BSCCS2005: Practice Assignment
Questions with Solutions
Week 3

1. Consider the following code. [Nalini: MCQ: 2points]

```
public class Example1{
    public int a = 10;
    public void display( ){
        System.out.print("In parent class");
    }
}
public class Example2 extends Example1{
    public int b = 20;
    public void display( ){
        System.out.println("In child class");
    }
    public static void main(String[] args) {
        Example1 obj = new Example2( );
    }
}
```

What is the output of `obj.display( )`?

- ◯ In parent class
- √ **In child class**
- ◯ Compiler Error
- ◯ In parent class
  In child class

Solution:

- obj is a reference variable of type Example1 but is referring to an object of type Example2.
- display( )method is overridden in subclass.
- So calling obj.display( ) prints In child class.

2. Consider the following code. [Nalini: MSQ: 2points]

```
public class Polygon {
    public static void perimeter( ){
        System.out.print("In Polygon perimeter");
    }
    public void area( ) {
        System.out.println("In Polygon area");
    }
    public void sides( ){
         System.out.println("In Polygon sides");
    }
    public void angleSum( ) {
        System.out.println("In Polygon angleSum");
    }

}
public class Pentagon extends Polygon{
    public static void perimeter( ){
        System.out.println("In Pentagon perimeter");
    }
    public void area( ) {
         System.out.println("In Pentagon area");
    }
    public int sides( ){
         return 5;
    }
    public void angleSum(int x) {
        System.out.println("In Pentagon angleSum");
    }
}
```

Which method/s hide/s methods in the parentclass?

- √ **perimeter**
- ◯ area
- ◯ sides
- ◯ angleSum

Solution:

- If the static methods of same signature are present in both parentclass and subclass then static methods of parentclass are hidden in subclass

3. Consider the following code. [Nalini: MSQ: 2points]

```
1    public class Animals{
2        final public void swim( ){
3            System.out.println("Animals swim");
4        }
5        public static void communicate( ){
6            System.out.println("Animals communicate");
7        }
8    }
9    public class Birds extends Animals{
10        public void swim( ){
11            System.out.println("Birds swim");
12        }
13        public void communicate( ){
14            System.out.println("Birds communicate");
15        }
16        public void fly( ){
17            System.out.println("Birds fly");
18        }
19   }
```

Identify the line/s which has/have error.

- ◯ Line 1
- ◯ Line 2
- ◯ Line 5
- ◯ Line 9
- √ **Line 10**
- √ **Line 13**
- ◯ Line 16

Solution:

- final methods cannot be overridden
- static methods cannot be overridden

4. Identify the steps in implementation of:
   A. structured programming
   and
   B. object-oriented programming.

   1. Identify the data to be manipulated and maintained
   2. Design a set of procedures for specific tasks and combine them to build a complex system
   3. Devise / identify algorithms to operate on the data
   4. Design data structures to suit procedural manipulations

   [ARUP: MCQ: 2 points]

   ◯ A. 1 → 4
   B. 2 → 3

   ◯ A. 1 → 3
   B. 2 → 4

   ◯ A. 1 → 2
   B. 3 → 4

   √ **A. 2 → 4**
   B. 1 → 3

   Solution: In structured programming,

   - First, design a set of procedures for specific tasks and combine them to build a complex system
   - Then, design data structures to suit procedural manipulations

   In object oriented programming,

   - First, identify the data need to be maintained and manipulated
   - Then, identify algorithms to operate on the data

5. Consider the Java code given below. [ARUP: MSQ: 2 points]

```
public class Parent{
    public void f1(){
        System.out.println("f1() called from parent class");
    }
}
public class Child extends Parent{
    public void f1(){
        System.out.println("f1() called from child class");
    }
}
public class FClass{
    public static void main(String[] args){
        //LINE 1
    }
}
```

Which of the following statements at `LINE 1` will generate the output as:
`f1() called from parent class`

- √ `new Parent().f1();`
- ◯ `((Child)new Parent()).f1();`
- ◯ `new Child().f1();`
- ◯ `((Parent)new Child()).f1();`

Solution:

- In option-1, `new Parent()` creates an unnamed object of `Parent` type, so `f1()` will be called from `Parent` class. Thus, output is `f1() called from parent class`.
- In option-2, `new Parent()` creates an unnamed object of `Parent` type. However, type casting from a `Parent` type to `Child` type (subtype) is illegal. Thus, it generates compiler error.
- In option-3, `new Child()` creates an unnamed object of `Child` type, so `f1()` will be called from `Child` class. Thus, output is `f1() called from child class`.
- In option-4, `new Child()` creates an unnamed object of `Child` type, which is type casted to `Parent` type. Although, the static type of the object is `Parent`, dynamic type is `Child`. Thus, dynamic dispatch calls the `f1()` from `Child` class. Thus, the output is `f1() called from child class`.

