Linguagem de Programação

JAVA

Tratamento de Exceções

Profa. Joyce Miranda

Tratamento de Exceções e Controle de Erros

- **Tópicos**

Tratamento de Exceções

- **O que é uma exceção?**
- **O que é uma falha?**

Tratamento de Exceções

- **Exemplos de falhas:**
  - Falha de Memória
  - Erro de I/O
  - Arrays fora de faixa
  - Valores de variáveis (Divisão por Zero)
  - Erros da aplicação
    - Saldo insuficiente
    - Usuário não existe
    - Nota inválida

Tratamento de Exceções

- **Encontrar erro é desagradável.**

- **O que deve ser feito?**
  - Notificar o usuário de um erro;
  - Permitir que o usuário saia elegantemente do programa;
  - Conseguir salvar todo o trabalho.

Tratamento de Exceções

- **Na programação estruturada...**

```
void f() {
     if (<teste da condição de erro 1>) {
          <comandos que determinam o que fazer se o erro 1 ocorreu>
     }

     else if (<teste da condição de erro 2>) {
          <comandos que determinam o que fazer se o erro 2 ocorreu>
     }

     else if....<testando e tratando outros possíveis erros>

     else {
          <comandos para processar a função em condições normais>
     }
}
```

Tratamento de Exceções

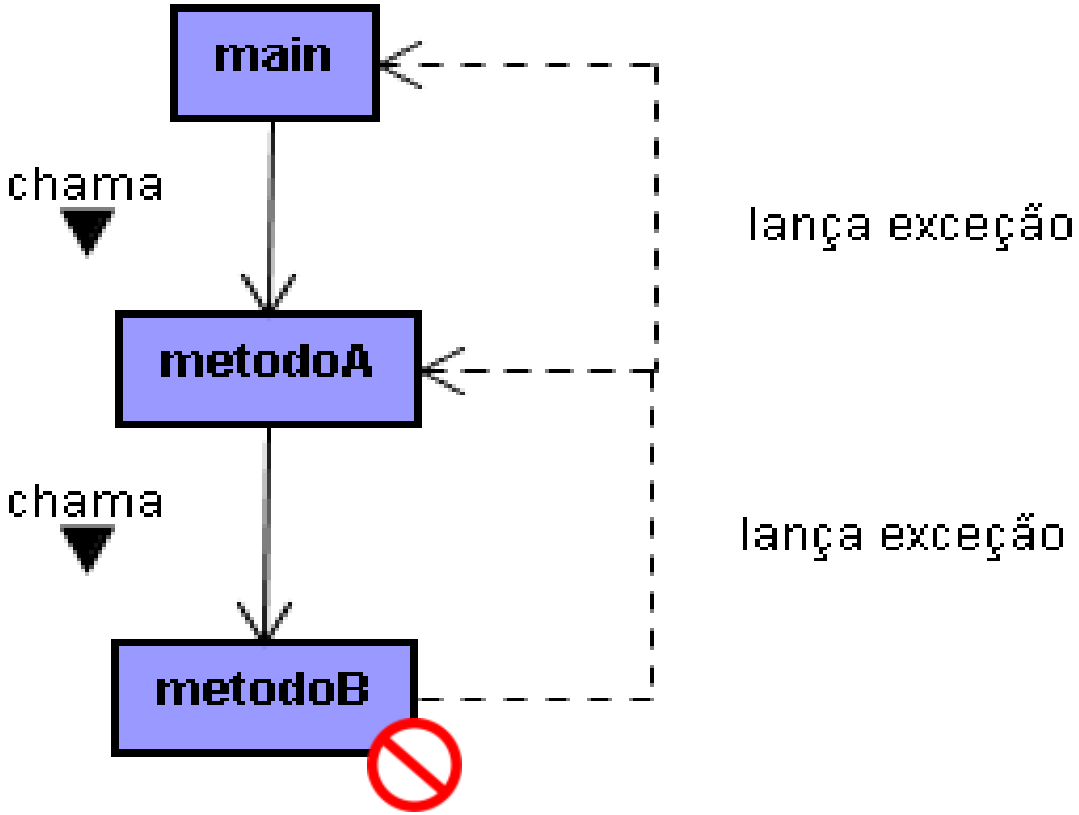
- **Na orientação a objetos (JAVA)**
  - Exceções são representadas por uma hierarquia particular de objetos.
  - Permite manter um código de manipulação de exceções nitidamente separado do código que gerará a exceção;

