

## Exercício 22

Implemente em C++ uma classe que represente um **ponto com três dimensões**. Defina um método que calcule a **distância** entre dois pontos e outro que permita **alterar** as suas coordenadas.

## Ponto.h

```
#pragma once           //código do exercício 15
```

```
class Ponto{  
protected:  
    float x, y;  
public:  
    Ponto();  
    Ponto(float, float);  
    void print() const;  
    float dist(const Ponto &p) const;  
    void set(float, float);  
};
```

## Ponto.cpp

```
#include<iostream>
```

```
#include<math.h>
```

```
#include"Ponto.h"
```

```
using namespace std;
```

```
Ponto::Ponto(){x=y=0;}
```

```
Ponto::Ponto(float x1, float y1){x=x1; y=y1;}
```

```
void Ponto::print() const{
```

```
    cout << "(" << x << ", " << y << ")" << endl;
```

```
}
```

```
float Ponto::dist(const Ponto &p) const{
```

```
    return((float)sqrt(pow(x-p.x, 2)+pow(y-p.y, 2)));
```

```
}
```

```
void Ponto::set(float a, float b){x=a; y=b;}
```

## Ponto3D.h

```
#pragma once
```

```
#include "Ponto.h"
```

```
class Ponto3D: public Ponto{
```

```
    float z;
```

```
public:
```

```
    Ponto3D();
```

```
    Ponto3D(float, float, float);
```

```
    void print() const;
```

```
    double dist(Ponto3D &) const;
```

```
    void set(float , float , float );
```

```
};
```

## Ponto3D.cpp

```
#include "Ponto3D.h"
#include <iostream>
#include <math.h>
using namespace std;

Ponto3D::Ponto3D(){z=0;}
Ponto3D::Ponto3D(float a, float b, float c): Ponto(a, b){z=c;}

void Ponto3D::print() const{
    cout << "(" << x << ", " << y << ", " << z << ")" << endl;
}

double Ponto3D::dist(Ponto3D &p) const{
    return(sqrt(pow(x-p.x, 2) + pow(y-p.y, 2) + pow(z-p.z, 2)));
}

void Ponto3D::set(float a, float b, float c){
    Ponto::set(a, b);
    z=c;
}
```

```
#include "Ponto3D.h"
```

```
#include <iostream>
```

```
using namespace std;
```

```
void main(){
```

```
    Ponto3D p0, p1(1,2,3);
```

```
    p0.print();
```

```
    p1.print();
```

```
    cout<<endl;
```

```
    p0.set(-1,-2,-3);
```

```
    p0.print();
```

```
    cout<<"Distancia entre p0 e p1= "<<p0.dist(p1)<<" = "
```

```
                                <<p1.dist(p0)<<endl;
```

```
}
```

