Uttara InfoSolutions

www.uttarainfo.com

Uttara Practicals 2:

(See example demo code, compile and execute first)

1) There are Cows. A Cow has a name and can moo. When you ask a Cow to moo, it shouts out its name. Design a class for Cow and the tester code. After testing if cows work correctly, create 2 references and then point to same object. Verify what happens if you change the state of the object using one ref when you invoke behaviour using the second.

```
Cow
        String name;
        public void moo()
        {
                SOP("I am cow and I am moo mooing…"+name);
        }
```

2) Check if instance variables of type string, int and boolean are given default values. Check if local variables are given default values.
How to do this? Create a class with 4 instance variables with String, boolean, int and boolean as datatypes. Create a Tester class with main where you create an object and using reference print the instance variable values to monitor using SOP.
Ex:

```
class X
{
        int i;
        String str;
```

```
    public static void main(String[] args)
    {
        X obj = new X();
        System.out.println(obj.i);
        System.out.println(obj.str);
    }
}
```

3) Test passing reference as parameter. Create a Fish class with String name in it. Create a Croc class with String name and a method called eat().Accept a Fish reference as parameter. In the eat(), print both the crocs name and fish`s name (public void eat(Fish f)). In TestPass class, code a main() where you should create a croc object, a fish object, assign names “cocky” and “fishy” and invoke eat() and pass fish ref as parameter.Print the fish`s name before calling eat() and after calling eat(). Run and test.

<u>Test the following as well:</u>

a) in eat(), after SOP, assign f to null. Check what happens.
b) in eat(), assign f to new Fish object and assign name “Flippy”. Now check the SOPs
c) in eat(), assign f.name to “Flippy” and observe outputs. What happens here? How are references getting passed?
d) pass null to eat() and check what happens

4) There are Persons. A person has a age, name. Persons can eat, sleep. A person sleeps more if his age is < 40 and eats less. If the persons age is >= 40, he sleeps less but eats more. Design a Person class and test it.

```
Person
    String name;

    int age;
    public void eat() {

        if(age < 40) sdfsdkf

        else
            adfkgldafgk
```

```
}
public void sleep() {

}
```

5) There are Dogs. Every Dog has a name and a size. Dogs can bark. If the size of the dog is > 10, it "meows". If the size <=10, then as many times, it "bow wow" its

name to the monitor. Test Dog design. After testing the same, make the size variable private and then add setSize()/getSize() method. See how this impacts your tester class.

6) Cars can be started, driven, reversed or stopped. You have to start the car to drive/reverse/stop it. When car is being driven / reversed, the fuel reduces. Once the car has no fuel, the car stops. Every car has a name. Write a tester class to test cars.

```
Car
    - name
    - fuelQty
    - isStarted : boolean
    - start()
        // if fuelQty > 0
        //      set isStarted = true;
    - drive()

        // check if the car has started, only if
        yes,
        if(isStarted)
            // check if fuelQty>0, then only drive
            // reduce fuelQty by 1 unit
            // if fuelQty==0
                stop();

    - reverse()
    - stop()
        // isStarted = false!
```

7) Test Math class methods using following code sample (try for different values):

```
int x = 100;
```

```
double d1 = Math.sqrt(100);
double d2 = Math.cbrt(200);
double d3 = Math.random();
System.out.println("d1 = "+d1);
System.out.println("d2 = "+d2);
System.out.println("d3 = “+d3);
```

Now print 10 random integer numbers between 0-100 to the monitor in a for loop.

8)WAP to add the sum of the digits of a given int input and then test whether it is even. Use Steps-to-program.

9)WAP to test whether an int input is a power of 3. Use Steps-to-program.

10) WAP to test whether a given input int number is a square.
first find the sqrt of the num
double d = Math.sqrt(num);
if (d - (int)d) == 0) then it is a square
or
if (((int)d)*((int)d) == num) then is square

11) WAP to create an array with 10 size and of int holding ability. Store values 1-10 in it. Using the length variable, access each element of the array and print it to the console. Create the array in 2 ways - one using the new operator and then storing the values individually, two by creating the array with the values directly in the array. Loop over the array using an index variable.Also loop over it using for-each loop.

Ex:

```
int[] arr1 = new int[10]; // to create an empty array

int[] arr2 = {10,20,30}; // to create a literal array


for(int i = 0 ; i < arr1.length ; i++) // to loop over the array

   System.out.println("value in "+i+"th box = "+arr1[i]);
```

```
for(int val :  arr2)
        System.out.println(val);
```