na saída standard o inventário de uma dada sala, com indicação discriminada do valor de cada um dos seus equipamentos e, caso faça sentido, o valor total estimado para o mobiliário que equipa os lugares. [3.6 val.]

Grupo II

[5 valores]

No Anexo A do presente enunciado encontra-se o código de um pequeno programa em C++. Depois de o analisar com a devida atenção, responda às seguintes questões.

a) Diga quais são os métodos construtores (definidos explicitamente ou não) que estão presentes em cada uma das classes do problema. [0.6 val.]

b) Apresente o resultado que será visualizado na saída standard com a execução do programa. [2.8 val.]

c) Diga que consequências é que teria, naquilo que é visualizado, caso o método `print()` da classe `A` fosse virtual. [0.4 val.]

d) Dê uma breve explicação do erro cometido quando se acrescenta ao `main` do programa do Anexo A uma última linha com cada um dos seguintes códigos: [1.2 val.]

(1) `cout << objeto << endl;`

(2) `A vetor[3];`

(3) `A *p = new B();`

(4) `C *p = new C(objeto); delete p; objeto.print();`

ANEXO A

```
#include<string>
#include<iostream>
using namespace std;


class A {
	int val;
public:
	A(int v) :val(v) { cout << "Criado A(" << val << ')' << endl; }
	void print() { cout << "A(" << val << ")"; }
	~A() { cout << "Destruido A" << endl; }
};


class B {
	A obj;
	static int cont;
public:
	B(): obj(++cont) {cout << "Criado " << 'B' << endl;}
	void print() { cout << "B:"; obj.print();}
	~B() { cout << "Destruido B" << endl; }
};

int B::cont=0;


class C {
	B *vetor;
	int nbs;
public:
	C(int n=1) {
		nbs = n;
		vetor = new B[nbs];
		cout << "Criado um C com " << nbs << " Bs" << endl;
	}
	void print() {
		for (int i = 0; i < nbs; i++)
			{ cout << "C:"; vetor[i].print(); cout << endl; }
	}
	~C() { cout << "Destruido C" << endl; delete[] vetor;}
};


void main() {
	C objeto(3);
	cout << "-1-" << endl;
	objeto.print();
	cout << "-2-" << endl;
}
```

ANEXO B

Template Colecao

```
#include<set>
using namespace std;

template<class K>
class Colecao: public set<K>{
public:
	bool insert(const K &c);
	K *find(const K &c);
	int size() const;
	void erase(const K &);
	//void clear();
	//bool empty() const;
	//iterator begin();
	//iterator end();
};
```

Template ColecaoHibrida

```
#include<set>
using namespace std;

template<class T>
class less_pointers{
public:
	bool operator()(const T &left, const T &right) const {
		return (*left < *right);
	}
};

template<class K>
class ColecaoHibrida: public set<K, less_pointers<K>>{
public:
	bool insert(const K &c);
	K find(const K &c);
	int size() const;
	void erase(const K &);
	//void clear();
	//bool empty() const;
	//iterator begin();
	//iterator end();
};
```