C:\Windows\system32\cmd.exe

```
(0, 0, 0)  
(1, 2, 3)
```

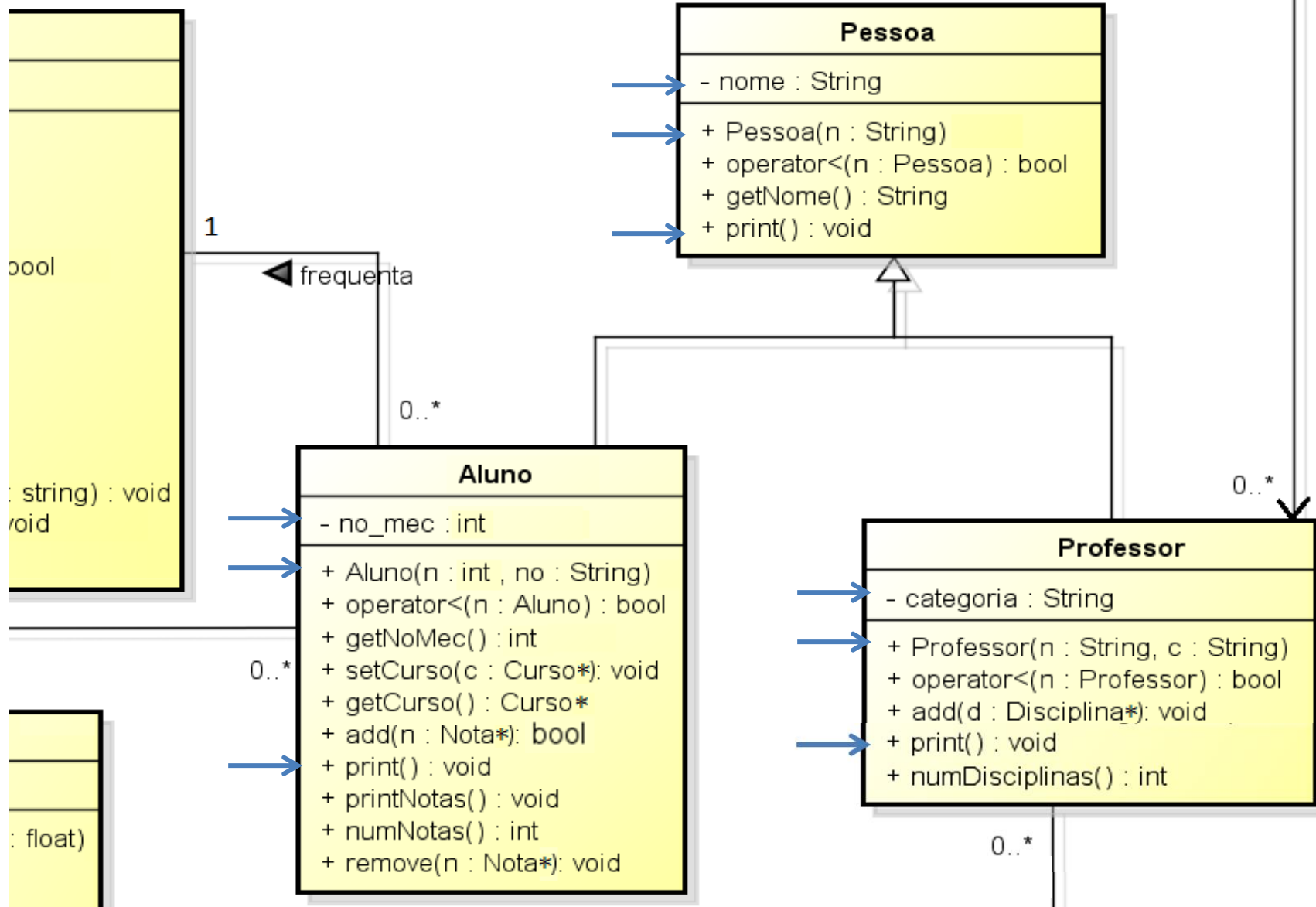
```
(-1, -2, -3)
```

```
Distancia entre p0 e p1= 7.48331 = 7.48331
```

```
Prima qualquer tecla para continuar . . . _
```

## Exercício 23

Implemente em C++ uma classe que represente uma **pessoa**, em que são descritos o atributo **nome** e a operação **print** para escrever os atributos na saída standard. Defina de seguida, uma classe **Aluno** adicionalmente caracterizada pelo **número** mecanográfico e outra **Professor** adicionalmente caracterizada pela **categoria**. Ambas as classes devem escrever os atributos na saída standard na operação **print**.





as declarações das classes deveriam ser em .h e implementação em .cpp

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

class Pessoa{
protected:
    string nome;
public:
    Pessoa(const string &n):nome(n){}
    void Print(){cout << nome << endl;}
};

class Aluno: public Pessoa{
    int no_mec;
public:
    Aluno(const string &n, int no):Pessoa(n) {no_mec=no;}
    void Print(){ Pessoa::Print();
                  cout << no_mec << endl;
    }
};
```

## continuação ...

```
class Professor: public Pessoa{
    string categoria;
public:
    Professor(const string &n, const string &c):
                                                Pessoa(n), categoria(c){}

    void Print(){
        Pessoa::Print();
        cout << categoria << endl;
    }
};
```