INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Tecnologia e de Gestão

Departamento de Informática e Comunicações

Engenharia Informática / Informática de Gestão
2º Ano
Época Normal

Programação Orientada por Objectos — Data: 25/01/2010
1º Semestre — Duração: 2h00

Grupo I
(10 valores)

Suponha que quer utilizar os conhecimentos adquiridos em POO para desenvolver uma pequena aplicação que lhe permita fazer a gestão diária de um conjunto de salas de aula, utilizadas para a leccionação de diferentes Unidades Curriculares (UC). Cada sala é caracterizada pela sua identificação (ex: sala 12) e pela lotação, e cada UC pelo seu nome e número de alunos inscritos.
No sentido de simplificar a solução, considere que todas as aulas têm uma hora de duração, que iniciam sempre a horas certas (9h, 10h, 11h, …), e repare que apenas se pretende gerir a alocação das salas ao longo de um único dia.

a) Apresente, através de um diagrama de classes em UML, a solução por si idealizada para sustentar a aplicação descrita, indicando para cada classe apenas os atributos estritamente necessários, um método construtor e outros componentes que considere essenciais ao funcionamento da solução (*getters*, *setters* e, nas classes em que se justifique, o operador que permita que os objectos sejam coleccionáveis).

b) Codifique em C++ a sua solução, tal como a descreveu no diagrama da alínea anterior.

NOTA: As colecções deverão ser implementadas com base no template de classes *Coleccao* que utilizou nas aulas de POO. Para efeitos de consulta, disponibilizam-se os protótipos dos seus principais métodos*:*

```
template<class K>
class Coleccao: 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();
};
```

c) Implemente o(s) método(s) necessário(s) para registar uma nova sala na aplicação.

d) Implemente o(s) método(s) necessário(s) para listar as aulas alocadas a uma dada sala.

e) Implemente o(s) método(s) necessário(s) para alocar uma aula de uma dada UC a uma sala. Se a sala indicada não se encontrar disponível à hora pretendida ou se a sua lotação não for suficiente, a operação de alocação deve ser rejeitada.
Considere que a classe Sala já dispõe de um método, de protótipo `bool disponivel(int h)`, que verifica se a mesma se encontra disponível à hora `h` (i. e., entre a hora h e h+1).

Grupo II
(10 valores)

Considere o seguinte programa:

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

class Ponto{
      float x, y; //coordenadas do ponto bidimensional
public:
      Ponto(){x=y=0;}
      Ponto(float a, float b){x=a; y=b;}
      float getx() const{return x;}
      float gety() const{return y;}
      ~Ponto(){cout<<"Ponto diz Xau\n";}
};

ostream &operator<<(ostream &os, const Ponto &p){
      os<<'('<<p.getx()<<','<<p.gety()<<')';
      return os;
}

class Figura{
protected:
      Ponto posic; //localização da figura
public:
      Figura(const Ponto& p): posic(p){}
      virtual void mover(const Ponto& p){posic=p;}
      virtual void mostrar() const=0;
      virtual ~Figura(){cout<<"Figura diz Xau\n";}
};
```

```
class Circunferencia: public Figura{
protected:
	int raio;
public:
	Circunferencia(const Ponto& c, int r): Figura(c){raio=r;}
	void ampliar(int a){raio*=a;}
	void mostrar() const{
		cout<<"Circunferencia de raio "<<raio;
		cout<<" com centro em "<<posic<<endl;
	}
	~Circunferencia(){cout<<"Circunferencia diz Xau\n";}
};

void main(){
	Ponto p1(0.1, 10), p2(0.2, 20);
	Circunferencia circ(p1, 5);
	Figura *f=&circ;
	f->mostrar();
	f->ampliar(3);
	f->mostrar();
	f->mover(p2);
	f->mostrar();
}
```

a) Identifique o tipo de relação existente entre *Figura e Circunferencia* e entre *Figura* e *Ponto*.

b) Na função main(), a invocação de um dos métodos resulta em erro. Identifique o erro e proponha uma solução adequada para que essa invocação passe a ser possível.

c) Apresente o resultado que será visualizado na saída standard após a execução do programa e depois corrigido o erro referenciado na alínea anterior.

d) Se acrescentarmos à função main() a linha de código que se segue, daí resultará algum erro na nossa aplicação? Se sim, proponha uma solução adequada para que essa linha de código deixe de dar erro.

```
	Circunferencia varias[10];
```

e) Acrescente ao programa uma nova classe de nome Circulo que, para alem de um centro e raio, tenha ainda como atributo a cor de preenchimento.