this keyword in java :
----
* As we know inside the method /constructor / block, method level variables are having more priority than class level variable.

* Whenever instance variable name and local/parameter variable name both are exactly same then inside the method /constructor / block body always method level variables are having more priority.

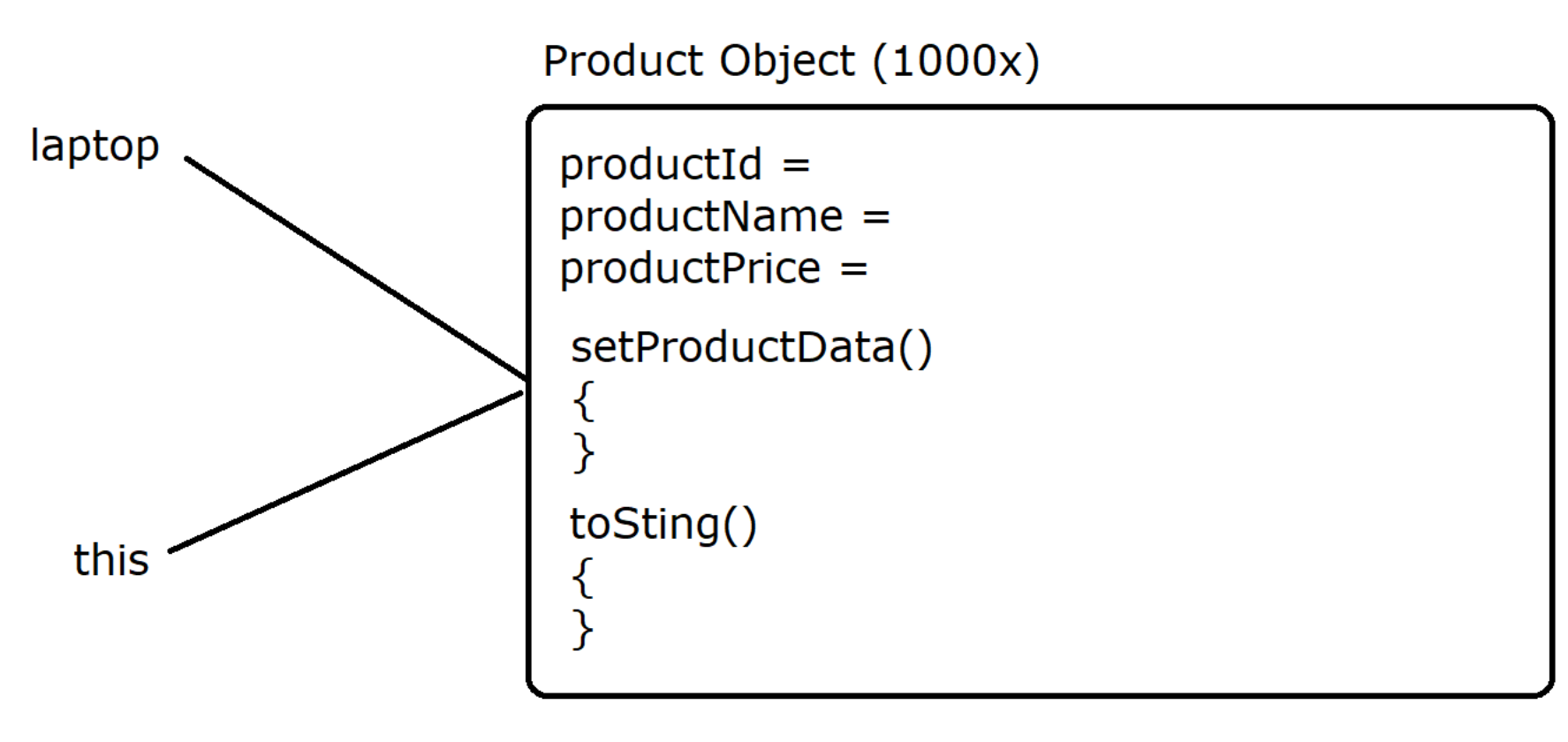
* If We want to represent instance variable from the method /constructor / block body then we should use this keyword, if instance variable and method level variables both are having same name.

Diagram for Product.java :

```java
package com.ravi.this_keyword;

public class ProductDemo
{
    public static void main(String[] args)
    {
        Product laptop = new Product();
        laptop.setProductData(98, "HP IRIS", 95000);
        System.out.println(laptop);
    }

}
```



* Whenever we create an object in java then the current object is also refer by this keyword as shown in the above diagram.

* By using this keyword we can represent the object member (non static fields + non static methods) from any non static area within the class.

