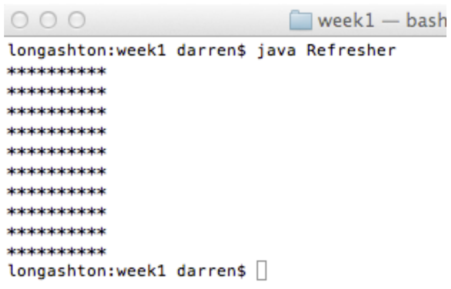
This lab is a short programming refresher. During this lab we will complete a number of small programming tasks in Java. Some of these are similar to the type of questions employers may ask you to solve as part of technical interviews.

Exercise 1a,
Create the classic "Hello, World!" program.

Exercise 1b
Create a program that outputs the numbers between 1 and 200 but replaces any number divisible by three with the word "fizz", and any number divisible by five with the word "buzz".
Eg.1, 2, Fizz, 4, Buzz, Fizz, 7, 8, Fizz, Buzz, 11, Fizz, 13, 14, Fizz Buzz, 16,…

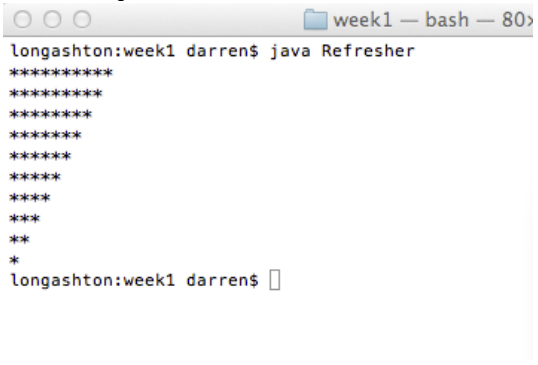
Exercise 2,
Create a java program that use two for nested loops to display a grid of "*" to the screen as shown in Fig 1.



```
longashton:week1 darren$ java Refresher
**********
**********
**********
**********
**********
**********
**********
**********
**********
**********
longashton:week1 darren$
```

Figure 1 - Output of Exercise 2

Exercise 3,
Create a java program that use two nested for loops to display a grid of "*" that decrements by as shown in Fig 2.

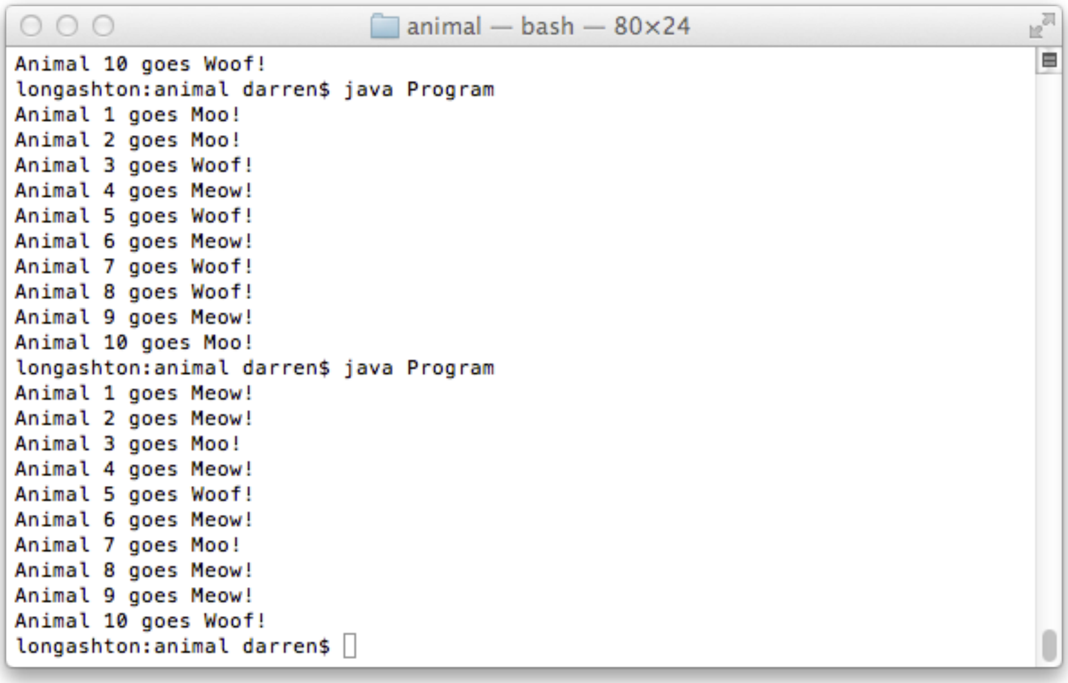


```
longashton:week1 darren$ java Refresher
**********
*********
********
*******
******
*****
****
***
**
*
longashton:week1 darren$
```

