Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

Experiment No. 7
Implement a program using super and final keyword.
Date of Performance:
Date of Submission:

Aim: To implement the concept of super and final keyword.

Objective: To understand the usage of superclass and final method , variables and class

Theory:

super and **final** keywords are two popular and useful keywords in Java. They also play a significant role in dealing with Java programs and their classes. In this chapter, you will learn about how to use super and final within a Java program.

Syntax: `super.<method-name>();`

- Super variables refer to the variable of a variable of the parent class.
- Super() invokes the constructor of immediate parent class.
- Super refers to the method of the parent class

Instance refers an instance variable of the current class by default, but when you have to refer parent class instance variable, you have to use super keyword to distinguish between parent class (here employee) instance variable and current class (here, clerk) instance variable.

What is final in Java?

Final is a keyword in Java that is used to restrict the user and can be used in many respects. Final can be used with:

- Class
- Methods

- Variables

A method declared as final cannot be overridden; this means even when a child class can call the final method of parent class without any issues, but the overriding will not be possible.

Once a variable is assigned with the keyword final, it always contains the same exact value. Again things may happen like this; if a final variable holds a reference to an object then the state of the object can be altered if programmers perform certain operations on those objects, but the variable will always refer to the same object. A final variable that is not initialized at the time of declaration is known as a blank final variable. If you are declaring a final variable in a constructor, then you must initialize the blank final variable within the constructor of the class. Otherwise, the program might show a compilation error.

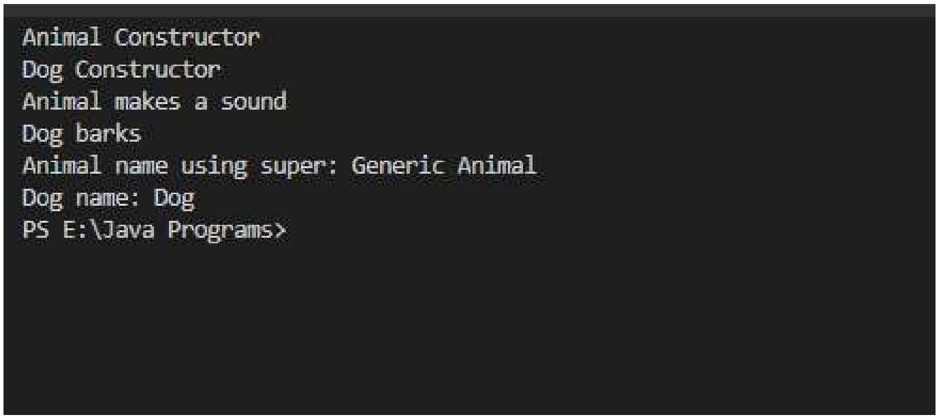
Code:

```
class Animal {
   String name = "Generic Animal";

   Animal() {
      System.out.println("Animal Constructor");
   }

   void sound() {
      System.out.println("Animal makes a sound");
   }
}

class Dog extends Animal {
   String name = "Dog";

   Dog() {
      super();
      System.out.println("Dog Constructor");
   }

   void sound() {
      super.sound();
      System.out.println("Dog barks");
      System.out.println("Animal name using super: " + super.name);
      System.out.println("Dog name: " + name);
   }
}

public class Main {
   public static void main(String[] args) {
      Dog dog = new Dog();
      dog.sound();
   }
}
```

Output:

```
Animal Constructor
Dog Constructor
Animal makes sound
Dog barks
Animal name using super: Generic Animal
Dog name: Dog
```

Screenshot:



```
Animal Constructor
Dog Constructor
Animal makes a sound
Dog barks
Animal name using super: Generic Animal
Dog name: Dog
PS E:\Java Programs>
```

Conclusion:

Comment on how you have used super and final keyword.

ANS: In this program, the `super` keyword is used to access the `sound()` method and `name` variable from the parent class `Animal` within the `Dog` class. It allows the `Dog` class to invoke the `sound()` method of `Animal` and reference the `name` variable of `Animal` while also having its own implementations. The `final` keyword is not used in the remaining code after the removal of the `Constants` class. Originally, it was intended to restrict the modification of the `MAX_SPEED` variable and prevent overriding the `displayMaxSpeed()` method, but these features are not present in the current code.