Tratamento de Exceções

exceções são lançadas e capturadas

Tratamento de Exceções

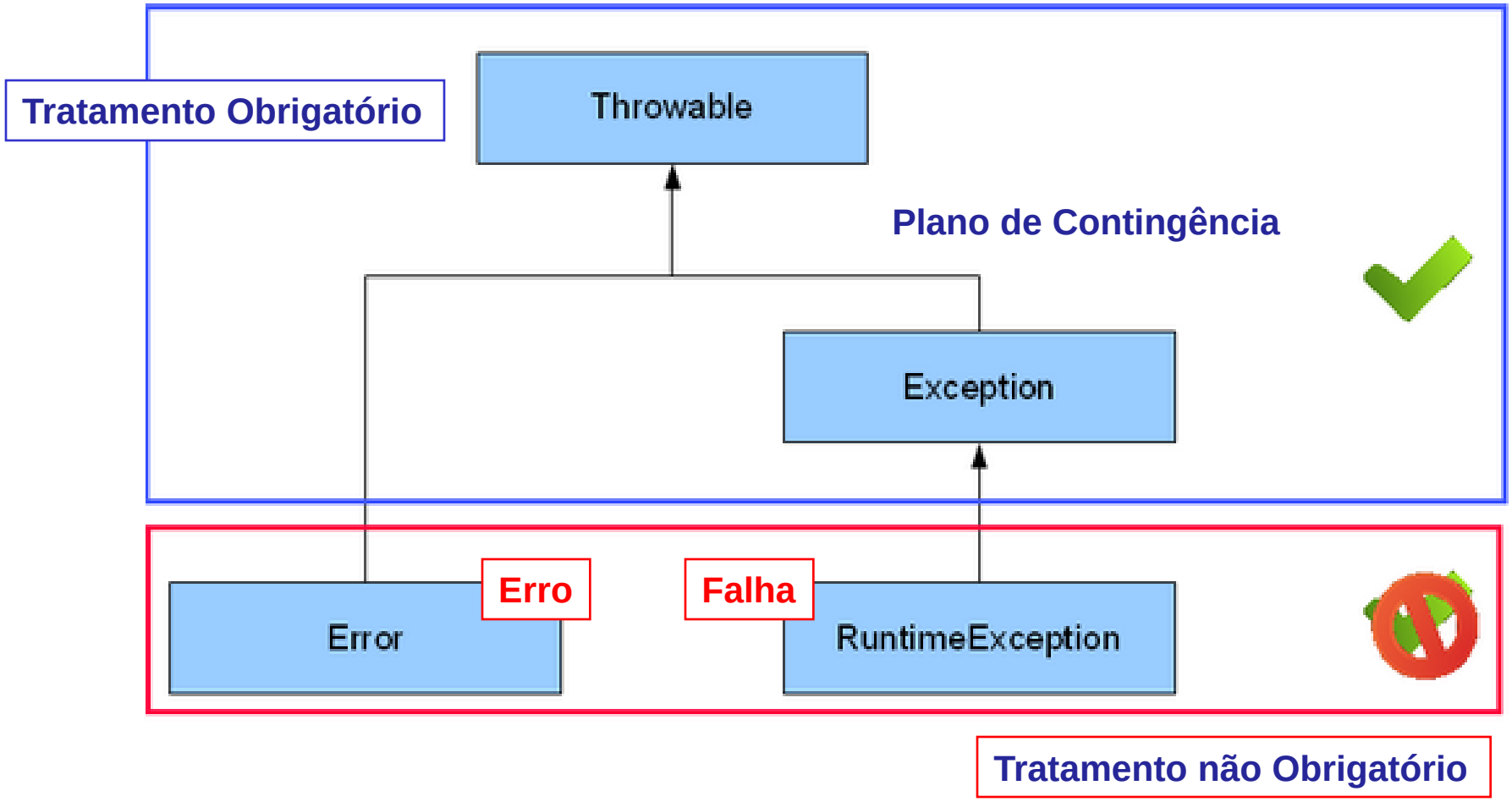
- **Invocação de Método X Lançamento de Exceções**

