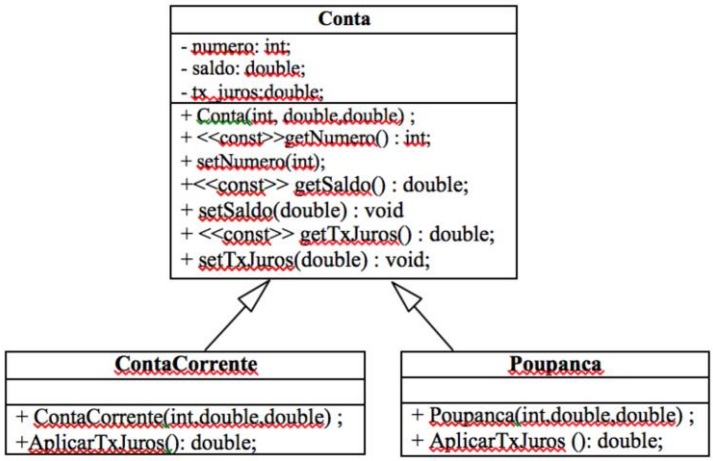
UNIVERSIDADE FEDERAL DE VIÇOSA
DEPARTAMENTO DE INFORMÁTICA
Inf 213 - Estrutura de dados
Primeira prova – 04/04/16
Prof. Marcus Vinícius Alvim Andrade



1) (25%) a) Declare as classes definidas no diagrama UML ao lado. **NÃO PRECISA** implementar as funções membros.

b) Implemente a função *AplicarTxJuros* das classes *ContaCorrente* e *Poupanca* que deve ter o seguinte comportamento: dado o valor obtido multiplicando a *tx_juros* pelo *saldo*, no caso de *Conta Corrente*, se o saldo for negativo, o saldo deve ser reduzido deste valor; no caso de *Poupanca*, o saldo deve ser acrescido deste valor.

c) Escreva uma função que recebe como parâmetro um array contendo todas as contas existentes num banco e o tamanho do array e atualiza o saldo destas contas realizando a aplicação da taxa de juros. OBS: caso necessário, você pode alterar a declaração das classes *Conta*, *ContaCorrente* e *Poupanca.*

2) (25%) Dada a classe Q2 e a função *f* abaixo, mostre o resultado dos trechos de código. Se houver erro de compilação, faça as correções necessárias **NA CLASSE** e depois liste o resultado. **OBS:** você **só pode** alterar a classe.

```
template <Class T>
class Q2 {
  public:
    Q2(int n=10) { ptr = new T[n];  dim=n;}
    ~Q2() { cout << "Desalocou ptr \n"; delete ptr;}
    void set(T x, int i) {ptr[i]=x; }
    T get(int i) { return ptr[i];}

  private:
    T *ptr;
     int dim;
};

void f(Q2<int> u) {
      cout << u.get(0) << endl;
}
```

a)
```
Q2<int> *v;
    v = new Q2<int>[4];
    delete [] v;
```

b)
```
Q2<int> v;
    v.set(18,0);
    f(v);
    cout << v.get(0) << endl;
```

c)
```
Q2<char> c;
    c.set('a',0);
    {
       Q2<char> d = c;
       cout << d.get(0) << endl;
     }
    cout << c.get(0) << endl;
```

3) (25%) a) Escreva uma função que recebe um array de **apontadores para inteiros** com tamanho *n* e um inteiro *x* e retorna o índice (posição) do array que contém um apontador que aponta para uma área que contém o valor de *x*. Caso este índice (posição) não seja encontrado, a função deve lançar uma exceção para indicar este fato. Além disso, a função também deve lançar uma outra exceção para indicar que existe uma posição no array com valor *NULL*. OBS: caso ache necessário, você pode criar classes para indicar cada uma dessas exceções.
b) Escreva um programa que cria um array de **apontadores para inteiros** com tamanho 10, a função deve ser chamada e no retorno da função, caso o valor de *x* tenha sido encontrado, programa deve imprimir a posição retornada pela função. O programa deve tratar duas exceções informando ao usuário qual exceção ocorreu, isto é, o programa deve imprimir "o valor procurado não foi encontrado" ou "há uma posição no array que contém o valor *NULL*".

CONTINUA NO VERSO!!!

4) (25%) Considerando as declarações as classes *P2* e *P3* abaixo, verifique se os trechos de códigos dados nas letras a), b) e c) podem ser compilados e, em caso afirmativo, escreva o resultado obtido. Caso o trecho não possa ser compilado, indique a causa do erro.

```
class P2  {
public:
    P2() { x = y = 0; }
    virtual void reset() { x = y = 0; }
    void set(double xi, double yi ) { x = xi; y = yi; }
    virtual double calc() { return (x+y)/2;}
    void print() {cout << x << "  " << y;}
protected:
    double x,y;
};
```

```
class P3 : public P2  {
public:
   P3() : P2() { x = 0; c = '\0';};
   virtual void reset() { x = 0; c = '\0'; }
   void set(double xi,  double yi, double zi=0, char ci='#')
            {x = xi; y = yi; z = zi; c = ci; }
   virtual double calc()  { return sqrt(x*x + y*y + z*z); }
   void print() {P2::print(); cout << "  " << z << "  " << c;}
protected:
   double z;
   char c;
};
```

a)

```
    P2 m;
    P3 n;
    m.set(5,10);
    n = m;
    cout << "m = "; m.print();
    cout << " -- m.calc = " << m.calc() << endl;
    cout << "n = "; n.print();
    cout << " -- n.calc = " << n.calc() << endl;
    cout << endl;
```

b)

```
    P2 p;
    P3 q;
    p.set(5,10);
    q.set(1,2,2,'a');
    cout << "p = "; p.print();
    cout << " -- p.calc = " << p.calc() << endl;
    cout << "q = "; q.print();
    cout << " -- q.calc = " << q.calc() << endl;
    p = q;
    cout << "p = "; p.print();
    cout << " -- p.calc = " << p.calc() << endl;
    cout << "q = "; q.print();
    cout << " -- q.calc = " << q.calc() << endl;
    cout << endl;
```

c)

```
    P2 *a = new P2;
    P3 *b = new P3;
    a->set(10,20);
    b->set(4,3);
    cout << "a = "; a->print();
    cout << " -- a.calc = " << a->calc() << endl;
    cout << "b = "; b->print();
    cout << " -- b.calc = " << b->calc() << endl;
    a = b;
    cout << "a = "; a->print();
    cout << " -- a.calc = " << a->calc() << endl;
    cout << "b = "; b->print();
    cout << " -- b.calc = " << b->calc() << endl;
```