

## Lab 5 – Introduction to Objects

### Aim

This week is focused on objects, and being comfortable with using them. This will require you to be comfortable with using methods, and with using arrays. If you have not finished Lab 4, it is important that you finish this first.

### Tips:

1. Objects are a very important Java concept, so it is important you understand what you are doing. If unsure, ask.
2. If you do not have week 4's checkpoint, ask at the start of the lab.

### Person Classes

At the end of week 4, we created a class with several arrays to hold information about people. Lecture 5 introduced the concept of using objects. Rather than have static classes and multiple arrays we can create Person objects.

- Finish your arrays project from last week before starting this one.
- Create an Empty NetBeans project called Lab5-Person, with a main class. Do not make any changes to this main class yet.
- Create a new Java class. This is going to be our data class, so call it "Person". You should now have 2 Java classes, your main class, and a new Person class.
- Note that Person should have no Main class, why not?
- The Lecture notes discuss creating a cake object, and this week, you should create a Person. Use the same attributes as you used in your arrays last week or create your own. Do not give them initial values, just **declare** their name and type. E.g.

```
String name;           //name
int age;               //age of person
String address;       // a city: Suzhou or London
int id;               // an ID code
```

- You will also need to create a constructor. Again, consult your lecture notes for guidance. A constructor requires parameters, which are used to initialise your variables.
- Add a System.out.println in the constructor, so you will know when a new Person has been created.

## Encapsulation

As discussed in your lecture, variables and methods should be private unless they need to be accessed by other classes or objects. We will discuss this more in week 6.

- Your instance variables (the variables in the Person class) should be set to private
- However, doing this means that we will not be able to access our data! How can we solve this?
- We can add “getters” and “setters” for each instance variable
- Create methods like this for each instance variable. This will let you access the variables:

```
public String getName(){
    return name;
}

public void setName(String newName){
    name = newName;
}
```

## Creating a Person Object

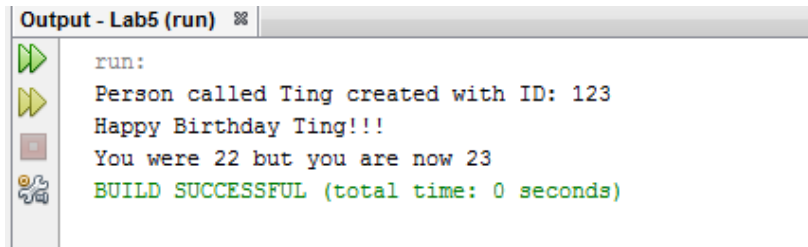
You now have a person class, which means that you are ready to use it to create people!

- Open your main (controller) class. You can now create an object, and instantiate it. Consult your lecture notes and ask a TA if you are not sure how to do this.
- You need to create a variable of **type** Person, and you need to create a **new** Person, call the constructor in Person, and **pass in** suitable parameters.
- `Person person1 = new Person("Joe", 87, "London", 123);`
- For example, when a Person has been created, the console output from the constructor should look like:  
`Person called Ting created.`
- This is how we create an **object**.
- Try to create 3 people who have their own unique attributes, call them *person1*, *person2*, *person3*.

## Adding methods

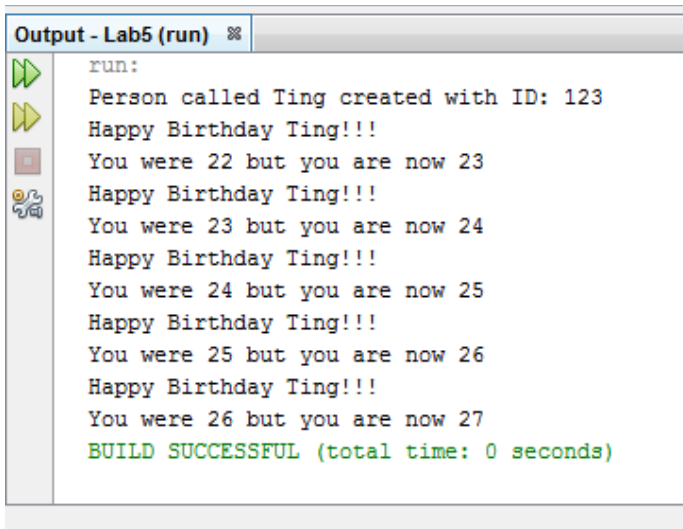
Your person class now has attributes and a constructor, and getter and setter methods