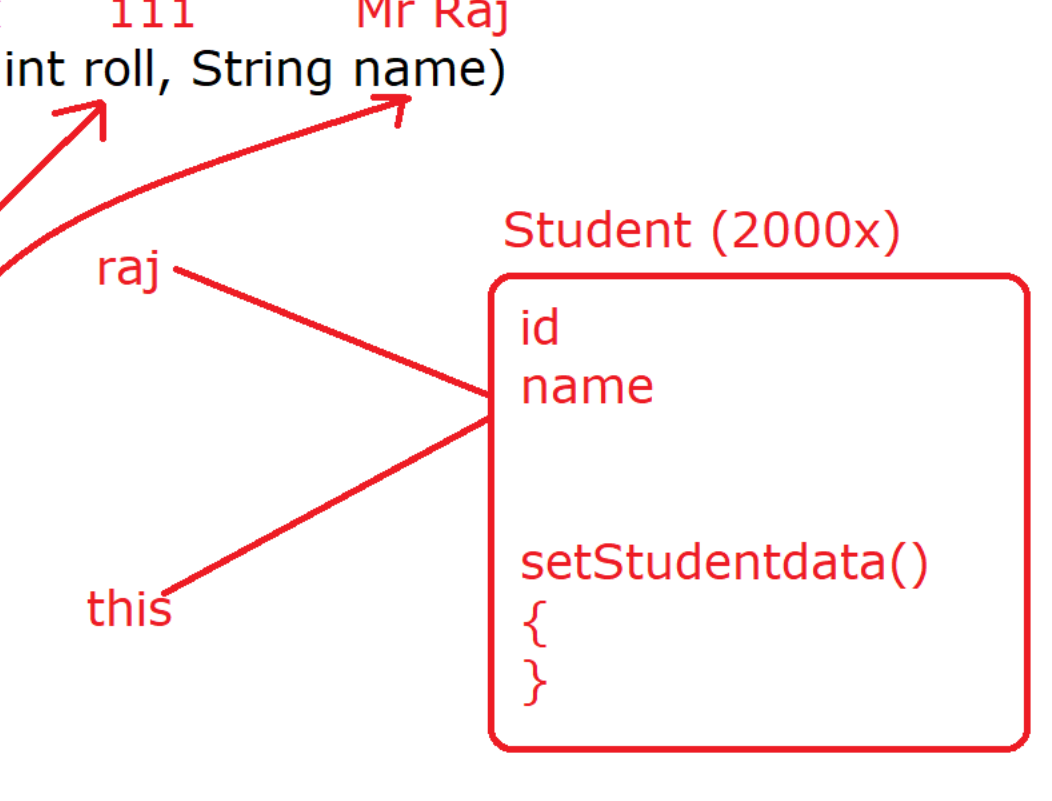
* WHENEVER WE CREATE AN OBJECT IN JAVA THEN AT THE TIME OF OBJECT CREATION AUTOMATICALLY COMPILER WILL ADD **final this reference variable** as a first parameter to all the non static methods and constructors.

Example :

```java
public class Student
{
  int roll;
  String name;

  public Student(final Student this)
  {
  }
                                          2000x      111        Mr Raj
  public void setStudentData(final Student this, int roll, String name)
  {
     this.roll = roll;
     this.name = name;
  }
}

Student raj = new Student();
     raj.setStudentData(raj, 111, "Mr Raj");
```



* this keyword we cannot use from static context. We can use from constructor and non static method.

//Program on this keyword :

```java
package com.ravi.this_keyword;

public class Product
{
    private int productId;
    private String productName;
    private double productPrice;

    public void setProductData(int productId, String productName, double 
productPrice)
    {
        this.productId = productId;
        this.productName = productName;
        this.productPrice = productPrice;
    }

    @Override
    public String toString()
    {
        return "Product [productId=" + productId + ", productName=" + productName + 
", productPrice=" + productPrice
                + "]";
    }

}
```

```java
package com.ravi.this_keyword;

public class ProductDemo
{
    public static void main(String[] args)
    {
        Product laptop = new Product();
        laptop.setProductData(98, "HP IRIS", 95000);
         System.out.println(laptop);
    }

}
```

What is method local search algorithm :
----
* Here we will learn the **variable searching algorithm** by compiler and JVM.
* Whenever we use any variable inside the method OR constructor then compiler will serch the variable declaration inside the method/constructor body first [Method Level], If not available then only It will search at class level.
* Based on the variable type compiler will also perform some operation as shown below :

```java
public class Test
{
   static int a = 100;
   int b = 200;

   public void accept(int c)
   {
      int d = 400;
       System.out.println(Test.a);
       System.out.println(this.b);
       System.out.println(c);
       System.out.println(d);
   }
}
public class ELC
{
   public static void main(String [] args)
   {
      Test t1 = new Test();
      t1.accept(300);
   }
}
```