Fardin Ahmed
Professor Zhao
CISC 3115
November 8th, 2023

```
// 13.1
/*
 * Option (e)
 * abstract class A{
 * abstract  void unfinished();}
 */

 //13.3
 /*
  * a. False.
  * b. True.
  * c. False.
  * d. False.
  * e. True.
  */

  //13.4
  /*
   Number numberRed=new Integer(0);
   Double doubleRef=(Double)numberRef;
```

Number is a superclass of Double and Integer during compile-time but the object referred to by numberRef is is an instace of Integer which cannot be casted to a double. It throws exception during runtime.

Correct code:

```
Number numberRef=0;
Double doubleRef= new Double(numberRef.doubleValue());  */

   //13.5
   /*
    *Number[] numberArray=new Integer[2];
    numberArray[0]=new Double(1.5);
```

Double cannot be stored in an Integer array, so it causes runtime error.

Correct code:

```
Number[] numberArray=new Number[2];
numberArray[0]=new Double(1.5);   */

//13.6
*/ Output:
3
3.0
/*

//13.7
*/ new Integer(int) has been deprecated since java 9
Constructor for wrapper classes like Integer or Double deprecated.
Java automatically converts 'Integer' to int.

Corrected Code:
public class Test{
Public static void main(String[]args){
Integer x=3
System.out.println(x.intValue());
System.out.println(x.compareTo(4));
}
}            /*

// 13.8
*/
 The code doesn't use autoboxing
Corrected code:

public class Test{
Public static void main(String[]args){
Integer x=3
System.out.println(x.intValue());
System.out.println(((Integer)x).compareTo(4));
}
}   /*

//13.13
*/
Instance cannot be created new A() because interfaces are abstract and cannot be
instantiated.
/*

//13.14
*/ Reference variable x with Type A can be declared. /*
```

//13.15
*/
Correct choice : (d)
```
 interface A{
  void print()
}         /*
```

//13.16
*/
M1 by default has a package private access. It has to be declared as public

```
  class B implements A{
          public void m1(){
          System.out.println (“m1”);
          }
          }         /*
```

//13.17
/ True /

//13.18
/ public int compareTo(String o); /

//13.19

*/ Integer n1=new Integer(3);
Object n2= new Integer(4);
System.out.println(n1.compareTo(n2));

Code cannot be compiled because compareTo method expects an Integer argument.
Correct Code:
Integer n1=new Integer(3);
Integer n2= new Integer(4);
System.out.println(n1.compareTo(n2));   /*

//13.20
/ Implementing Comparable interface is beneficial because it allows objects of a class to be used with algorithms that need ordering. /

//13.21
/ Person class does not implement a Comparable interface but in class Test its being used with Arrays.sort /

```
//13.22
/*
 * clone() method cannot be invoked without implementing java.lang.Cloneable
 *
 */

 //13.23
// Document doesn't have 13.11 listed.

//13.24
/*
 * true
 * false
 * true
 */

 //13.25
 /*
 true
false
list is [New York]
list1 is [New York]
list2.get(0) is New York
list2.size() is 1
  */

//13.26
/*Cloneable interface needs to be implemented and provide a public clone() method.
clone() as is is protected
 * by default.
*/

//13.29
/*
 * a. True.
 * b. True.
 * c.True.
 * d. Flase.
 * e. False.
 */
```