
LABORATÓRIO 12

CONVERSÕES DE TIPO PARA CLASSES

EXERCÍCIOS DE REVISÃO

VOCÊ DEVE ACOMPANHAR PARA REVISAR OS CONCEITOS IMPORTANTES

1. O uso da conversão no código abaixo vai causar a criação de um **objeto temporário** da classe Tempo que será atribuído a variável t e depois descartado.

```
Tempo t { 4, 25 };  
cout << t << '\n';  
t = 3; // cria objeto temporário  
cout << t << '\n';
```

Para visualizar este processo acontecendo, adicione um destrutor a classe Tempo e exiba uma mensagem no construtor e destrutor para ver quando e quantas vezes cada um deles é chamado.

```
class Tempo  
{  
private:  
    int horas;  
    int minutos;  
  
public:  
    Tempo(int h = 0, int m = 0);  
    operator double();  
    operator int();  
  
    Tempo operator+(const Tempo& t) const;  
    friend Tempo operator+(const Tempo& t, int h);  
    friend Tempo operator+(int h, const Tempo& t);  
    friend ostream& operator<<(ostream& os, const Tempo& t);  
};
```

2. Considerando as várias formas de inicializar um objeto em C++, teste quais delas usam **conversões implícitas**. Para isso modifique todos os construtores com a palavra-chave explicit.

```
Tempo a = Tempo(2,10); // #1  
Tempo b (2,10);        // #2  
Tempo c = {2,10};       // #3  
Tempo d {2,10};         // #4
```

EXERCÍCIOS DE FIXAÇÃO

VOCÊ DEVE FAZER OS EXERCÍCIOS PARA FIXAR O CONTEÚDO

1. Suponha que a classe `Tempo` tenha sido definida sem as funções amigas que fazem soma com valores inteiros e sem as funções de conversão, como mostrado abaixo.

```
class Tempo
{
private:
    int horas;
    int minutos;

public:
    Tempo(int h = 0, int m = 0);
    operator double();
    operator int();

    Tempo operator+(const Tempo& t) const;
    friend Tempo operator+(const Tempo& t, int h);
    friend Tempo operator+(int h, const Tempo& t);
    friend ostream& operator<<(ostream& os, const Tempo& t);
};
```

Verifique o que acontece em cada uma das situações a seguir. Se necessário adicione a exibição de mensagens dentro das funções ou use o depurador para acompanhar a execução do programa passo a passo.

- a) Ainda é possível fazer as somas com inteiros do exemplo a seguir?

```
int main()
{
    Tempo a { 1,10 };
    Tempo b, c;

    b = a + 2;
    c = 2 + a;

    cout << b << endl;
    cout << c << endl;
}
```

- b) E se substituirmos a sobrecarga de `operator+()` por uma função amiga?

```
friend Tempo operator+(const Tempo& t1, const Tempo& t2);
```

- c) E se adicionarmos de volta a função de conversão para inteiros?

```
operator int();
```

- d) E se removermos a função `operator+()` mantendo a função de conversão?

```
friend Tempo operator+(const Tempo& t1, const Tempo& t2);
operator int();
```

EXERCÍCIOS DE APRENDIZAGEM

VOCÊ DEVE ESCREVER PROGRAMAS PARA REALMENTE APRENDER

1. Implemente uma classe Horário para representar horas e minutos do relógio. Forneça funções amigas para executar a operação de subtração de horários e a exibição de horários com o cout.

```
class Horário
{
private:
    int horas;
    int minutos;

public:
    Horário(int h = 0, int m = 0);

    friend Horário operator-(const Horário& h1, const Horário& h2);
    friend ostream& operator<<(ostream& os, const Horário& h);
};
```

Use uma classe Tempo simplificada, apenas com funções amigas para a operação de soma e de exibição com cout, como mostrado abaixo:

```
class Tempo
{
private:
    int horas;
    int minutos;

public:
    Tempo(int h = 0, int m = 0);

    friend Tempo operator+(const Tempo& t1, const Tempo& t2);
    friend ostream& operator<<(ostream& os, const Tempo& t);
};
```

Primeiro verifique a mensagem de erro apresentada pelo compilador para o programa abaixo. Depois implemente uma função de conversão de Horário para Tempo de forma que o programa funcione.

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

int main()
{
    Horário inicio { 18, 30 };
    Horário fim { 21, 00 };

    // Horário convertido para Tempo
    Tempo duracao = fim - inicio;
    cout << duracao << endl;
}
```