Array of Object

Teaching Team

Algorithm and Data Structure

2024/2025

Topics

- Review of Array
- Array of Object
- Declaring Array of Object
- Accessing Element of Array of Object
- Instantiating Object in Array of Object
- Accessing Atribut/Method of Element of Array of Object
- Using Loop to Manage Array of Object
- Error
- Using Constructor
- Instantiating Object using different constructor

Learning Outcome

- After finishing this materials, students must be able to:
  - Understand the concept of applying arrays to manage multiple objects (array of objects).
  - Understand the declaration of an array of objects.
  - Understand how to instantiate objects within each array element.
  - Understand how to access attributes as well as methods of object within an array element.
  - Understand the types of error that often occur when applying an array of objects.
  - Understand the use of looping to access array element.

Array (a review)

- Array is a complex variabel (can save more than 1 value) with the same data type
- Array is identified by [ ]
- Declaring Array:

```
int data[] = new int[3];
```

- Initializing Array:

```
int data[] = {30,3,2};
```

- Accessing array → using **index of array**

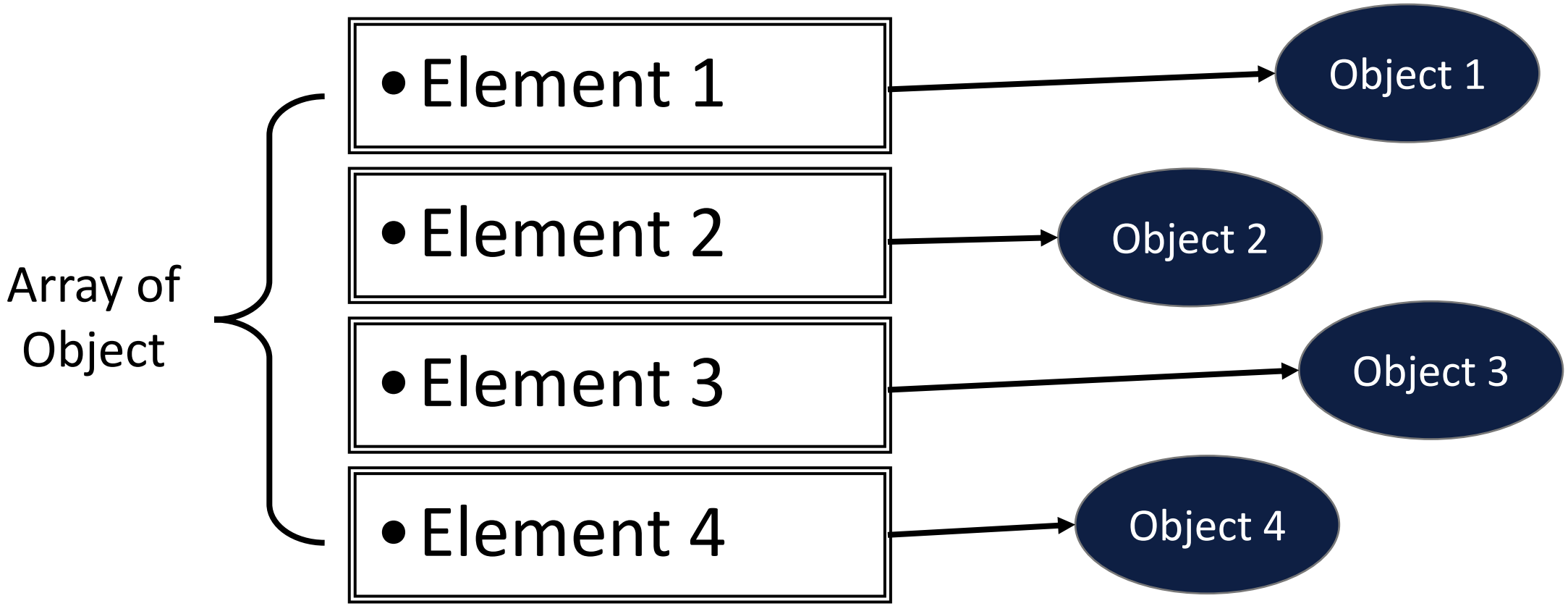
```
data[1] = 340;
int result = data[0]+data[1]+data[2];
System.out.println(data[2]);
```

Array of Object

- In the previous slide we have reviewed array of primitive data type
- Array also could be made from object of a class, instead of primitive data type (int, float, double, long, short, boolean, etc)
- In the primitive data type array the value is stored in each element of array. While in the array of object the value stored in each element is the **object** of a class.
- And basicly, how to manage array of object is the same with primitive array. The only difference is just the type of value stored inside the array. If Array object will store **object of a class** in the element of array

Array of Object

If there is an array with 4 object elements, the illustration is as follows:



Declaring Array of Object (1)

- If the array with primitive data type declared by using primitive data type, then array of object declared by using the **name of class** from which the object created as the type
- Format

```
ClassName arrayName[] = new ClassName[length];
```

Declaring Array of Object (2)

