Tratamento de Exceções

Prof. Edwar Saliba Júnior
Fevereiro de 2011

Conceito

- **Robustez**: capacidade de um *software* continuar seu processamento mesmo sob condições anormais;
- Problemas: Como prevenir-se de erros em tempo de execução? Como prevenir-se de situações anormais?
- Exemplos de situações anormais:
  - Divisão por zero, fim de arquivo, *overflow*, utilização de um objeto não instanciado, acesso a um índice inválido de um vetor.

Exceção

- É um evento ocorrido durante a execução normal de um programa que desvia o fluxo normal de execução;

- É uma condição provocada por uma situação excepcional, que requer uma ação específica imediata.

(Varejão, 2004)

Mecanismo para Tratar Exceções

- Existem linguagens de programação que não possuem tais mecanismo. Como por exemplo:
  - Linguagem C e
  - Linguagem Pascal;
- Neste caso o programador deve produzir código para fazer o tratamento das possíveis exceções.
- Exemplo:

```
if(x != 0)
   res = y / x;
else
   printf("Erro de divisão por zero!");
```

Mecanismo para Tratar Exceções

- Por outro lado, existem linguagens que possuem tais mecanismos. Como:
  - Delphi,
  - C#,
  - Java e etc.;
- Benefícios:
  - Diminuição da sobrecarga do texto do programa;
  - Maior comodidade, para o programador, no tratamento das condições anormais.

Exemplo de Exceções em Java

- `NullPointerException` – ocorre quando a operação realizada gera um acesso a uma área não alocada (nula);
- `IndexOutOfBoundException` – ocorre quando se tenta acessar um índice de um vetor que está fora de seus limites.

Lançamento de Exceções

- O lançamento (ou sinalização) de exceções pode ser realizado da seguinte forma:
  - **Automaticamente**: o próprio mecanismo existente na linguagem é o responsável por lançar a exceção. Pode ocorrer em qualquer ponto do programa passível de geração de situação anormal.
    - Exemplo: o usuário do *software* entra com um valor do tipo `string` num campo que espera um valor do tipo `float`.
  - **Explicitamente**: o programador escreve código para lançar uma possível exceção e consequentemente capturá-la.

Lançamento de Exceções

- Em casos específicos um programador pode fazer o lançamento de exceções que não são tratadas pela JVM;
- O exemplo a seguir mostra um lançamento de exceção explícito em Java. Para tal, emprega-se o comando `throw`;

```
throw new Exception();
```

Tratamento de Exceções

- Exceções podem ser tratadas a partir de trechos de códigos que tomam determinadas atitudes quando da ocorrência da anormalidade;
- Em Java para tratar uma exceção usa-se a instrução `try`:
  - A instrução `try` é composta por três blocos:
    - bloco `try`,
    - bloco `catch` e o
    - bloco `finally`. (Facultativo.)

Exemplo

- Exemplo .pdf

Tratamento de Exceções

- No exemplo anterior a cláusula `catch(NumberFormatException x)` captura exceções de formatação numérica;
- A cláusula `catch(ArithmeticException y)` captura exceções ocorridas em operações aritméticas e a
- Cláusula `catch(Exception z)` captura qualquer tipo de exceção ocorrida dentro da instrução `try`;
- Os comandos dentro dos blocos `catch` são o tratamento das possíveis exceções;
- Já as variáveis `x`, `y` e `z`, cada qual, contém a sua respectiva exceção no idioma da linguagem de programação, ou seja, o inglês, mais os detalhes técnicos da mesma.

Propagação de Exceções

- Quando uma exceção ocorre a JVM busca pela cláusula `catch` associada ao seu tipo. A sequência para realização da busca é a que aparece no código;
- Quando uma exceção não é tratada no bloco em que ocorreu, ela é propagada para o bloco mais externo;
- No caso de chamada de métodos, é propagada para o método chamador.

Propagação de Exceções

- Se o bloco ou método para o qual a exceção foi propagada não fizer seu tratamento, então a exceção continua sendo propagada;
- Se a exceção chegar ao método principal (`main`) e também não for tratada, então o programa é abortado.

Relançamento de Exceções

- Em alguma situações, pode ser necessário que o local onde ocorreu a exceção a trate de maneira parcial, deixando o restante para blocos mais externos;
- Neste caso, utiliza-se o recurso de relançamento de exceções.

Exemplo

```
...
try{
   try{
      // Aqui pode ocorrer uma exceção do tipo IOException.
   }
   catch(IOException e){
      ...          // Tratamento parcial.
      throw e;     // Relançamento da exceção.
   }
}
catch(IOException e){
   ...   // Restante do tratamento.
}
```

Continuação após o Tratamento de Exceções

- Em algumas situações pode ser necessária a execução de um conjunto de comandos, independentemente de ter ocorrido uma exceção na instrução `try`;
- A cláusula `finally` de Java provê este recurso;
- Em geral, este recurso é utilizado quando deseja-se restabelecer o estado de algum objeto de forma independente da ocorrência e da propagação de exceções;
- Exemplo:
  - Encerramento conexões com banco de dados ou fechamento de arquivos, quando houver a ocorrência de determinadas exceções.

Continuação após o Tratamento de Exceções

- É possível a existência de um bloco `try` sem um bloco `catch`. Mas não é possível existir uma instrução `try` sem, pelo menos, um bloco `catch` ou um bloco `finally`;
- Havendo um bloco `finally` na instrução `try` e independente do que aconteça, o código dentro do bloco `finally` sempre será executado.

Exemplos

```
try{
   ...
   // Código qualquer.
   ...
}
finally{
   ...
   // Código qualquer.
   ...
}
```

```
try{
   // Código qualquer.
}
catch(Exception e){
   // Código qualquer.
}
finally{
   // Código qualquer.
}
```

Bibliografia

- DEITEL, H. M.; DEITEL, P. J. **Java Como Programar**; tradução Edson Furmankiewicz; revisão técnica Fábio Lucchini. 6a. ed., São Paulo: Pearson, 2005.
- FERREIRA, Kecia Aline Marques. *Slides* da disciplina de Programação de Computadores II. CEFET-MG, 2009.