VIII. Conversões Implícitas

Construtores de Tipos

Operadores de Atribuição

Conversões de Classes

- Usar um construtor para especificar uma conversão de tipos é conveniente
  - mas tem implicações indesejáveis
- Um construtor não pode fazer:
  - uma conversão implícita entre um tipo definido pelo usuário e um tipo básico;
    - porque tipos básicos não são classes.
  - a conversão de uma nova classe a uma classe previamente definida;
    - sem modificar a declaração da classe antiga.

Como fazer as seguintes conversões?

- Fazer as seguintes conversões entre a classe String e tipos básicos da linguagem:
  - Int –> String
  - Double –> String
  - String –> int
  - String –> double

Exemplo

```
int main() {
    int i;
    double d;
    String s1, s2;
    s1 = i;
    s2 = d;
    i = s1;
    d = s2;
}
```

Operador de Conversão

- Uma função membro **X::operator T()**, onde T é um nome de tipo, define a conversão de X para T.

```
class String {
   operator int() const;
   operator double() const;
};

int main() {
   i1 = s1;
   d1 = s1;
}
```

Operador de Atribuição

- Uma função membro **X::operator =(T)**, onde T é um nome de tipo, define a conversão de T para X.

```
class String {
   operator =(int);
   operator =(double);
};

int main() {
   i1 = s1;
   d1 = s1;
}
```

EXEMPLO 1

Conversões Objetos $\Rightarrow$ Tipos Básicos

- Teste no main():

```
String s10("123"), s11("12.34");
int i = s10;
double d = s11;
```

Conversões Objetos $\Rightarrow$ Tipos Básicos

- Teste no main():

  String s10(“123”), s11(“12.34”);

  int i = s10;

  double d = s11;

- Agora faça e teste novamente:

  operator **int**() const { return atoi(buffer); }

  operator **double**() const { return atof(buffer); }

Conversões Tipos Básicos $\Rightarrow$ Objetos

- Teste no main()

  s10 = f;

  s11 = i;

Conversões Tipos Básicos ⇒ Objetos

- Teste no main()

  s10 = f;

  s11 = i;

- Agora faça e teste novamente:

  void operator =(int n) { itoa(n, buffer, 10); }

  void operator =(double d) { ftoa(d, buffer, 10); }

Não funciona no C++11

Conversões Tipos Básicos $\Rightarrow$ Objetos

- Teste no main()

  s10 = f;

  s11 = i;

- Agora faça e teste novamente:

  void operator **=**(int n) { buffer = to_string(n); }

  void operator **=**(double d) {buffer = to_string(d); }

EXEMPLO 2

Conversões entre objetos de classes diferentes

- Teste no main()

```
String s1 = “321”, s2;  // nosso objeto
string z1, z2=“prog”;  // classe do c++
z1 = s1;
s2 = z2;
```

Conversões entre objetos de classes diferentes

- Teste no main()

```
String s1 = “321”, s2;
string z1, z2=“prog”;
z1 = s1;
s2 = z2;
```

- Agora faça e teste novamente:

```
String(string z) { strcpy(buffer, z.c_str()); }

operator string() {
   string z = buffer;
   return z;
}
```

EXERCÍCIOS

Exercícios

1) Vamos estender a classe FRAÇÃO. Esta classe deve armazenar o numerador e o denominador da fração, como 2 números inteiros. Inclua:
   a) dois construtores: o primeiro sem argumentos, e o outro que recebe numerador e denominador da fração.
   b) dois getters
   c) as operações de adição, subtração, multiplicação e divisão
   d) as operações de incremento e decremento, pré e pós-fixadas
   e) um conversor de tipo, que converte um objeto fração para um tipo float
   f) um operador de atribuição do tipo int para fração

Classe Fração

2) Fazer um construtor que recebe um "float/double" e cria uma fração, ou diz que é impossível (caso seja dízima não periódica).

3) Fazer a operação de comparação de igualdade (como devolver que ½, 2/4 e 3/6 são equivalentes?)
   - DICA: faça uma rotina que simplifica a fração até que, tanto numerador, quanto denominador, sejam números primos.