Composição: Objetos Como Membros de Classes

Composição

- Uma classe hipotética *RelogioComAlarme* deve saber o horário para soar o alarme
  - Então ele pode incluir um objeto da classe hipotética *Relogio*.
- Este relacionamento é do tipo "**tem um**" e é denominado **composição**;
- Vejamos um exemplo com estas duas classes hipotéticas.

Relogio.h

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

class Relogio
{
      public:
      Relogio(int, int, int);
      void setRelogio(int, int, int);
      void printRelogio();

      private:
      int h, m, s;
};
```

Relogio.cpp

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

Relogio::Relogio(int hr=0, int min=0, int seg=0)
{
        setRelogio (hr, min, seg);
}
void Relogio::setRelogio(int hr, int min, int seg)
{
      h = hr;
      m = min;
      s = seg;
}
void Relogio::printRelogio()
{
      cout<<h<<':'<<m<<':'<<s;
}
```

Composição

- Note que quando temos um construtor com parâmetros padronizados e separamos a interface da implementação, só utilizamos os parâmetros padronizados na implementação;
- Depois de definida e implementada a classe *Relogio*, podemos utilizar objetos dela em outra classe
  - Caracterizando a **composição** ou **agregação**;
  - Qual a diferença entre as duas?

RelogioComAlarme.h

```
#include<iostream>
#include<string>
using namespace std;
#include"Relogio.h"

class RelogioComAlarme
{
	public:
		RelogioComAlarme();
		void setAlarme(string, bool, int, int, int);
		void printAlarme();

	private:
		bool ligado;
		Relogio alarme;
		string tom;
};
```

Composição

- Na classe *RelogioComAlarme*, um dos atributos é um objeto da classe *Relogio*
  - Quando um objetos *RelogioComAlarme* for destruído, o objeto *Relogio* também será;
  - Caracterizando assim uma **composição**.

RelogioComAlarme.cpp

```
#include<iostream>
#include<string>
using namespace std;
#include"RelogioComAlarme.h"

RelogioComAlarme::RelogioComAlarme():alarme(10, 10, 10)
{
	ligado = false;
	tom = "Battery";
}

void RelogioComAlarme::setAlarme(string musica, bool flag, int h, int m, int s)
{
	ligado = flag;
	alarme.setRelogio(h, m, s);
	tom = musica;
}

void RelogioComAlarme::printAlarme()
{
	alarme.printRelogio();
}
```

Composição

- Há um detalhe importante no construtor da classe *RelogioComAlarme*:
  - Precisamos chamar o construtor do objeto da classe *Relogio* também;
  - Fazemos isso depois da assinatura da implementação do construtor;
  - Colocamos **:** e depois chamamos o construtor do objeto da composição
    - Mesmo que não haja parâmetros.
- O objeto da composição pode ser utilizado normalmente, chamando seus próprios métodos
  - Não é possível acessar os membros privados do objeto da composição, exceto por *getters* e *setters*.

driverRelogioComAlarme.cpp

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

int main()
{
	RelogioComAlarme despertador;
	despertador.setAlarme("Enter Sandman", true, 6, 0, 0);
	despertador.printAlarme();
	return 0;
}
```

Composição

- Novamente, compilamos as implementações das classes usando a *flag* –c
  - Depois compilamos o programa principal;
  - g++ -c Relogio.cpp;
  - g++ -c RelogioComAlarme.cpp;
  - g++ *.o driverRelogioComAlarme.cpp –o programa.