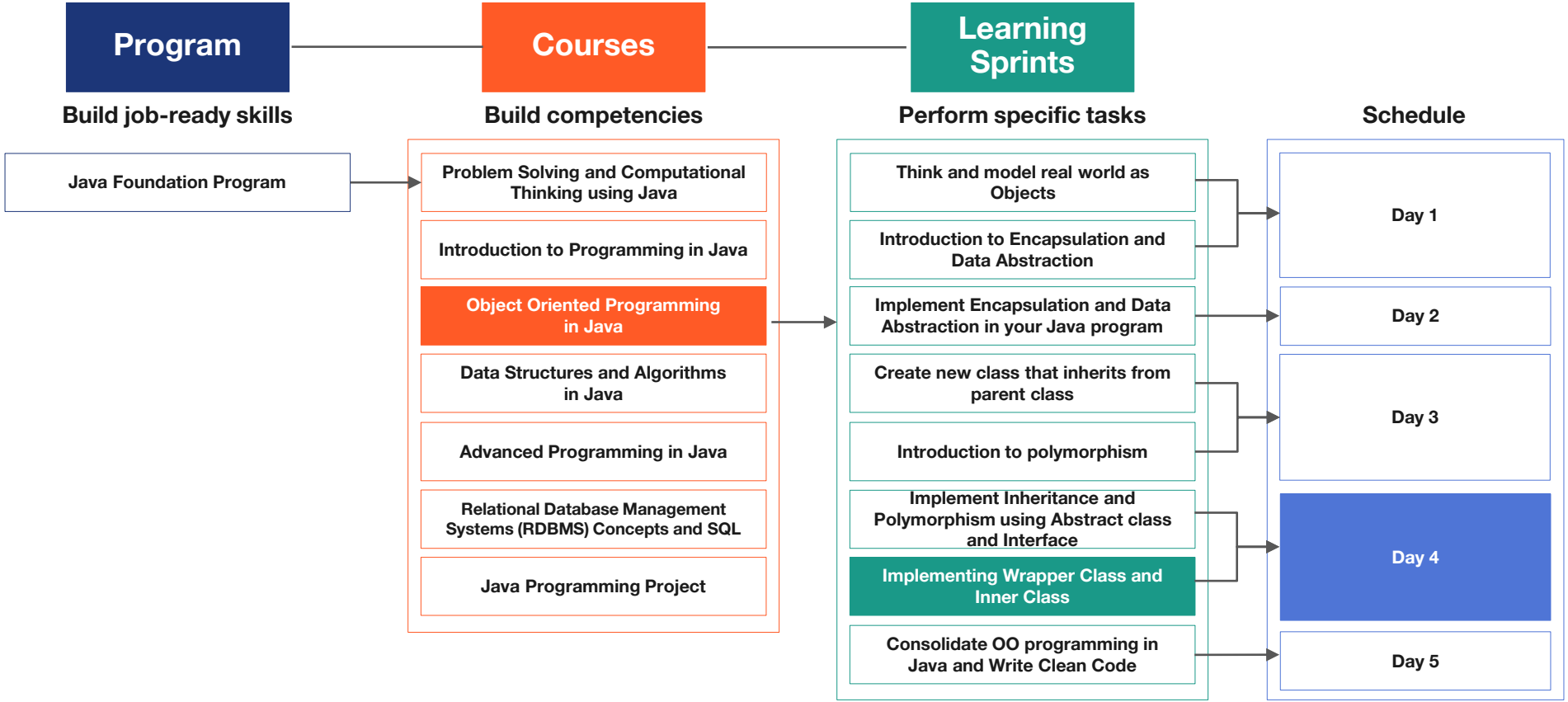
Java Program: Course 3: Plan



Think and Tell

How can we create a method in Java that accepts the details of a player (name, age, and score) as primitive data types and stores each of these as an object?

Implement Wrapper Classes and Inner Classes

Learning Objectives

- Explain wrapper class
- Implement boxing and unboxing
- Use nested classes
- List the types of inner classes

Wrapper Class

- Wrapper classes make the primitive data types to act as objects because most of the time we need to use the primitive types as objects for a program
- A wrapper class wraps the primitive data type into an object

Primitive data types	Wrapper classes
char	Character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean

Wrapper Class (contd.)

- Primitive type

```
int ageOfEmployee = 35;
```

- Wrapper class

```
Integer salaryOfEmployee = new Integer(35000);
```

Wrapper Class (contd.)