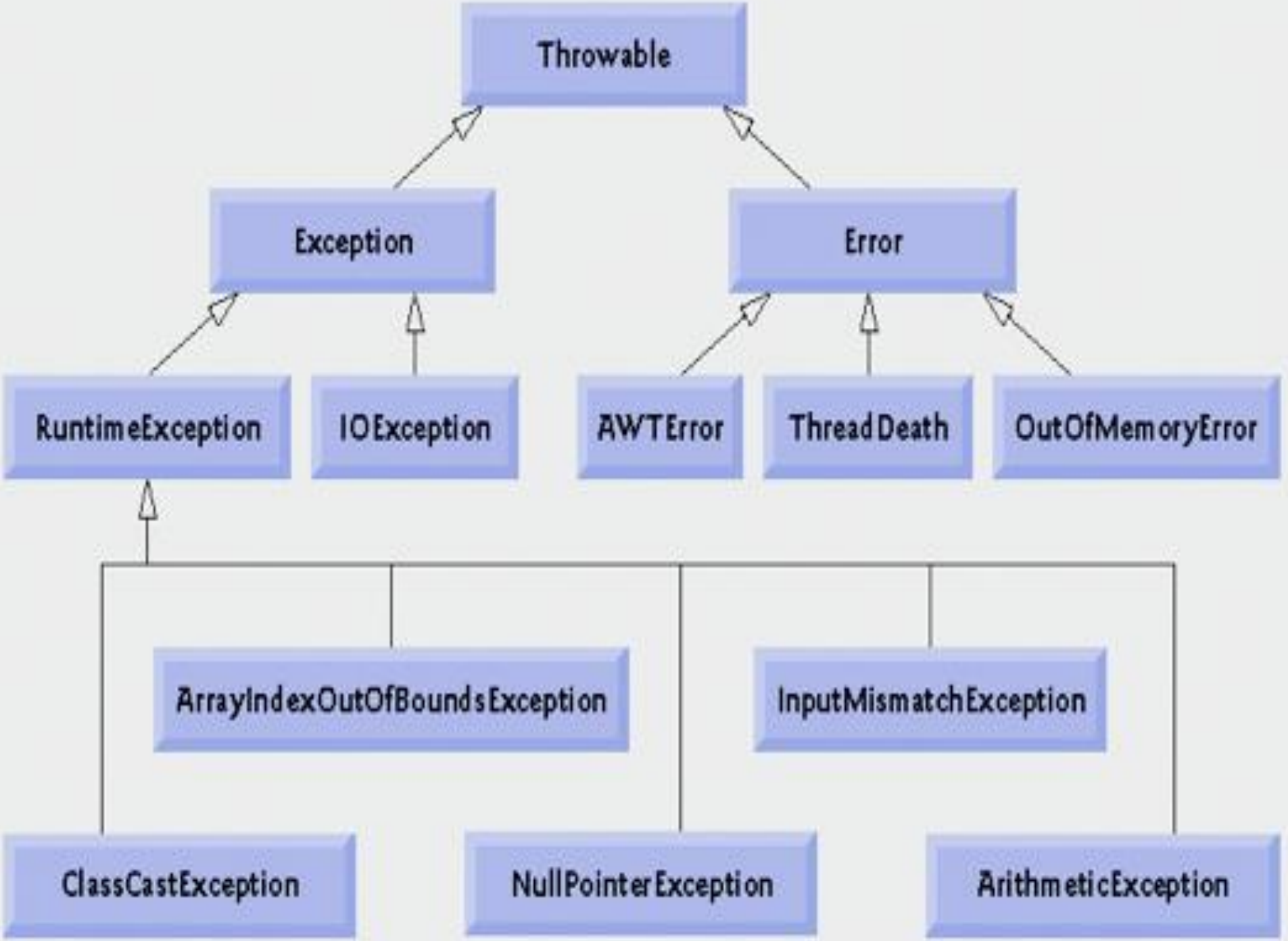


Trabalhar com exceções é decidir onde capturar as exceções e o que fazer uma vez que elas foram capturadas.

Tratamento de Exceções

- **Exceções Verificadas x Exceções Não Verificadas**




Throwable
Exception
Error
RuntimeException
IOException
AWTError
ThreadDeath
OutOfMemoryError
ArrayIndexOutOfBoundsException
InputMismatchException
ClassCastException
NullPointerException
ArithmeticException

Tratamento de Exceções

- **Erros (Error)**
  - Exceções graves e fatais.
  - Problemas de SO ou hardware.
  - Exemplos
    - OutOfMemory
    - StackOverflowError

Tratamento de Exceções

- **Falhas (RuntimeException)**
  - Podem ser tratadas pela aplicação, mas não é obrigatório.
    - Se não for tratada é capturada pela JVM
  - Causadas por problemas de Lógica de Programação.
  - Exemplos
    - IndexOutOfBounds
    - IllegalArgumentException
    - NullPointerException
    - NumberFormatException

Tratamento de Exceções

- **Exceções de Contingência (Exception)**
  - Devem ser tratadas pela aplicação, sendo obrigatório.
  - Exceções Previsíveis
  - Exemplos
    - IOException
      - FileNotFoundException
    - SQLException

Tratamento de Exceções

- **Capturando Exceções**

```
try {
    // Executa código que pode disparar exceção
} catch (Exception ex){
    // A excecao ocorreu.
    // Tentamos evitar o problema,
    // fazendo a operacao de forma diferente
} finally {
    // Este bloco é sempre executado,
    // independente de ocorrer exceção
}
```

Tratamento de Exceções

- **Capturando Exceções**

```
try {
  // aqui executamos um método que tenta ler um arquivo

}catch (FileNotFoundException   e) {
      // se o arquivo não existir esta exceção é lançada.

      // aqui colocamos a resolução
}catch (EOFException e) {
      // quando esta exceção acontece significa que aconteceu
      // um problema na leitura do arquivo.

      // aqui colocamos a resolução
} catch (IOException e) {
      // uma outra exceção de I/O aconteceu.

      // aqui colocamos a resolução
}
```

