FULLSTACKDEVELOPMENT–WORKSHEET5

**FINDOUTPUTOFTHEPROGRAMSWITHEXPLANATION**

Q1.//StringbufferpublicclassMain

{

public static void main(String args[])

{

String s1 = "abc";String s2 = s1;

s1 += "d";

System.out.println(s1 + " " + s2 + " " + (s1 == s2));StringBuffer sb1 = new StringBuffer("abc");StringBuffer sb2 = sb1;

sb1.append("d");

System.out.println(sb1 + " " + sb2 + " " + (sb1 == sb2));

}

}

Ans:

abcdabcd true

Explanation:In Java, String is immutable and string buffer is mutable.   
So string s2 and s1 both pointing to the same string abc. , after making the changes the string s1 points to abcd and s2 points to abc, hence false. Since string buffer are mutable, making changes in one string also make changes to the other string.

Q2.// Method overloadingpublic class Main

{

public static void FlipRobo(String s)

{

System.out.println("String");

}

public static void FlipRobo(Object o)

{

System.out.println("Object");

}

public static void main(String args[])

{

FlipRobo(null);

}

}

Ans: String

Explanation:In case of [method overloading](https://www.geeksforgeeks.org/overloading-in-java/), the most specific method is choosen at compile time. As ‘java.lang.String’ is a more specific type than ‘java.lang.Object’. In this case the method which takes ‘String’ as a parameter is chosen.

Q3.

class First

{

public First() {System.out.println("a"); }

}

class Second extends First

{

public Second(){System.out.println("b"); }

}

class Third extends Second

{

public Third(){System.out.println("c"); }

}

public class MainClass

{

public static void main(String[] args)

{

Third c = new Third();

}

}

Ans: a

b

c

Exaplanation: While creating a new object of ‘Third’ type, before calling the default constructor of Third class, the default constructor of super class is called i.e, Second class and then again before the default constructor of super class, default constructor of First class is called

Q4.

public class Calculator

{

intnum = 100;

public void calc(intnum){ this.num = num \* 10;}public void printNum() {System.out.println(num);}

public static void main(String[] args)

{

Calculator obj = new Calculator();obj.calc(2);

obj.printNum();

}

}

Ans: 20

Explanation: obj.cal(2) calls calc method and num value goes as 2 and hence num=2\*10=20. And obj.printnum() calls printNum method which prints value of num after operation that is 20.

Q5.

publicclassTest

{

public static void main(String[] args)

{

StringBuilder s1 = new StringBuilder("Java");String s2 = "Love";

s1.append(s2);s1.substring(4);

intfoundAt = s1.indexOf(s2);System.out.println(foundAt);

}

}

Ans: 4

Explanation: append*(*String*str)*method*,*concatenate the str to s1*.* The substring(int index) method return the String from the given index to the end. But as there is no any String variable to store the returned string,so it will bedestroyed.

Now indexOf(String s2) method return the index of first occurrence of s2. So 4 is printed as s1=”JavaLove”

Q6.

classWriter

{

public static void write()

{

System.out.println("Writing...");

}

}

classAuthorextendsWriter

{

public static void write()

{

System.out.println("Writingbook");

}

}

public class Programmer extendsAuthor

{

public static void write()

{

System.out.println("Writingcode");

}

public static void main(String[] args)

{

Author a = new Programmer();a.write();

}

}

Ans: Writing book

Explanation: Since static methods can’t be overridden, it doesn’t matter which class object is created. As *a* is a*Author* referenced type, so always *Author* class method is called.

Q7.

classFlipRobo

{

public static void main(String args[])

{

String s1 = new String("FlipRobo");String s2 = new String("FlipRobo");if (s1 == s2)

System.out.println("Equal");

else

}

}

System.out.println("Not equal");

Ans: Not equal

Explanation: Since, s1 and s2 are two different objects the references are not the same, and the == operator compares object reference.

Q8.

classFlipRobo

{

public static void main(String args[])

{

try

{

System.out.println("First statement of try block");

intnum=45/3;

System.out.println(num);

}

catch(Exception e)

{

System.out.println("FlipRobo caught Exception");

}

finally

{

System.out.println("finally block");

}

System.out.println("Main method");

}

}

Ans: First statement of try block

15

finally block

Main method

Explanation: Since there is no exception, the catch block is not called, but the finally block is always executed after a try block whether the exception is handled or not.

Q9.

classFlipRobo

{

// constructorFlipRobo()

{

System.out.println("constructor called");

}

staticFlipRobo a = new FlipRobo(); //line 8

public static void main(String args[])

{

FlipRobob;//line12b = new FlipRobo();

}

}

Ans: constructor called

constructor called

Explanation:

Q10.

classFlipRobo

{

staticintnum;staticStringmystr;

// constructorFlipRobo()

{

num = 100;

mystr = "Constructor";

}

// First Static blockstatic

{

System.out.println("Static Block 1");

num = 68;

mystr = "Block1";

}

// Second static blockstatic

{

System.out.println("Static Block 2");

num = 98;

mystr = "Block2";

}

public static void main(String args[])

{

FlipRobo a = new FlipRobo();System.out.println("Value of num = " + a.num);System.out.println("Valueofmystr="+a.mystr);

}

}

Ans: Static Block 1

Static Block 2

Value of num = 100

Value of mystr = Constructor

Explanation:Static block gets executed when the class is loaded in the memory. A class can have multiple Static blocks, which are executed in the same sequence in which they have been written into the program.