BÀI TẬP 1: NGOẠI LỆ VÀ XỬ LÝ NGOẠI LỆ

Nội dung: Lập trình ngoại lệ và xử lý ngoại lệ cơ bản

Yêu cầu:

- **Hãy nhập, chạy, phân tích các chương trình xử lý ngoại lệ với các từ khóa try, catch, finally, throw, throws.**
- **Tham khảo và chạy các ví dụ khác theo link:** Exception Handling in Java | Java Exceptions - javatpoint.
- **Định nghĩa lớp ngoại lệ do người sử dụng tự định nghĩa và sử dụng nó.**

Bài 1: Sử dụng try…catch

a) Một try nhiều catch

Ví dụ 1:

```
public class TestMultipleCatchBlock {
   public static void main(String args[]) {
      try {
         int a[] = new int[5];
         a[5] = 30 / 0;
      } catch (ArithmeticException e) {
         System.out.println("task1 is completed");
      } catch (ArrayIndexOutOfBoundsException e) {
         System.out.println("task 2 completed");
      } catch (Exception e) {
         System.out.println("common task completed");
      }

      System.out.println("rest of the code...");
   }
}
```

Ví dụ 2:

```
public class MultipleCatchBlock2 {

    public static void main(String[] args) {

        try{
            int a[]=new int[5];

            System.out.println(a[10]);
           }
           catch(ArithmeticException e)
             {
              System.out.println("Arithmetic Exception occurs");
             }
           catch(ArrayIndexOutOfBoundsException e)
             {
              System.out.println("ArrayIndexOutOfBounds Exception occurs");
             }
           catch(Exception e)
```

```
                {
                 System.out.println("Parent Exception occurs");
                }
               System.out.println("rest of the code");
    }
}
```

Ví dụ 3:

```
public class MultipleCatchBlock4 {

    public static void main(String[] args) {

        try{
            String s=null;
            System.out.println(s.length());
           }
           catch(ArithmeticException e)
             {
              System.out.println("Arithmetic Exception occurs");
             }
           catch(ArrayIndexOutOfBoundsException e)
             {
              System.out.println("ArrayIndexOutOfBounds Exception occurs");
             }
           catch(Exception e)
             {
              System.out.println("Parent Exception occurs");
             }
           System.out.println("rest of the code");
    }
}
```

Ví dụ 4:

```
class MultipleCatchBlock5{
  public static void main(String args[]){
   try{
    int a[]=new int[5];
    a[5]=30/0;
   }
   catch(Exception e){System.out.println("common task completed");}
   catch(ArithmeticException e){System.out.println("task1 is completed");}
   catch(ArrayIndexOutOfBoundsException e){System.out.println("task 2 completed");}

   System.out.println("rest of the code...");
  }
}
```

b) Sử dụng try…catch lồng nhau:

Ví dụ 1:

```
public class TestException {
    public static void main(String args[]) {
```

```
        try {
            try {
                System.out.println("Thuc hien phep chia");
                int b = 39 / 0;
            } catch (ArithmeticException e) {
                System.out.println(e);
            }

            try {
                int a[] = new int[5];
                a[5] = 4;
            } catch (ArrayIndexOutOfBoundsException e) {
                System.out.println(e);
            }

            System.out.println("khoi lenh khac");
        } catch (Exception e) {
            System.out.println("xy ly ngoai le");
        }

        System.out.println("tiep tuc chuong trinh..");
    }
}
```

Ví dụ 2:

```
class Excep6{
 public static void main(String args[]){
  try{
    try{
     System.out.println("going to divide");
     int b =39/0;
    }catch(ArithmeticException e){System.out.println(e);}

    try{
    int a[]=new int[5];
    a[5]=4;
    }catch(ArrayIndexOutOfBoundsException e){System.out.println(e);}

    System.out.println("other statement);
  }catch(Exception e){System.out.println("handeled");}

  System.out.println("normal flow..");
 }
}
```

Bài 2: Sử dụng finally

a) Sử dụng finally khi ngoại lệ không xẩy ra

```
public class TestFinallyBlock {
    public static void main(String args[]) {
```

```
        try {
            int data = 25 / 5;
            System.out.println(data);
        } catch (NullPointerException e) {
            System.out.println(e);
        } finally {
            System.out.println("finally block is always executed");
        }
        System.out.println("rest of the code...");
    }
}
```

b) Sử dụng finally khi ngoại lệ xẩy ra nhưng không xử lý

```
public class TestFinallyBlock1 {
    public static void main(String args[]) {
        try {
            int data = 25 / 0;
            System.out.println(data);
        } catch (NullPointerException e) {
            System.out.println(e);
        } finally {
            System.out.println("finally block is always executed");
        }
        System.out.println("rest of the code...");
    }
}
```

c) Sử dụng khối lệnh finally nơi ngoại lệ xảy ra và được xử lý.

```
public class TestFinallyBlock2 {
    public static void main(String args[]) {
        try {
            int data = 25 / 0;
            System.out.println(data);
        } catch (ArithmeticException e) {
            System.out.println(e);
        } finally {
            System.out.println("finally block is always executed");
        }
        System.out.println("rest of the code...");
    }
}
```

d) Sử dụng khối lệnh finally trong trường hợp trong khối try có lệnh return.

```
public class TestFinallyBlock3 {
    public static void main(String args[]) {
        try {
```

```
            int data = 25;
            if (data % 2 != 0) {
             System.out.println(data + " is odd number");
             return;
            }
        } catch (ArithmeticException e) {
            System.out.println(e);
        } finally {
            System.out.println("finally block is always executed");
        }
        System.out.println("rest of the code...");
    }
}
```

Bài 3: Sử dụng từ khóa throw

a) Ném ra ngoại lệ nhưng không xử lý

```
public class TestThrow1 {
    static void validate(int age) {
        if (age < 18)
            throw new ArithmeticException("not valid");
        else
            System.out.println("welcome");
    }

    public static void main(String args[]) {
        validate(13);
        System.out.println("rest of the code...");
    }
}
```

b) Ném ra ngoại lệ nhưng có xử lý

```
public class TestThrow2 {
    static void validate(int age) {
        try {
            if (age < 18)
                throw new ArithmeticException("not valid");
            else
                System.out.println("welcome");
        } catch (ArithmeticException ex) {
            System.out.println(ex.getMessage());
        }
    }

    public static void main(String args[]) {
        validate(13);
        System.out.println("rest of the code...");
    }
```

```
}
```

Bài 4: Sử dụng từ khóa throws

a) Ví dụ phương thức dùng throws ném trả về ngoại lệ checked

```
import java.io.IOException;

public class TestThrows1 {
   void m() throws IOException {
      throw new IOException("Loi thiet bi");// checked exception
   }

   void n() throws IOException {
      m();
   }

   void p() {
      try {
         n();
      } catch (Exception e) {
         System.out.println("ngoai le duoc xu ly");
      }
   }

   public static void main(String args[]) {
      TestThrows1 obj = new TestThrows1();
      obj.p();
      System.out.println("luong binh thuong...");
   }
}
```

----------Hết----------