Resolução de Sobrecarga em C++

Introdução

A resolução de sobrecarga em C++ refere-se ao processo pelo qual o compilador seleciona a função apropriada a ser chamada quando várias funções com o mesmo nome estão disponíveis. Este mecanismo permite a criação de funções com o mesmo nome, mas com diferentes listas de parâmetros, aumentando a flexibilidade e a legibilidade do código.

1. Definição e Sintaxe

- Definição: A resolução de sobrecarga é o processo de determinar qual função, entre várias sobrecarregadas, deve ser chamada com base nos argumentos fornecidos na chamada da função.
- Sintaxe:

```
void funcao(int a);
void funcao(double b);
void funcao(int a, double b);
```

2. Exemplo de Sobrecarga de Função

- Exemplo:

```
void imprimir(int a) {
    std::cout << "Inteiro: " << a << std::endl;
}

void imprimir(double b) {
```

```
    std::cout << "Double: " << b << std::endl;
}

void imprimir(int a, double b) {
    std::cout << "Inteiro: " << a << ", Double: " << b << std::endl;
}

int main() {
    imprimir(5);        // Chama imprimir(int)
    imprimir(3.14);     // Chama imprimir(double)
    imprimir(5, 3.14);  // Chama imprimir(int, double)
    return 0;
}
```

3. Preferência na Resolução de Sobrecarga

- Definição: O compilador segue uma série de regras para determinar qual função chamar. As preferências incluem a correspondência exata, a promoção de tipos e a conversão de tipos.
- Exemplo:

```
void funcao(int a) {
    std::cout << "int" << std::endl;
}

void funcao(double b) {
    std::cout << "double" << std::endl;
}
```

```
void funcao(float c) {
    std::cout << "float" << std::endl;
}

int main() {
    funcao(10);       // Chama funcao(int)
    funcao(10.0);     // Chama funcao(double)
    funcao(10.0f);    // Chama funcao(float)
    return 0;
}
```

4. Conversões Padrão e Promocionais

- Definição: O compilador pode realizar conversões padrão (como de `int` para `double`) e promoções (como de `float` para `double`) para encontrar a correspondência mais adequada.
- Exemplo:

```
void funcao(long a) {
    std::cout << "long" << std::endl;
}

void funcao(double b) {
    std::cout << "double" << std::endl;
}

int main() {
```

```
    funcao(10);      // Chama funcao(long), pois a promoção de int para long é preferida
    funcao(10.0);    // Chama funcao(double)
    return 0;
}
```

5. Funções Template e Resolução de Sobrecarga

- Definição: Funções template podem ser sobrecarregadas com funções não-template, e a resolução de sobrecarga considera as especializações dos templates.
- Exemplo:

```
template<typename T>
void funcao(T a) {
    std::cout << "Template" << std::endl;
}

void funcao(int a) {
    std::cout << "int" << std::endl;
}

int main() {
    funcao(10);      // Chama funcao(int)
    funcao(10.0);    // Chama funcao<double> (template)
    return 0;
}
```

6. Ambiguidade na Resolução de Sobrecarga

- Definição: Ambiguidade ocorre quando o compilador não consegue determinar qual função chamar devido a múltiplas correspondências igualmente válidas.

- Exemplo:

```
  void funcao(int a) {
      std::cout << "int" << std::endl;
  }

  void funcao(double b) {
      std::cout << "double" << std::endl;
  }

  void funcao(float c) {
      std::cout << "float" << std::endl;
  }

  int main() {
      // funcao(10.0f);  // Ambiguidade: pode chamar funcao(float) ou funcao(double)
      return 0;
  }
```

7. Resolução de Sobrecarga com Funções Membros

- Definição: A resolução de sobrecarga também se aplica a funções membros de classes, levando em conta o tipo do objeto que chama a função.

- Exemplo:

```
class Classe {
public:
    void funcao(int a) {
        std::cout << "int" << std::endl;
    }

    void funcao(double b) {
        std::cout << "double" << std::endl;
    }
};

int main() {
    Classe obj;
    obj.funcao(10);      // Chama funcao(int)
    obj.funcao(10.0);    // Chama funcao(double)
    return 0;
}
```

Dicas de Boas Práticas

- Clareza: Evite sobrecarregar funções com assinaturas que podem causar ambiguidade.
- Documentação: Documente claramente as funções sobrecarregadas para facilitar a compreensão e manutenção.
- Teste: Teste as chamadas de função para garantir que a função correta está sendo chamada.

Esta seção abrange os conceitos sobre resolução de sobrecarga em C++. Para mais detalhes,

consulte a documentação oficial: https://en.cppreference.com/w/cpp/language/overload_resolution