- Example: class Rectangle

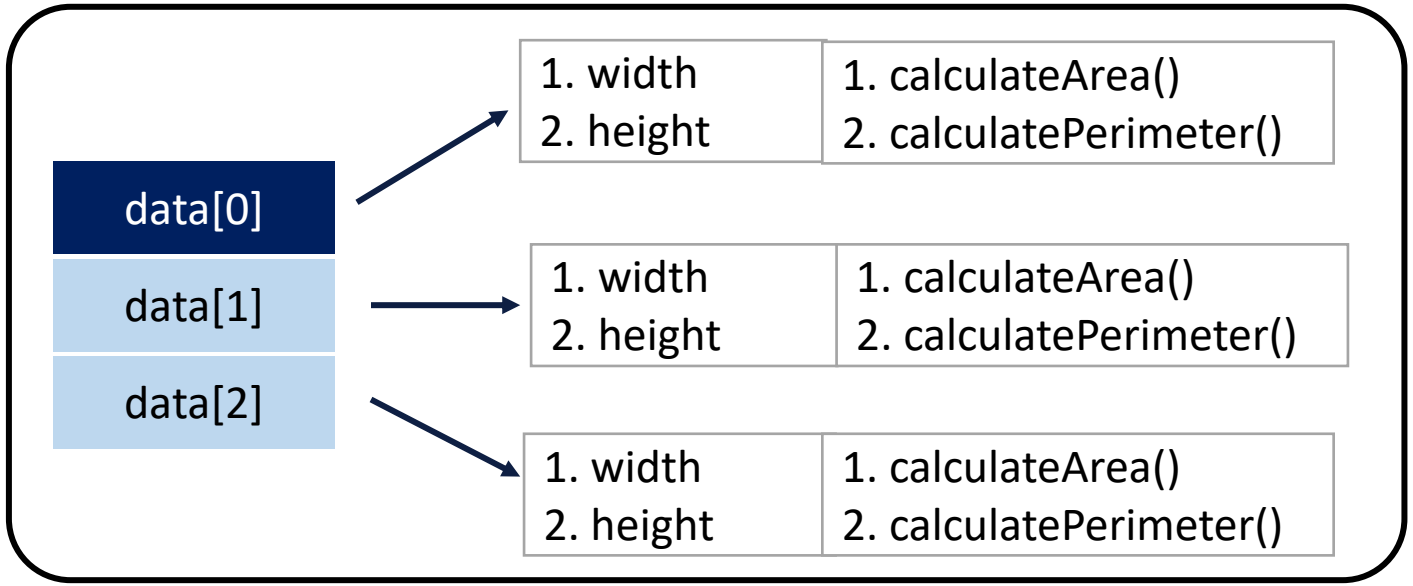
```
public class Rectangle {
    int width;
    int height;

    int calculateArea(){
        return width*height;
    }
    int calculatePerimeter(){
        return 2*(width+height);
    }

}
```

- Array of Object Declaration could be:

```
Rectangle[] data = new Rectangle[3];
```

- In the code above, we create an array **data** that could be used to store **3** objects from class **Rectangle.**

Declaring Array of Object (3)



Each element of the array has **its own object** (complete with the attributes and methods)

Accessing Element of Array of Object

- To access the element of array of object we need to use the index of array of the element we need to access.
- Since in array of object, every single array element handles an object, then don't forget to access attribute and method from the object
- Example:

```
data[0].width = 100;
data[1] .height = 80;
data[0].calculateArea();
```

Instantiating Object in the Element of Array of Object

- Because Array of Object will store object in each element, then we need to create(instantiate) object at in each element of array.
- Example:

```
Rectangle[] data = new Rectangle[3];
data[0] = new Rectangle();
data[1] = new Rectangle();
data[2] = new Rectangle();
```

Accessing Attribute/Method of Element of Array of Object

- Each element of array of object will store object. And object has attribute and method. So that we could access them.
- The format to access the element of array by using index, followed by dot (.) and attribute name or method name of the object. Example:

Accessing attribute

Accessing method

```
Rectangle[] data = new Rectangle[3];
data[0] = new Rectangle();
data[0].height = 10;
data[0].width = 5;
System.out.println("Area = "+data[0].calculateArea());
System.out.println("Perimeter = "+data[0].calculatePerimeter());
```

Using Loop to Manage Array of Object

- We can iterate each element of array through a loop

```
Rectangle[] data = new Rectangle[3];
for(int i=0;i<data.length;i++){
    data[i] = new Rectangle();
    data[i].height = 15*i;
    data[i].width = 4*i;
    System.out.println("Area = "+data[i].calculateArea());
}
```

Error

- ArrayIndexOutOfBoundsException : caused by accessing array element at out-of-range index
- Example :

```
Rectangle[] data = new Rectangle[3];
data[3] = new Rectangle();
```

- Array **data** has length 3, so the last index is 2, so if we access index 3, it will give us ArrayIndexOutOfBoundException.