- Add a method for a birthday! Name it birthday. When this method is called, you want to display a “Happy Birthday!!” message and add one to the age. You should display a helpful message when the method is called



```
run:
Person called Ting created with ID: 123
Happy Birthday Ting!!!
You were 22 but you are now 23
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Call it several times to make sure it works!



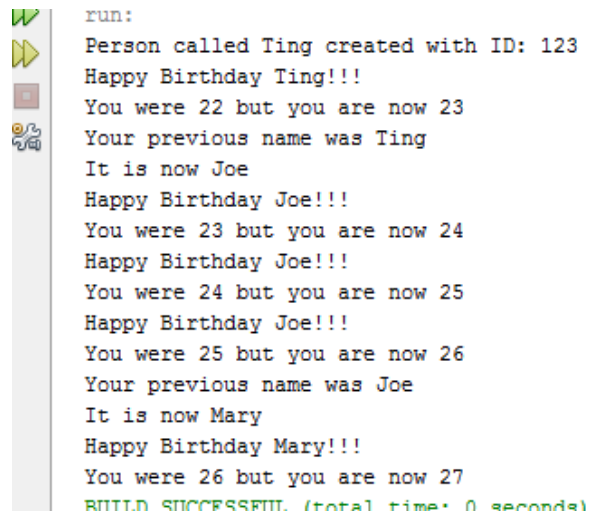
```
run:
Person called Ting created with ID: 123
Happy Birthday Ting!!!
You were 22 but you are now 23
Happy Birthday Ting!!!
You were 23 but you are now 24
Happy Birthday Ting!!!
You were 24 but you are now 25
Happy Birthday Ting!!!
You were 25 but you are now 26
Happy Birthday Ting!!!
You were 26 but you are now 27
BUILD SUCCESSFUL (total time: 0 seconds)
```

## Using your Getters

You can also use your getters and setters to change information

- You should have created a `getName()` method in your person class, use it to return the name of a person and display it in your main method. Here, we do not change the getter. We will call the `system.out.` in the main method:  
  

```
System.out.println("your previous name was " + person1.getName());
```
- You should also have a `setName(String n)` method in your person class. Use it to change the name of the person
- You can then use the get name method again to display the new name
- Now, using your birthday and change name methods, you should be able to give your people a more interesting life!

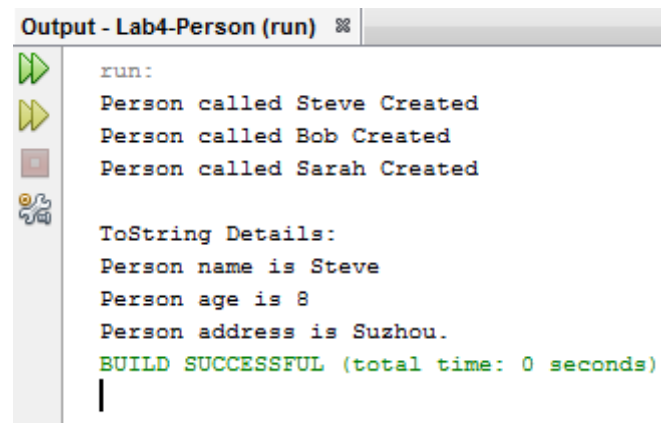