6. Consider the Java code given below. [ARUP: MCQ: 2 points]

```
public class Rectangle{
    private final void computeArea(){
        System.out.println("area of rectangle");
    }
}
public class Cube extends Rectangle{
    public final void computeArea(){
        System.out.println("area of cube");
    }
}
public class FClass{
    public static void main(String[] args){
        Cube r = new Cube();
        r.computeArea();
    }
}
```

What will be the output?

◯ `area of rectangle`
`area of cube`

◯ `area of rectangle`

√ `area of cube`

◯ It generates compiler error

Solution: Although `final` keyword makes a method not overridable, the `computeArea()` in `Rectangle` is invisible outside of the class. Thus, when the method redefined `computeArea()` in `Cube` it would not be considered as overriding, rather it is considered as a new method defined in `Cube`. Thus, it prints `area of cube`.

7. Consider the code segment given below. [ARUP: MSQ: 2 points]

```
public class Data{
    //definition of MAX as constant
}
public class FClass{
    public static void main(String[] arg){
        System.out.println(Data.MAX);
    }
}
```

Consider that `MAX` is defined as constant in `class Data`. Identify the most appropriate definition(s) of `MAX` in `class Data` such that the output of the code is `100`.

- ◯ `public final int MAX = 100;`
- ◯ `private final static int MAX = 100;`
- ◯ `private static int MAX = 100;`
- √ `public final static int MAX = 100;`

Solution: Since `MAX` is accessed using the class name, it must defined as `static`.
Since we have considered that `MAX` is a constant, , it must defined as `final`.
Since `MAX` is accessed from the outside of the class where it is defined, it must be defined with the default or `public` access specifier.

8. Consider the Java code given below. [ARUP: MCQ: 2 points]

```
public class Employee{
    private int empid;
    private String name;

    public Employee(int empid_, String name_){
        empid = empid_;
        name = name_;
    }
    public Employee(){
        this(0, "unknown");
    }
    public void print()
        System.out.print(empid + " : " + name + " : ");
    }
}
public class Manager extends Employee{
    private String department;

    public Manager(int empid_, String name_, String department_){
        department = department_;
    }
    public void print(){
        super.print();
        System.out.println(department);
    }
}
public class FClass{
    public static void main(String[] args){
        Manager m = new Manager(101, "Nutan", "HR");
        m.print();
    }
}
```

What will be the output?

◯ `0 : unknown`

√ `0 : unknown : HR`

◯ `HR`

◯ `101 : Nutan : HR`

9. Consider the following code. [Bhaskar:MCQ:2points]

```
public class A{
    public float g;

    public A(){
        g = 9.8f;
    }
    public void show(){
        System.out.println("g = "+g);
    }
}
public class B extends A{
    public double e;

    public B(double num){
        e = num;
    }
    public void show(){
        System.out.println("e = "+e);
    }
}
public class FClass{
    public static void main(String args[]){
        A obj1 = new B(2.718);

        A ptr1 = (A)obj1;
        A ptr2 = (B)obj1;
        B ptr3 = (B)obj1;

        ptr1.show();
        ptr2.show();
        ptr3.show();
    }
}
```

What will be the output of the given code?

- ◯ g = 9.8
  g = 9.8
  e = 2.718
- ◯ g = 9.8
  e = 2.718
  e = 2.718

√ **e = 2.718**
e = 2.718
e = 2.718

◯ g = 9.8
g = 9.8
g = 9.8

Solution:
The method `show()` is overridden in the child class therefore decision regarding which version of `show()` is to be executed is done at runtime depending upon the instance/object type (and not on reference type).
`(A)obj1`, `(B)obj1` is casting the type of reference not object, the object's type is *B* in all the cases and thus `show()` method of *class B* is invoked in each of them.