Tratamento de Exceções

- **Capturando Exceções - Exemplo**

```
static int vet[] = {0, 1, 2, 3, 4, 5};

public static void printVet(int pos){
     System.out.println(vet[pos]);
}

public static void main (String args[]){
     Scanner s = new Scanner(System.in);
     int pos = s.nextInt(); //entrada -1
     printVet(pos);
}
```

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: -1
```

Tratamento de Exceções

Capturando Exceções - Exemplo

```
public static void main (String args[]){
    boolean finalizou = false;
    while(!finalizou){
        try {
            Scanner s = new Scanner(System.in);
            int pos = s.nextInt();
            printVet(pos);
            finalizou = true;
        } catch (ArrayIndexOutOfBoundsException e) {
            System.err.println("Informe uma posicao válida!");
        }
    }
}
```

Tratamento de Exceções

Capturando Exceções - Exemplo

```
public static void main (String args[]){
    boolean finalizou = false;
    while(!finalizou){
        try {
            Scanner s = new Scanner(System.in);
            int pos = s.nextInt();
            printVet(pos); //entrada: S
            finalizou = true;
        } catch (ArrayIndexOutOfBoundsException e) {
            System.err.println("Informe uma posicao válida!");
        } catch (InputMismatchException e) {
            System.err.println("Informe um número!");
        }
    }
}
```

Tratamento de Exceções

- **Tipos Comuns de Exceção**
  - ArithmeticException
  - NumberFormatException
    - IllegalArgumentException
  - InputFormatException
  - IndexOutOfBounds
    - ArrayIndexOutOfBoundsException
    - StringIndexOutOfBounds
  - NullPointerException
  - ClassNotFoundException

Tratamento de Exceções

- **Lançando Exceções (throw)**

```
public static void sqrt(int value){
    if(value < 0){
        throw new IllegalArgumentException("Informe valores positivos!");
    }else{
        System.out.println(Math.sqrt(value));
    }
}
```

```
public static void main (String args[]){
        Scanner s = new Scanner(System.in);
        int pos = s.nextInt(); //entrada -1
        sqrt(pos);
}
```

```
Exception in thread "main" java.lang.IllegalArgumentException: Informe valores positivos!
```

Tratamento de Exceções

- **Repassando Responsabilidades (throws)**

```
public static void sqrt(int value) throws IllegalArgumentException {
    if(value < 0){
        throw new IllegalArgumentException();
    }else{
        System.out.println(Math.sqrt(value));
    }
}
public static void main (String args[]){
    boolean finalizou = false;
    while(!finalizou){
        Scanner s = new Scanner(System.in);
        int pos = s.nextInt(); //entrada -1
        try {
          sqrt(pos);
          finalizou = true;
        } catch (IllegalArgumentException e) {
            System.err.println("Informe valores positivos!");
        }
    }
}
```

Tratamento de Exceções

- **Criando Exceções**
  - **Passo 1: Crie uma classe que herde de Exception ou RuntimeException**

```
public class ContatoNaoEncontradoException extends Exception {
        public ContatoNaoEncontradoException()
        {
                super("Contato não encontrado");
        }
}
```

Tratamento de Exceções

- **Criando Exceções**
  - **Passo 2: No método que dispara a exceção**
    - **Coloque a cláusua throws**
    - **Crie o objeto da classe de exceção**
    - **Dispare (throw)**

```
public Contato buscar(String nome)
 throws ContatoNaoEncontradoException{
   // Laco para procurar contato pelo nome
   for (int i = 0 ; i < quantidade ; i++)
      if (contatos[i].nome().equals(nome))
   return contatos[i];
   // Se sair do laço e não tiver retornado
   // o contato não esta cadastrado
   // deve disparar excecao
   throw new ContatoNaoEncontradoException();
}
```

Tratamento de Exceções

- **Criando Exceções**
  - **Passo 3: Tratar exceção na chamada do método**

```
private void buscarContato() {
    System.out.print("Digite o nome: ");
    String nome = sc.nextLine();

    try {
        Contato contato = agenda.buscar(nome);
    } catch (ContatoNaoEncontradoException e) {
        //tratamento de exceção
    }
}
```

Tratamento de Exceções

- **Exercícios**
  - Crie a classe Operacao, defina dois atributos valorA e valorB referentes a dados numéricos. Em seguida, defina um método que seja responsável por retornar o resultado da divisão entre os dois números.
  - O método deve repassar a responsabilidade de tratamento de exceção para a classe que chamou o método.
  - Implemente o tratamento para as seguintes exceções:
    - Leitura de valores não numéricos
    - Leitura de valores negativos
    - Divisão por zero