ASSIGNMENT-4

COURSE CODE-CS261
COURSE NAME-OBJECT ORIENTED DESIGN AND PROGRAMMING

Q1) Write a program to print sound of barking of dog , if Dog then sound of barking is woof, if hound which is a type of dog then sound will be bowl. Use function overriding.

ANS.

```
class dog
{
  public void sound()
  {
        System.out.println("woof");
  }
}
class hound extends dog
{
        public void sound()
        {
                System.out.println("bowl");
        }
}
class bark
{
        public static void main(String args[])
        {
                dog a=new dog();
                dog b=new hound();
                b.sound
        }
}
```

Q2) Write a program to illustrate why multiple inheritance is not supported in Java.

ANS. In java multiple inheritance is not supported because if we create two classes which have two functions of the same name then it will become difficult for the compiler to choose which of the two function have to be executed.

```
class father
{
	public void character()
	{
		System.out.println("Talking");
	}
}
class mother
{
	public void character()
	{
		System.out.println("cooking");
	}
}
class child extends father,mother
{
	public static void main(String args[])
	{
		child c=new child();
		c.character();
	}
}
```

Q3) Predict the output and justify the result.

i)

```
class Base {
private void fun() {
System.out.println("Base fun");
}
}
class Derived extends Base {
private void fun() { // overrides the Base's fun()
```

```
System.out.println("Derived fun");
}
public static void main(String[] args) {
Base obj = new Derived();
obj.fun();
}
}
```

ANS.

The output of the above program will be an error,because method of class Base is defined private and we know that private function cannot be inherited therefore we cannot override the private function.

Q4) Predict the output

```
class Base {
private void fun() {
System.out.println("Base fun");
}
}
class Derived extends Base {
public void fun() { // overrides the Base's fun()
System.out.println("Derived fun");
}
public static void main(String[] args) {
Base obj = new Derived();
obj.fun();
}
}
```

ANS.

The output of the above program will be an error message,due to the private access of the method define in Base class.

Q5) Predict the output and justify the result

```
class Parent {
```

```
static void m1()
{
System.out.println("From parent "
+ "static m1()");
}
void m2()
{
System.out.println("From parent "+ "non-static(instance)
m2()");
}
}
class Child extends Parent { static
void m1()
{
System.out.println("From child static m1()");
}
@Override
public void m2()
{
System.out.println("From parent "
+ "non-static(instance) m2()");
}
}
class Main {
public static void main(String[] args)
{
Parent obj1 = new Child();
obj1.m1();
obj1.m2();
}
}
```

ANS.

The output of the above progarm will be:-

From parent static m1()
From parent non-static(instance) m2()

From the above output we can justify that the static member cannot be override.

Q6) Predict the Output and justify your answer

```
class SuperClass {
SuperClass get() {
System.out.println("SuperClass");
return this;
}
}
public class Tester extends SuperClass {
Tester get() {
System.out.println("SubClass");
return this;
}
public static void main(String[] args) {
SuperClass tester = new Tester();
tester.get();
}
}
```

ANS.

The output of the above program will be:-
SubClass

In java we have the property that if we have two function of same name defined in parent as well as in child function with differnt return type but the return type of the child class should be the sub type of of the return type of the parent class then it can be overriden.This is called covarient return type in java.

Q7) Swap return type of get() and see the result and justify

```
class SuperClass {
```

```
Tester get() {
System.out.println("SuperClass");
return this;
}
}
public class Tester extends SuperClass {
SuperClass get() {
System.out.println("SubClass");
return this;
}
public static void main(String[] args) {
SuperClass tester = new Tester();
tester.get();
}
}
```

ANS.

The output of the above program will be an error,because the return type of the Tester class is not the subtype of the return type of SuperClass class.So, therefore it cannot be override.

Q8) Predict output and justify

```
class Parent {
// private methods are not overridden
private void m1()
{
System.out.println("From parent m1()");
}
public void m2()
{
System.out.println("From parent m2");
}
}
class Child extends Parent {
// new m1() method
```

```
private void m1()
{
System.out.println("From child m1");
}
@Override
protected void m2()
{
System.out.println("From child m2");
}
}
// Driver class
class Main {
public static void main(String[] args)
{
Parent obj1 = new Parent();
obj1.m2();
Parent obj2 = new Child();
obj2.m2();
}
}
```

ANS.

The output of the above class will be an error because protected is more restricted then public (i.e.protected is of less access privilege) and we cannot define less privileged access modifier in place of more privileged access modifiers.

Q9) Write a program to access private member of parent class inside child class

ANS.

```
class A
{
	private int x;

	public int access(int a)
	{
```

```
            x=a;
            return x;
          }
}
class B extends A
{
          public static void main(String args[])
          {
            B o=new B();
            int m=o.access(15);
        System.out.println(m);
          }
}
```

NAME-AMAN KUMAR KANOJIA
ROLL NO.-201851014
SECTION-1
GROUP-1A