10. Consider the following code. [Bhaskar:MCQ:2points]

```
1. public class Mammal{
2.     public String name;
3.     public int lifespan;
4.
5.     public Mammal(){
6.         name = "Giraffe";
7.         lifespan = 45;
8.     }
9.
10.    public void show(){
11.         System.out.println("Giraffe");
12.         System.out.format("name = %s : lifespan = %d",name,lifespan);
13.    }
14. }
15.
16. public class Endangered extends Mammal{
17.     public boolean endanger_status;
18.
19.     public Endangered(){
20.         endanger_status = false;
21.     }
22.
23.     public void show(){
24.         System.out.println("endanger status of "+this.name
                                  + " is "+endanger_status);
25.     }
26.     public void display(){
27.         this.show();
28.     }
29. }
30.
31. public class FClass{
32.     public static void main(String args[]) {
33.         Mammal m1 = new Endangered();
34.         m1.show();
35.         m1.display();
36     }
37. }
```

Choose the correct option regarding the given program.

◯ The program generates output:
name = Giraffe : lifespan = 45
endanger status of Giraffe is false

◯ The program generates output:
endanger status of Giraffe is false
endanger status of Giraffe is false

◯ The program produces a compilation error on `LINE 34`

√ **The program produces a compilation error on** `LINE 35`

◯ The program produces a compilation error on `LINE 23`

Solution:

In this example the `show() method of Mammal class is overridden in the Endangered class but display() method is a method of Endangered class only. Therefore display() method will always be invoked depending upon the reference type and hence it can be invoked by a reference of Endangered type.`
`But here m1 is a reference of Mammal type and hence the statement m1.display() produces a compilation error.`

11. What will be the output of the following Java program?

[Bikash:MCQ:2 points:Lecture 3.2]

```
public class Numbers{
    public int x = 1;
    public double y = 2.1;

    public Numbers(int a, double b){
        add(a,a);
        add(b,a);
    }
    public void add(int a, int b){
        this.x = a + b;
    }
    public void add(double c, int d){
        this.y = c + d;
    }
}
public class Example{
    public static void main(String args[]){
        Numbers obj = new Numbers(2,3.2);
        System.out.println(obj.x + " " + obj.y);
    }
}
```

◯ Compilation failed.

◯ 1 2.1

◯ 4 6.4

√ **4 5.2**

Solution: The steps of execution of the above program is:

- We have a parametric constructor that accepts an `int` and a `double`.
- This constructor calls the 2 overloaded methods `add`, one after another, depending upon the passed arguments.
- These methods changes the values of x and y respectively and we get the desired output.

12. Consider the Java program given below and select the correct statement.

[Anand : MCQ : 2 points]

```
public class A{
    private int a=10;
    private int b=20;
}
public class B extends A{
}
public class Example{
    public static void main(String[] args){
        B ob=new B();
        A oa=new A();
        ob.a=20;     // Line 1
        System.out.println(oa.a);  // Line 2
        System.out.println(ob.a);  // Line 3
    }
}
```

◯ Illegal access of variable `a` at Line 1.

◯ Illegal access of variable `a` at Line 2.

◯ Illegal access of variable `a` at Line 3.

√ **All of the above.**

Solution: Line 1 it's not possible to change the private instance variable of super class directly.
Line 2 it's not possible to access/change the private instance variable outside the class.
Line 3 it's not possible to access the private instance variable of super class directly.

13. Consider the statements given below and select the correct option.

[Anand : MCQ : 2 points]

Statement 1: If the same static method exists in both super class and subclass, then the subclass method hides the super class method.
Statement 2: Both the variables and the methods in a super class can be overridden in a subclass.

- ◯ Both statements are true.
- ◯ Both statements are false.
- √ **Statement 1 is true and Statement 2 is false.**
- ◯ Statement 1 is false and Statement 2 is true.

Solution: Static methods of super class can be hidden by the subclass methods; overriding of static methods in subclass is not possible.
Methods are polymorphic, we can override the methods in subclass of parent class.
Variables are not polymorphic, we cannot override the variables in subclass of parent class.

14. Consider the Java program given below and predict the output.

[Anand : MCQ : 2 points]

```
public class A{
    public static void print(){
        System.out.println("Class A print");
    }
    public void show(){
        System.out.println("Class A show");
   }
}
public class B extends A{
    public static void print(){
        System.out.println("Class B print");
    }
   public void show(){
        System.out.println("Class B show");
   }
}
public class MyClass{
    public static void main(String args[]){
        A oa=new B();
        oa.print();
        oa.show();
    }
}
```

- √ **super class print**
  sub class show
- ◯ super class print
  super class print
- ◯ sub class show
  sub class show
- ◯ Compilation failed.

