COMP90041 Programming and Software Development Semester 1, 2016 **Sample Answer**

1-5 CBCDE 6-10 BCABA 11-15 BACDE

16. Generics allow the same code to be used for different class types **(1 mark)**. An example of generic class we have learned is `ArrayList` (note, this is not interface, so `Iterator`, `Comparable`, etc are wrong here) **(1 mark)**.

17. `public`, `protected`, `private` **(1 mark)**; `private` **(1 mark)**

18. It outputs *"An apple"* **[1 mark]**. An object has both a type and an actual body (memory allocation) associated with it. An **upcasting** changes its type only, but not its body. Since the actual body is defined as "new Apple();", it will use the method in the class Apple **[1 mark]**. (**NOTE:** The second mark should be given if the student explained from the point of view of dynamic / late binding or polymorphism instead of upcasting.)

19. Method overloading means that within one class, there are more than one definitions of a single method name in a class; i.e., there are multiple method signatures for one method name.**[1 mark]** Method overriding means that a derived class redefines an inherited method in its base class.**[1 mark]**

For example, for a `Student` class, it may have multiple constructors with different types of parameters, e.g., (given name, family name) and (given name, family name, age), etc. Here, overloading is used. **[1 mark]**

For a `PostgraduateStudent` class extending the `Student` class, it may redefine the method `graduate()` in class `Student`, as the procedure for a postgraduate to graduate may differ from a general student. Here method overriding is used. **[1 mark]**

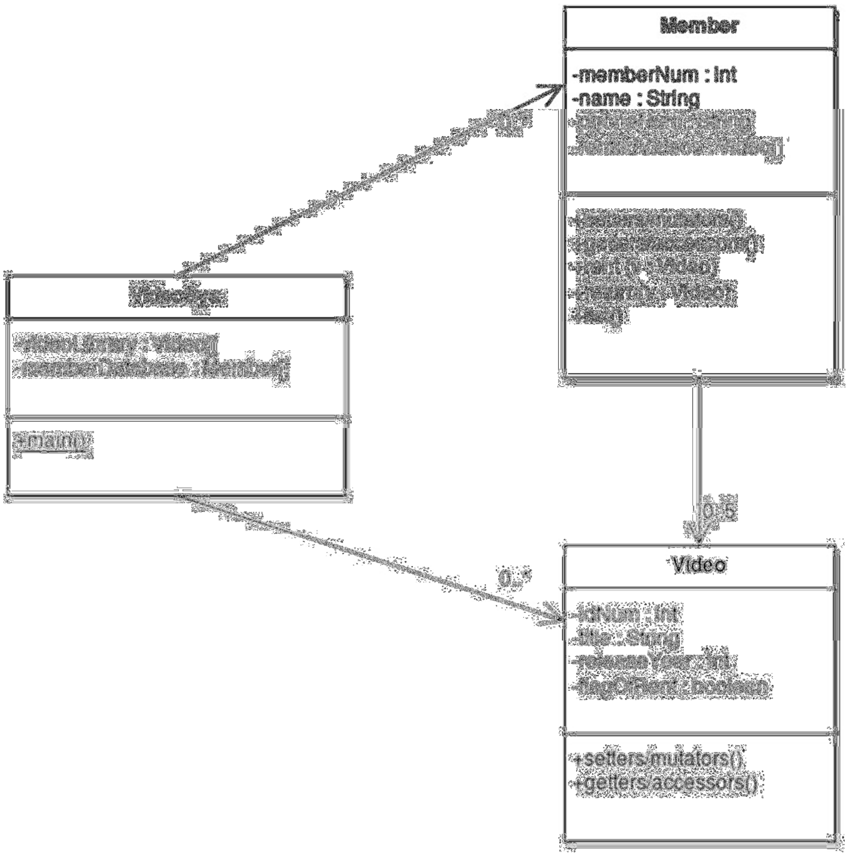
20.
```
import java.util.Scanner; (0.5 marks)
    public class StarTriangle {(1 mark)
        public static void main(String[] args) {
            Scanner s = new Scanner(System.in); (0.5 marks)
            System.out.println("Enter an integer:");
            int n = s.nextInt(); (1 mark)
            for (int i = 0; i < n; i++) { (1 mark)
                for (int j = 0; j <= i; j++) { (1 mark)
                    System.out.print("*");(0.5 marks)
                }
                System.out.println("\n"); (0.5 marks)

            }
        }
    }
```

21.
```
public int [] myArray = new int [10];(2 marks)
    for (int i = 0; i < 10; i++ ) {(2 mark)
        myArray[i] = i * i;(1 mark)
    }
```

22.
```
public interface XYInterface {(1.5 marks)
        (public) final static X = 5;(1.5 marks)
        (public) final static Y = 10;(1.5 marks)
        public int useXY();(1.5 marks)
   }
```

23. overall structure **(1 mark)**; each class **(1 mark)***3; associations **(1 mark)*3**.



24.
```
import java.util.Scanner; (1 mark)
  public class PrimeNumber {(1 mark)
       public static void main(String[] args) {
            Scanner s = new Scanner(System.in);
            System.out.println("Enter a number:");
            int n = s.nextInt(); (1 mark)
            //It is correct if i < n is used instead of i * i <= n
            for (int i = 2; i * i <= n; i++) {
                 if (n % i == 0) {
                      System.out.println(n + " is not a prime
       number.");
                      return;
                 }
            } (3 marks)
            System.out.println(n + " is a prime number.");
       }
  }
```

25.
```
public class InvalidDateException extends Exception (1 mark) {
        public InvalidDateException ()(1 mark) {
             super("Invalid date!"); (1 mark)
        }
        public InvalidDateException (String message) (1 mark) {
             super(message); (1 mark)
        }
   }
```