```
run:
Person called Ting created with ID: 123
Happy Birthday Ting!!!
You were 22 but you are now 23
Your previous name was Ting
It is now Joe
Happy Birthday Joe!!!
You were 23 but you are now 24
Happy Birthday Joe!!!
You were 24 but you are now 25
Happy Birthday Joe!!!
You were 25 but you are now 26
Your previous name was Joe
It is now Mary
Happy Birthday Mary!!!
You were 26 but you are now 27
BUILD SUCCESSFUL (total time: 0 seconds)
```

### Using a toString Method

A `toString` method takes whatever object that you have created and tries to convert it to a string. This will be discussed in week 6.

- Try it out with your `person1` object. Use the code:  
`System.out.println(person1.toString());`
- You should see something like :  
`lab5.person.Person@15db9742`
- Why? Because your `Person` object doesn't handle `toString()` correctly. Try to fix this. Try to create a `toString()` method that receives no parameters and returns a `String`. Your output should look like:



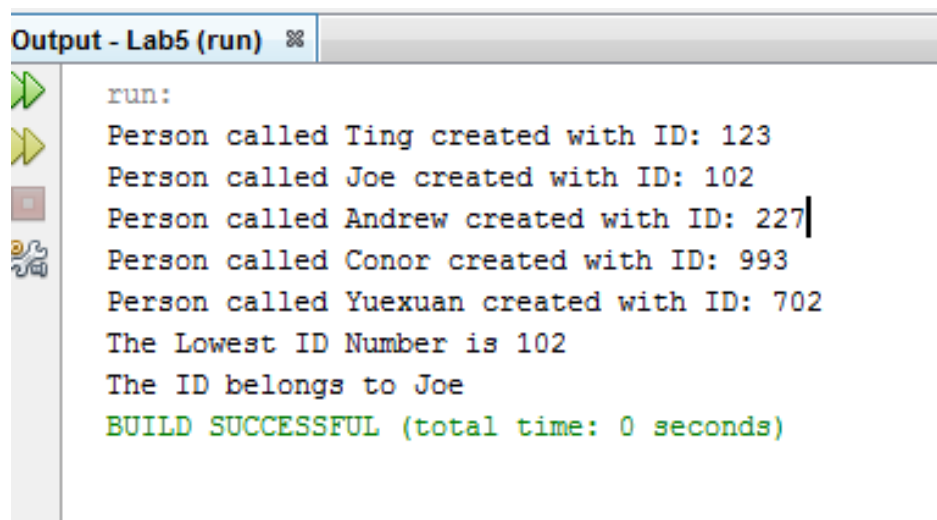
```
Output - Lab4-Person (run) %
run:
Person called Steve Created
Person called Bob Created
Person called Sarah Created

ToString Details:
Person name is Steve
Person age is 8
Person address is Suzhou.
BUILD SUCCESSFUL (total time: 0 seconds)
|
```

## Person Arrays

You worked with arrays last week, but an array can be of any type. For example you can create an array of Person objects!

- Create an array of 5 Person objects, and then use a For loop to view the toString method of each person. Ask if you don't understand!
- Use the setAge() setter method to change everyone's age!
- In week 4, you discussed how to search arrays for useful information. We want to find the person that has the lowest id number. There are several steps involved in this.
- Firstly, using an array of objects, and your getID() method, create a method in your Lab5 class that will search through the array, and identify the lowest student number. See some hints below!



```
Output - Lab5 (run) ✖
run:
Person called Ting created with ID: 123
Person called Joe created with ID: 102
Person called Andrew created with ID: 227
Person called Conor created with ID: 993
Person called Yuexuan created with ID: 702
The Lowest ID Number is 102
The ID belongs to Joe
BUILD SUCCESSFUL (total time: 0 seconds)
```

- HINT #1, in your main class, you may wish to create a Person array as a static class variable
- HINT #2, build this program slowly. Start by using your getID() method. Test it. Then try creating an array of people, and test it. Then try to identify the lowest student ID.
- HINT #3, look at the lecture notes from week 4 to try to search an array!
- HINT #4, to display a useful message at the end, can you use a method that will get a name?

## Next Steps – Homework 5

When you have completed this sheet, there is an excellent challenge on homework sheet 5! This will work with nested for loops. Go and try it!