

# Programação Orientada a Objetos

Prof. Hugo Marcondes  
hugo.marcondes@ifsc.edu.br

Aula 05

Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina

---

---

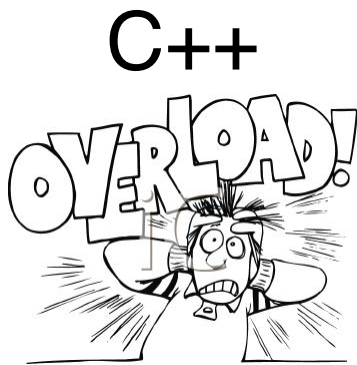
---

---

---

---

---



<sup>2</sup> IFSC - Programação Orientada a Objetos

---

---

---

---

---

---

---

## C++ Overloading

- C++ permite a sobrecarga (overloading) de funções e operadores
  - Function Overloading
  - Operator Overloading
- Uma declaração sobrecarregada é nada menos que declaração com mesmo nome, dentro de um mesmo escopo (namespace ou class), contudo, com parâmetros e argumentos diferentes, e implementações diferentes.
- Ao chamar uma função ou operador sobrecarregado, o compilador irá determinar qual é a definição mais apropriada através da comparação dos tipos dos argumentos da chamada utilizada.

<sup>3</sup> IFSC - Programação Orientada a Objetos

---

---

---

---

---

---

---

## Sobrecarga de Funções

- Definição da mesma função no mesmo escopo
  - Diferem pelo número e tipo dos parâmetros da função
  - O tipo de retorno não é considerado !

```
class printData
{
public:
    void print(int i) {
        cout << "Printing int: " << i << endl;
    }

    void print(double f) {
        cout << "Printing float: " << f << endl;
    }

    void print(char* c) {
        cout << "Printing character: " << c << endl;
    }
};
```

```
int main(void)
{
    printData pd;

    // Call print to print integer
    pd.print(5);
    // Call print to print float
    pd.print(500.263);
    // Call print to print character
    pd.print("Hello C++");

    return 0;
}
```

<sup>4</sup> IFSC - Programação Orientada a Objetos

---

---

---

---

---

---

---

## Sobrecarga de Operadores



- Você pode realizar a sobrecarga da maioria dos operadores disponíveis em C++
- Uso em tipos (classes) definidos pelo usuário
- Operadores são sobrecarregados são funções que possuem o modificador "operator" seguido do símbolo do operador sobrecarregado.

Keyword      Operator to be overloaded

```
ReturnType classname :: Operator OperatorSymbol (argument list)
{
    // Function body
}
```

5 IFSC - Programação Orientada a Objetos

---

---

---

---

---

---

---

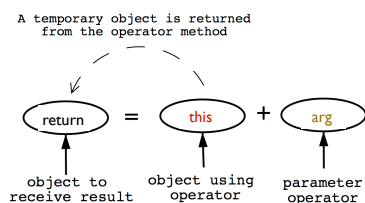
---

## Sobrecarga de Operadores



- A maioria dos operadores sobrecarregados podem ser definidos como funções não membro de uma classe, ou como uma função membro da classe.

```
Box operator+(const Box &arg);
```



6 IFSC - Programação Orientada a Objetos

---

---

---

---

---

---

---

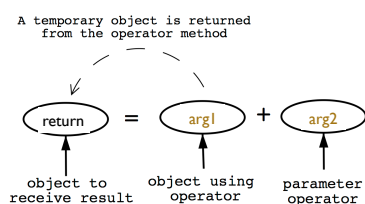
---

## Sobrecarga de Operadores



- A maioria dos operadores sobrecarregados podem ser definidos como funções não membro de uma classe, ou como uma função membro da classe.

```
Box operator+(const Box &arg1, const Box &arg2);
```



7 IFSC - Programação Orientada a Objetos

---

---

---

---

---

---

---

---

## Sobrecarga de Operadores



```
class Box {
public:
    Box(double len, double bre, double hei) {
        this->length = len;
        this->breadth = bre;
        this->height = hei;
    }
    Box() {
        this->length = 0;
        this->breadth = 0;
        this->height = 0;
    }
    double getVolume(void) {
        return length * breadth * height;
    }
    Box operator+(const Box& b) {
        Box box;
        box.length = this->length + b.length;
        box.breadth = this->breadth + b.breadth;
        box.height = this->height + b.height;
        return box;
    }
private:
    double length; // Length of a box
    double breadth; // Breadth of a box
    double height; // Height of a box
};
```

```
int main() {
    Box Box1(6.0, 7.0, 5.0);
    Box Box2(12.0, 13.0, 10.0);
    Box Box3;

    cout << "Volume of Box1 : ";
    cout << Box1.getVolume() << endl;

    cout << "Volume of Box2 : ";
    cout << Box2.getVolume() << endl;

    cout << "Volume of Box3 : ";
    cout << Box3.getVolume() << endl;

    cout << "Box3 = Box1 + Box2" << endl;
    Box3 = Box1 + Box2;

    cout << "Volume of Box3 : ";
    cout << Box3.getVolume() << endl;

    return 0;
}
```

8 IFSC - Programação Orientada a Objetos

---

---

---

---

---

---

---

---

## Operadores sobrecarregáveis



Operator Category	Operators
Arithmetic	+, -, *, /, %
Bit-Wise	&,  , ^, ~, ^
Logical	&&,   , !
Relational	<, >, ==, !=, <=, >=
Assignment	=
Arithmetic assignment	+=, -=, *=, /=, %=, &=,  =, ^=
Shift	>>, <<, >>=, <<=
Unary	++, --
Subscripting	[]
Function call	()
Dereferencing	->
Unary sign prefix	+, -
Allocate and free	new, delete

Table 9.1 C++ Overloadable Operators

9 IFSC - Programação Orientada a Objetos

## Exemplo: Output and Input Streams



```
class Box {
public:
    Box operator+(const Box& b) {
        Box box;
        box.length = this->length + b.length;
        box.breadth = this->breadth + b.breadth;
        box.height = this->height + b.height;
        return box;
    }

    friend ostream& operator<<(ostream& output, const Box& b) {
        output << "Box(" << b.length << ", " << b.breadth << ", " << b.height << ")";
        return output;
    }

    friend istream& operator>>(istream& input, Box& b) {
        cout << "Length ? ";
        input >> b.length;
        cout << "Breadth ? ";
        input >> b.breadth;
        cout << "Height ? ";
        input >> b.height;
        return input;
    }
};
```

10 IFSC - Programação Orientada a Objetos

## Exemplo: Output and Input Streams



```
int main() {
    Box Box1;
    Box Box2;
    Box Box3;

    cout << "We have 3 null boxes: " << Box1 << ", " << Box2 << ", " << Box3 << endl;

    cout << "Box1\n";
    cin >> Box1;
    cout << "Box2\n";
    cin >> Box2;

    cout << "Volume of " << Box1 << " " << Box1.getVolume() << endl;
    cout << "Volume of " << Box2 << " " << Box2.getVolume() << endl;
    cout << "Volume of " << Box3 << " " << Box3.getVolume() << endl;

    cout << "Box3 = Box1 + Box2" << endl;
    Box3 = Box1 + Box2;

    cout << "Volume of " << Box3 << " " << Box3.getVolume() << endl;

    return 0;
}
```

11 IFSC - Programação Orientada a Objetos

## Calculadora de Números Complexos



- Implemente uma calculadora simples (operações de soma, subtração, multiplicação e divisão)



12 IFSC - Programação Orientada a Objetos