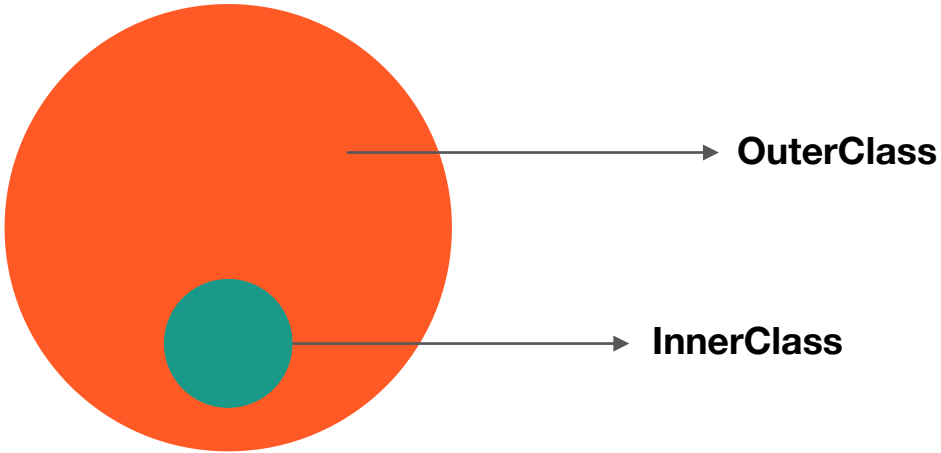
- Wrapping is done by the compiler
- If a primitive is used where an object is expected, then the compiler, in its wrapper class, boxes the primitive
- Similarly, if a number object is used where a primitive is expected, then the object is unboxed by the compiler

Example of Boxing and Unboxing

```
public class Main {
        public static void main(String[] args)
        {
                int num1 = 10;
                Integer numObj1 = new Integer(20);
                Integer numObj2 = new Integer(num1); //Boxing
                Integer numObj3 = num1; //Autoboxing

                System.out.println("numObj1= " + numObj1 + "\n" +
"numObj2= " + numObj2 + "\n" + "numObj3= " + numObj3 + "\n");

                Integer numObj4 = new Integer(36);
                int num2 = numObj4.intValue(); //Unboxing
                int num3 = numObj4; //AutoUnboxing

        System.out.println("num2= " + num2 + "\n" + "num3= " + num3);
    }
}
```

Inner Class

- In Java, a class can also contain another class. Such a class is known as an inner class or a nested class
- The class that contains the inner class is known as an outer class



Inner Class (contd.)

- An inner class has the following advantages:
  - It improves encapsulation, as a class is hidden inside another class
  - It provides better readability
  - It is useful for grouping classes logically
- The following code snippet demonstrates an inner class:

```
class outerclass
{
        class innerclass
        {
                ...
        }
}
```

Types of Inner Classes

Example of an Inner Class

```
public class Outer {
        class Inner
        {
                public void show()
                {
                        System.out.println("In a nested class method");
                }
        }
}
public class Main
{
        public static void main(String[] args)
        {
                Outer.Inner in = new Outer().new Inner();
                in.show();
        }
}
```

Example of an Inner Class (contd.)

```
public class Outer
{
        private static void outerMethod()
        {
                System.out.println("inside outerMethod");
        }
        static class Inner
        {
                public static void main(String[] args)
                {
                        System.out.println("inside inner class Method");
                        outerMethod();
                }
        }
}
```

Example of an Inner Class (contd.)

```
public class Outer {
        void outerMethod() {
                System.out.println("inside outerMethod");
                class Inner {
                        void innerMethod() {
                        System.out.println("inside innerMethod");
                }
        }
                Inner y = new Inner();
                y.innerMethod();
    }}
public class Main {
        public static void main(String[] args) {
                Outer x = new Outer();
                x.outerMethod();
        }
}
```

Example of an Anonymous Inner Class

```
abstract class Age{
   int age =22;
   abstract void printAge();
}
class ConcreteAge extends Age{
   void printAge(){
      System.out.println(age);
}
}

public class AgeMain {
   public static void main(String args[]) {
        Age age = new ConcreteAge();
         age.printAge();
 }
}
```

Example of an Anonymous Inner Class (contd.)

```
abstract class Age{
   int age =22;
   abstract void printAge();
}
public class AgeMain {
public static void main(String[] args) {
//        Age age = new ConcreteAge();
//        age.printAge();
       Age age= new Age() {
           public void printAge() {   System.out.println(age);
           }};
       age.printAge();}}
```

Example of an Anonymous Inner Class (contd.)

```
public interface Fruit{
    void printTaste();
}
public class FruitMain {
    public static void main(String args[]) {
        Fruit grapes= new Fruit() {
            public void printTaste() {
                System.out.println("Sweet");
            }
        };
        grapes.printTaste();
    }
```

Interactive Demo

Payment for products bought online is commonly done through credit cards. Validation of the credit card details entered by the users is an important step of the payment process. It is done for each credit card.

Write a program in Java to validate the credit card details entered by a user.

Key Takeaways

- Wrapper classes
- Primitive data types and wrapper classes
- Boxing and unboxing
- Benefits of inner classes
- Implementation of inner classes