Error

- NullPointerException : caused by accessing object that has not been instantiated yet. The object is still not created though
- Example

```
Rectangle[] data = new Rectangle[3];
data[0].height = 10;
```

- Index =0 of array **data** still does not have object (the object is not created since there is no instantiation process there). So the error will rise if we access its height, because the object is still not created

Using Constructor

- Constructor is special method that runs only at the instantiation process
- By default, each class has default constructor
- We can add parametric constructor as well

```
public class Rectangle {
    int width;
    int height;

    public Rectangle() {
    }

    public Rectangle(int w, int h) {
        width = w;
        height = h;
    }

    int calculateArea(){
        return width*height;
    }
    int calculatePerimeter(){
        return 2*(width+height);
    }

}
```

Object Instantiation using Different Constructor

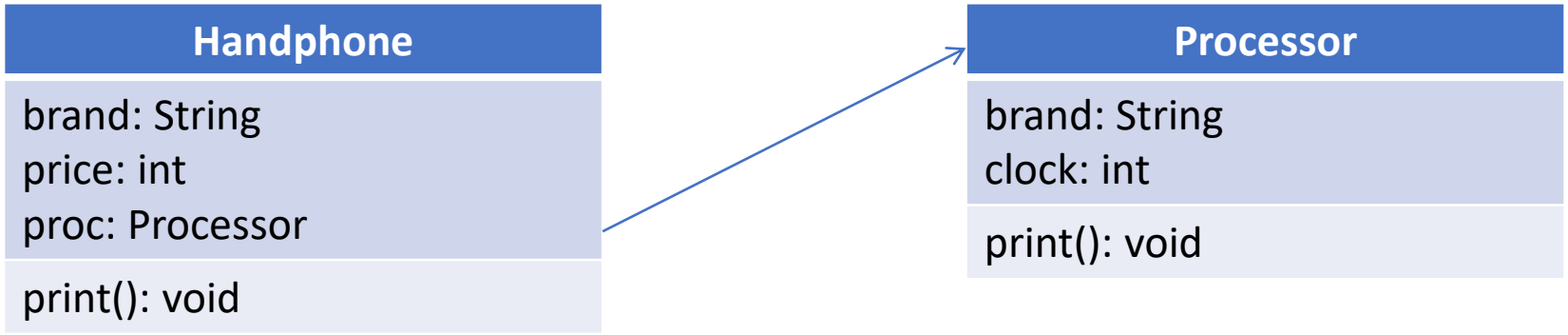
Instantiation by default constructor

Instantiation by parametric constructor

```
Rectangle[] data = new Rectangle[3];
data[0] = new Rectangle();
data[0].height = 10;
data[0].width = 5;
System.out.println("Area="+data[0].calculateArea());

data[1] = new Rectangle(20, 3);
System.out.println("Area="+data[1].calculateArea());
```

Object with Object Attribute

- A class can have attribute that is object of other class.
- For example, class **Handphone**. It has attribute **brand** and **price;** it also has attribute **proc** which is object of another class **Processor**



Object with Object Attribute

```
public class Handphone {
    String brand;
    int price;
    Processor proc;

    void print(){
        System.out.println("Phone brand = "+brand);
        System.out.println("Phone price = "+price);
        proc.print();
    }
}

class Processor{
    String brand;
    int clock;

    void print(){
        System.out.println("Processor brand = "+brand);
        System.out.println("Processor clock = "+clock);
    }
}
```

Object with Object Attribute

```
public class HandphoneMain {
    public static void main(String[] args) {
        Processor p = new Processor();
        p.brand = "intel";
        p.clock = 128;

        Handphone hp = new Handphone();
        hp.brand = "Samsung";
        hp.price = 1000;
        hp.proc = p;
        hp.print();
    }

}
```

Object **p** assigned to attibute **proc** of object **hp**.

Assignment

1. Give 3 examples of the implementation of array of objects in acertain system/application and specify the attributes and methods of each object!
2. Create class diagram for your no.1 answer! (3 class diagram)
3. Create class diagram for class ”Course” if there are:
   - *Attributes*: Course ID, Course Name, Lecturer Name, Class Quota, and List of Students
   - *Methods*: Edit Course ID, Edit Course Name, Set Lecturer, Add Class Quota, Reduce Class Quota, and Add Student


thank you
danke
謝謝
ngiyabonga
شكراً جزيلاً
рахмат
teşekkür ederim
Баярлалаа
спасибо
gracias
dank je
tapadh leat
faafetai lava
kiitos
dankie
vinaka
спасибі
blagodaram
misaotra
matondo
paldies
grazzi
mahalo
хвала
asante
manana
obrigada
tenki
murakoze
chokrane
mauruuru
hvala
bedankt
dziękuję
enkosi
köszönöm
akun
dankon
aciū
djiere dieuf
tau
mochchakkeram
mamnun
дякую
go raibh maith agat
chnorakaloutioun
gralias ago
grācies
sulpáy
sukriya
kop khun krap
taiku
obrigado
sobodi
dēkuji
sagolun
mësi
didi madloba
najis tuke
terima kasih
rahmat
kam sah hamnida
তোমাকে ধন্যবাদ
ありがとう
tanemirt
rahmet
grazie
arigatô
takk
dakujem
trugarez
shukriya
merce
мерси
diolch
dhanyavadagalu
ευχαριστώ
감사합니다
xièxie
merci