Solution: print() method is overridden in the subclass because it is a static method. show() overridden in the subclass because it is non-static method.

15. What will be the output of the following Java program?

[Bikash:MCQ:2 points:Lecture 3.3]

```
public class College{
    public void name(){
        System.out.println("IIT Madras.");
    }
}
public class Example extends College{
    public void name(){
        super.name();
        System.out.println("University of Calcutta.");
    }
    public static void main(String[] args){
        Example obj = new College();
        obj.name();
    }
}
```

√ **Compilation fails.**

◯ IIT Madras.
University of Calcutta.

◯ IIT Madras.

◯ University of Calcutta.

Solution:
The reference variable of the Parent class is capable of holding its object reference as well as its child object reference. But the reference variable of the child class can only hold its object reference. Hence, we can create an object of the parent class by taking reference of the child class.

Consider the Java program given below and answer the questions 16 to 18.

[Bikash:MCQ:2 points:Lecture 3.3]

```
public class A{
    public String features(){
        return "Write Once, Run Anywhere.";
    }
}
public class B extends A{
    public String features(){
        return "Object Oriented.";
    }
}
public class C extends A{
    public String features(){
        return "Multithreaded.";
    }
}
public class Example{
    public static void main(String args[]){
        A obj1 = new A();
        A obj2 = new B();
        A obj3 = new C();
        System.out.print(obj1.features()+"\n"
        +obj2.features()+"\n"+obj3.features());
    }
}
```

16. The above program is an example of

    √ **Dynamic dispatch.**

    ◯ Compile time polymorphism.

17. The output of the above program is:

```
Write Once, Run Anywhere.
Write Once, Run Anywhere.
Write Once, Run Anywhere.
```

    ◯ True

    √ **False**

18. What is the return type of the overridden method?

    ◯ void

√ **String**

Solution:
(a) The above program is a sheer implementation of dynamic dispatch. Based on the object being referred, the appropriate overridden method will be called at the run time.

(b) The output of the above program is:

```
Write Once, Run Anywhere
Object Oriented.
Multithreaded.
```

19. Consider the Java program given below and predict the output.

[Anand : MCQ : 2 points]

```
public class A{
    public void show(){
        System.out.println("A class show()");
    }
}
public class B extends A{
    public void show(){
         System.out.println("B class show()");
    }
}
public class C extends B{
}
public class  Example{
    public static void main(String[] args){
        C oc=new C();
        oc.show();
    }
}
```

◯ Compilation failed

◯ Nothing will be printed

◯ A class show()

√ **B class show()**

Solution: oc.show();
The above statement will first look into class C for show(). Since it does not find show() method in C, it will look into its base class B. If show() was not available in B, then it would look inside class A and so on.
show() available in B, therefore control will execute the show() from class B.

20. Consider the Java program given below and predict the output.

[Anand : MCQ : 2 points]

```
public class A{
    public A(){
        System.out.println("A() called");
    }
}
public class B extends A{
    public B(){
        System.out.println("B() called");
    }
}
public class C extends B{
    public C(){
        System.out.println("C() called");
    }
}
public class  Example{
    public static void main(String[] args){
        C oc=new C();
    }
}
```

◯ A() called

◯ C() called

◯ B() called
C() called

√ **A() called**
B() called
C() called

Solution: Whenever objects are created for subclass automatically object is created for base class also, whenever object created for class C its base class B also gets instantiated, Here class B has its parent class A also gets instantiated.
All constructors can be called in above program.

21. What will be the output of the following Java program?

[Bikash:MCQ:2 points:Lecture 3.6]

```
public final class Music{
    public void artist(){
        System.out.println("Anupam Datta");
    }
}
public class Poet extends Music{
    public void artist(){
        super.artist();
        System.out.println("Rabindranath Das");
    }
}
public class Example{
    public static void main(String args[]){
        Music obj = new Poet();
        obj.artist();
    }
}
```

◯ Anupam Datta

◯ Rabindranath Das

◯ Anupam Datta
Rabindranath Das

√ **Compile time error.**

Solution:
The class `Music` has been declared `final`. Hence, it cannot be inherited by any other class. The compiler will immediately throw an error after detecting that we are trying to extend a `final` class.