Exercise 1: Implementing the Singleton Pattern

class singleton{

private static final singleton instance = new singleton();

private singleton(){

System.out.println("Singleton pattern is created");

}

public static singleton getInstance() {

return instance;

}

}

public class Main

{

public static void main(String[] args) {

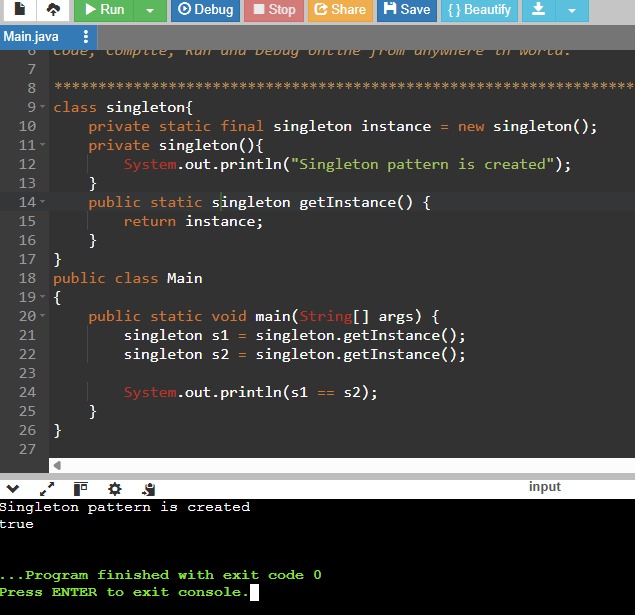
singleton s1 = singleton.getInstance();

singleton s2 = singleton.getInstance();

System.out.println(s1 == s2);

}

}

Exercise 2: Implementing the Factory Method Pattern

interface devices{

void devicetype();

}

class laptop implements devices{

public void devicetype(){

System.out.println("Laptop");

}

}

class Mobile implements devices{

public void devicetype(){

System.out.println("Mobile");

}

}

class Desktop implements devices{

public void devicetype(){

System.out.println("Desktop");

}

}

class devicefactory{

public devices gettype(String name){

switch(name){

case "laptop":

return new laptop();

case "mobile":

return new Mobile();

case "desktop":

return new Desktop();

default:

return null;

}

}

}

public class Main

{

public static void main(String[] args) {

devicefactory fact=new devicefactory();

devices dev1=fact.gettype("mobile");

dev1.devicetype();

}

}