# Introduction to Java

LESSON 3

## Learning Outcomes: Lesson 3

- ArrayList
- Overloading Constructors
- Object Oriented Programming
    - Encapsulation
    - Association

## Collections

- A Collection is a container that groups similar elements into an entity.
- Examples would include a list of bank accounts, set of students, group of telephone numbers.
- Collections used to store groups of related objects in memory
- Each type of a collection is defined in a class
- These classes provide efficient methods to organize, store and retrieve the data.
- One type of collection is ArrayList

## ArrayList

- An ArrayList is like an array of objects
- Difference between arrays and arrayList
  - Arrays have special syntax; ArrayLists don’t.
  - Array is a fixed size, but an ArrayList expands as you add to it. This means you don’t need to know the size beforehand.
- Arrays can hold primitive or objects but ArrayList can hold only objects.
  - Autoboxing make it appear that ArrayList can hold primitive.

## Using the ArrayList

- The arrayList is declared and created in the same way as an object of any class, except that you need to specify the base type as follows:

```
private ArrayList <Student> students = new ArrayList<Student>();
```

- In order to make use of the ArrayList class, it must first be imported

```
Import java.utill.ArrayList
```

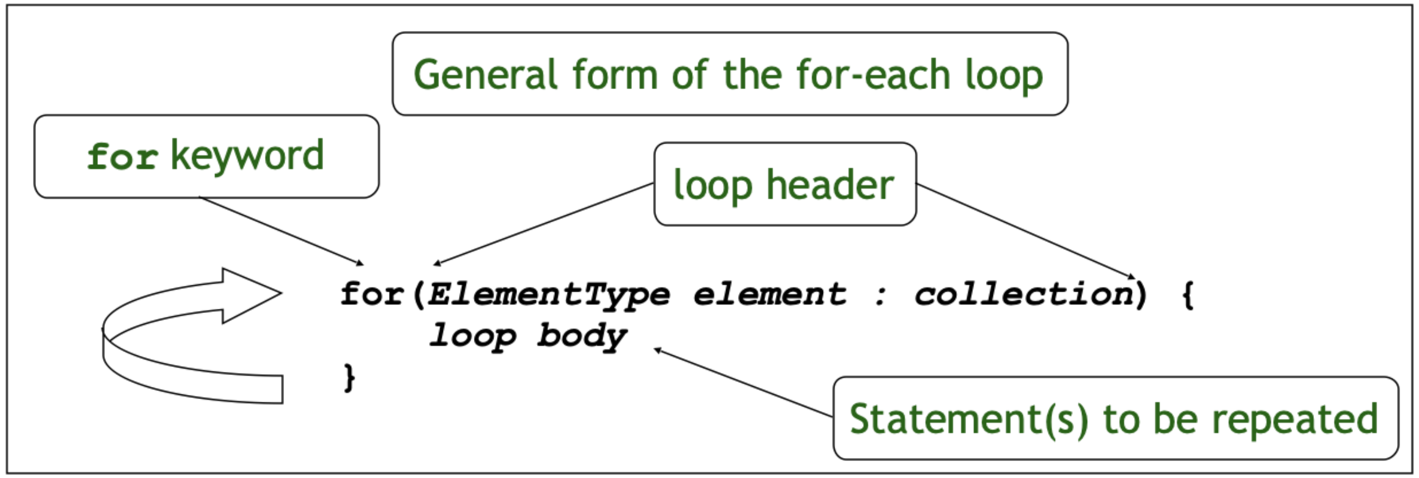
## Using the ArrayList

- Each element in the ArrayList is accessible by a  0-based index.
- ArrayList class provide a selection of powerful methods that can do many things.
- Some of the methods that we will work with is
  - add(Element e)
  - remove (int index)
  - size()
  - get(int index)

## Foreach Loop

- We often want to repeat some action over and over.
- With collections, we often want to repeat things once for every element in the collection.
- Starting with version 5.0, Java has added a new kind of loop called a *for-each* loop or *enhanced for* loop.
- This kind if loop has been designed to cycle through all the elements in a collection like the ArrayList.

# Foreach Loop Pseudo Code


General form of the for-each loop
for keyword
loop header
Statement(s) to be repeated

```
for(ElementType element : collection) {
    loop body
}
```

Pseudo-code expression of the actions of a for-each loop

For each *element* in *collection*, do the things in the *loop body*.

## Autoboxing

- Autoboxing is the automatic conversion that the Java compiler makes between the primitive types and their corresponding object wrapper classes.
  - ArrayList<Integer> intList = new ArrayList<Integer>();
  - ArrayList<Double> doubleList = new ArrayList<Double>();
  - ArrayList<Character> charList = new ArrayList<Double>();

## Overloading Constructors

- Both constructor and methods can be overloaded in the class
- Overloading allows a class to have two or more methods or constructors having same name if  their signature are different
  - Different number of parameters
  - Different parameters type
  - Different order of parameters

## Overloading Constructors Example

- public Book ()
- public Book (String title)
- public Book (int numPages)
- public Book (String title, int numPages)
- public Book (int numPages, String title)

## Object Oriented Programing

- Object-oriented programming is about creating objects that contain both data and methods.
- Object-oriented programming has several advantages:
  - OOP is faster and easier to execute
  - OOP provides a clear structure for the programs
  - OOP helps to keep the Java code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
  - OOP makes it possible to create full reusable applications with less code and shorter development time

## Encapsulation

- Encapsulation is one of the fundamental concepts in object oriented programming (OOP). It describes the idea of bundling data and methods that work on that data within one unit, e.g., a class in Java.
- If you have an attribute that is not visible from the outside of an object, and bundle it with methods that provide read or write access to it, then you can hide specific information and control access to the internal state of the object.
- Encapsulation is such a basic concept that most Java developers use it without thinking about it. It’s simply how you design a Java class. You bundle a set of attributes that store the current state of the object with a set of methods using these attributes.

## Association

Association is a general binary relationship that describes an activity between two classes. It represents **Has-A** relationship.

- Association relationship indicates how objects know each other and how they are using each other’s functionality. It can be one-to-one, one-to-many, many-to-one and many-to-many.
  - One to one : one family can live in one house and one house can contain one family.
  - One to many : one department can have many employees.
  - Many to many : customers can purchase many products and many products can be purchased by many customers