Welcome to the

Java Course

Module 3 – Day 05

Content of the course

- Object-Oriented Programming concepts
- Classes and objects
- Inheritance and polymorphism
- Encapsulation and accessibility
- Interface and abstract classes
- **Exceptions**

Air Flight Company Project - Step 4

- Update the Aircraft class to be abstract.

- Create an interface called Bookable. It will specify that certain class/es can be booked. It should be possible to know how many current bookings there are and how many are left. The interface should also provide a method to make a new booking.

- Which existing class should be updated to implement the Bookable interface?

Air Flight Company Project - Step 4

- Add a new feature to the project. The new class should also implement the Bookable interface.
    - Maybe passengers can book first class seats?
-
    - Maybe passengers can book special types of foods?
-
    - Maybe each flight needs to book a gate at the airport?

Exceptions

- Exceptions are events that disrupt the normal flow of a program's instructions.
- Exception handling is used to prevent the program from terminating or other erroneous situations from occurring.
- Exceptions are objects that represent an error or an unexpected event occurring during the execution of a program.

Exceptions

```
public void division( ) {
  Scanner scanner = new Scanner(System.in);
  double x = scanner.nextDouble();
  double y = scanner.nextDouble();
  System.out.println( "The result is " + (x / y) );
}
```

What if those aren’t numbers ?

Catching exceptions

```
try {
  // code we want to execute
} catch ( exception_case e ) {
  // code in case of exception
}
```

Catching exceptions

```
public void division( ) {
  Scanner scanner = new Scanner(System.in);

  try {
    double x = scanner.nextDouble();
    double y = scanner.nextDouble();
    System.out.println( "The result is " + (x / y) );
  } catch ( Exception e ){
    System.out.println("You must enter a number");
  }
}
```

Now YOUR TURN !

Let’s do exercise 1

Throwing exceptions

```
public class Person {
    private String name;
    private int age;

    public void setAge(int age) {
        this.age = age;
    }
}
```

What if the age is a negative number?

```
public static void main(String[] args) {
    Person person = new Person();
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter person age: ");
    age = scanner.nextInt();
    person.setAge(age);
}
```

Throwing exceptions

```
public class Person {
    private String name;
    private int age;

    public void setAge(int age) throws Exception {
        if (age <= 0) {
            throw new Exception("Age cannot be negative");
        }
        this.age = age;
    }
}
```

We should throw an exception to notify that something went wrong

Now YOUR TURN !

Let’s do exercise 2

Air Flight Company Project - Step 5

- Add exception handling to the project. Some examples could be:
    - What happens if the user enters an empty Flight number when creating a new flight?
    - What happens if he enters an empty destination?