Programação Orientada a Objetos em C++

Tratamento de Exceções

Agostinho Brito

2020

O que fazer quando ocorre um erro?

```
1  float media_harmonica(float a, float b){
2    return 2*a*b/(a+b);
3  }
4  int main(void){
5    float a, b;
6    std::cin >> a >> b;
7    std::cout << media_harmonica(a,b) << std::endl;
8  }
```

1. Torcer para nunca acontecer... 💣
2. Reescrever a função para lidar com códigos de erro.
3. Finalizar o programa via `exit()`.
4. Usar tratamento de exceções... em C++.

Lançando exceções

- Em C++, o tratamento de exceções pode ser feito para decidir o que fazer quando situações incomuns ocorrem no programa.

```
1  float media_harmonica(float a, float b){
2    if(a+b == 0){
3      throw "a = -b (denominador nulo)";
4    }
5    return 2*a*b/(a+b);
6  }
7  int main(void){
8    float a, b;
9    std::cin >> a >> b;
10   try{
11     std::cout << media_harmonica(a,b) << std::endl;
12   }
13   catch(const char* e){
14     std::cout << "erro: " << e << std::endl;
15   }
16 }
```

Sintaxe completa

```
1  try{
2    // codigo passivel de condicoes excepcionais
3  }
4  catch(tipo1 var1){
5    // opera como uma funcao com um unico parametro
6  }
7  catch(tipo2 var2){
8    // cada catch trata um tipo de excecao lancada
9  }
10 catch(...){
11   // excecao default
12   // ativada quando o tipo lancado
13   // nao combina com os outros catches
14 }
```

Praticando tratamento de exceções...

A classe `exception` - implementação padrão

```
1  class Excecao : public std::exception {
2    virtual const char* what() const throw() {
3      return "Lancou excecao";
4    }
5  };
6  int main() {
7    Excecao ex;
8    try {
9      throw ex;
10   } catch (std::exception& e) {
11     std::cout << e.what() << std::endl;
12   }
13   return 0;
14 }
```

- O método `what()` deve ser provido, pois retorna a causa da exceção.

Obrigado