Figure 2 Output of Exercise 3

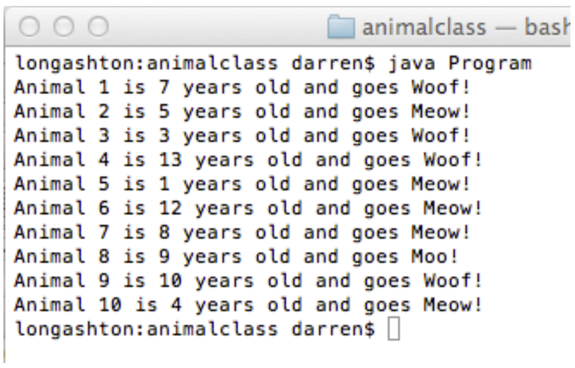
Exercise 4,
Create an animal interface that has a method speak that takes no parameters but returns a string. Create a dog, cat , cow classes that implement the animal interface with an appropriate speak method. Create a main program to exercise your animal class by creating an array of animas filled with random selection of dogs, cats and

cows. Then create a loop that loops through the array asking each animal to speak.



```
Animal 10 goes Woof!
longashton:animal darren$ java Program
Animal 1 goes Moo!
Animal 2 goes Moo!
Animal 3 goes Woof!
Animal 4 goes Meow!
Animal 5 goes Woof!
Animal 6 goes Meow!
Animal 7 goes Woof!
Animal 8 goes Woof!
Animal 9 goes Meow!
Animal 10 goes Moo!
longashton:animal darren$ java Program
Animal 1 goes Meow!
Animal 2 goes Meow!
Animal 3 goes Moo!
Animal 4 goes Meow!
Animal 5 goes Woof!
Animal 6 goes Meow!
Animal 7 goes Moo!
Animal 8 goes Meow!
Animal 9 goes Meow!
Animal 10 goes Woof!
longashton:animal darren$
```

Exercise 4b,
Re-implement the exercise 4 using a Class for Animal. Add an int to the Animal class called age that records the animal's age. Change your program to set the age of the animal when it is created to a random number between 1 and 14 with the output updated as in Fig 3.

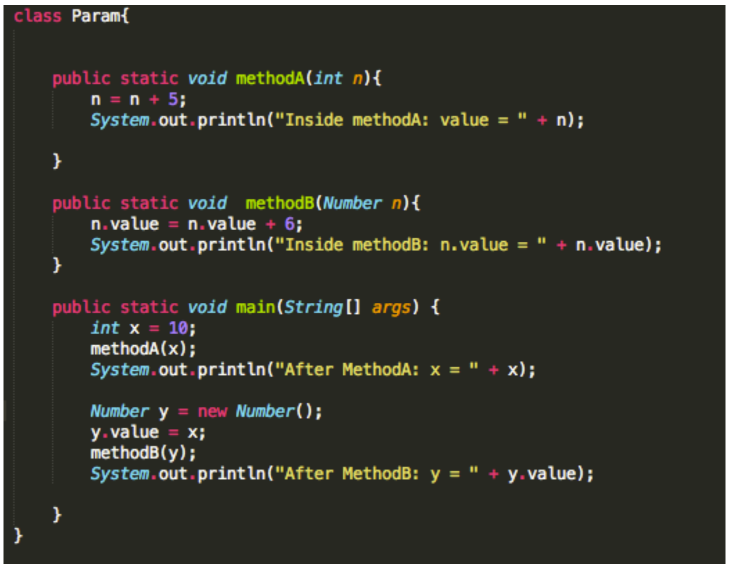


```
longashton:animalclass darren$ java Program
Animal 1 is 7 years old and goes Woof!
Animal 2 is 5 years old and goes Meow!
Animal 3 is 3 years old and goes Woof!
Animal 4 is 13 years old and goes Woof!
Animal 5 is 1 years old and goes Meow!
Animal 6 is 12 years old and goes Meow!
Animal 7 is 8 years old and goes Meow!
Animal 8 is 9 years old and goes Moo!
Animal 9 is 10 years old and goes Woof!
Animal 10 is 4 years old and goes Meow!
longashton:animalclass darren$
```

Figure 3 Output from Exercise 4b

Exercise 4c,
What is the difference between using classes and interface in java as used in 4a and 4b. When should you use a class when should you use and interface?

Exercise 5



```java
class Param{

    public static void methodA(int n){
        n = n + 5;
        System.out.println("Inside methodA: value = " + n);

    }

    public static void  methodB(Number n){
        n.value = n.value + 6;
        System.out.println("Inside methodB: n.value = " + n.value);
    }

    public static void main(String[] args) {
        int x = 10;
        methodA(x);
        System.out.println("After MethodA: x = " + x);

        Number y = new Number();
        y.value = x;
        methodB(y);
        System.out.println("After MethodB: y = " + y.value);

    }
}
```

Figure 4

Given the class Number which has a single int field named value explain why the program in Figure 4 outputs the following:

```
Inside methodA: value = 15
After MethodA: x = 10
Inside methodB: n.value = 16
After MethodB: y = 16
```