COMP16212
Notes on Mock Exam Questions
April/May 2019

Mock Exam

Attached you will find a Mock multiple choice question (MCQ) exam for COMP16212, to assist you in preparing for the actual COMP16212 MCQ Exam in May/June.

We have endeavoured to make this as much alike the real thing as possible. Here, there are 14 questions, and you should take 45 minutes to complete them (an average of 3.2 minutes per question). In the real exam you will have 25 questions, but 2 hours to complete them (an average of 4.8 minutes per question). The Marking Scheme on this mock exam is the same as will appear on the real exam.

When tackling the Mock Exam it is clearly important for you to do so under exam conditions:

- stick to the time limits
- do it on your own
- do not refer to textbooks or notes, the Internet, etc.

If you happen not to be able to complete it before we make the answers available, then you should *not* look at the answers until you have attempted the exam first.

Exam Technique and Marking Scheme

You may think that an MCQ is easier than a standard exam because the correct answer is there in front of you. Actually it is often harder, for example because there are no marks at all if you get most of a question right and then make a silly mistake leading you to the wrong answer.

Rather than merely reading the question and then picking the appropriate answer, often questions will require some thought and problem-solving, e.g. carefully reading code and working out how it would run. You should be prepared to do this.

In the Marking Scheme, you will see you score 4 marks for a correct answer. You will also see you score -1 marks for an incorrect answer **but you also score -1 marks for not giving any answer**.

So, if you are not sure of an answer, **YOU SHOULD DEFINITELY GUESS. You should return an answer for EVERY QUESTION on the exam paper**.

However, you should aim at the very least to eliminate as many options as you can before resorting to guessing. In general you should expect to have work hard to determine the correct answer.

COMP16212

Object-Oriented Programming with Java (II)

Mock Exam

Time allowed: 45 minutes

Please answer all questions on the question paper.

Use of electronic calculators is not permitted.

Each question has exactly one correct answer and should be answered by clearly writing the appropriate letter (A,B,C,D, or E) within the box provided. Any writing outside these boxes will be ignored.

Each question will be given equal weighting.

Correct answers achieve 4 marks.
Incorrect answers (including unanswered questions) will achieve -1 mark.

1. Which one of the following statements is TRUE about testing (according to the lecture from COMP16212)?

   (A) Black box testing is best done using structural testing, as it ensures that every line of code is executed.

   (B) White box testing is best done using equivalence partitioning as it helps to find input sets that should have the same results.

   (C) Test cases should be used when testing switch statements, whereas test data should be used when testing that files are handled properly.

   (D) Test data is chosen as a representation of a test case, and a test is regarded as successful if the result obtained using the test data matches the expected result for that test case.

   (E) Equivalance partitioning is more suitable for black box testing than it is for white box testing.

2. Given the code:

```
public class Machine
{
   public void go()
   {
      System.out.println("I must be going!");
   }

   public void go(int x)
   {
      System.out.println("Hi there! - your number is " + x);
      go();
   }
}


public class SpecialMachine extends Machine
{
   public void go(int x)
   {
      System.out.println("Hello!");
      super.go(x + 1);
   }
}
```

and the following code fragment:

```
Machine x = new SpecialMachine();
x.go(9);
```

which one of the following is printed when the above fragment is run?

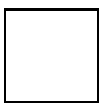
(A)
```
Hello!
Hi there!  - your number is 10
I must be going!
```

(B)
```
Hello!
Hi there!  - your number is 9
I must be going!
```

(C)
```
Hi there!  - your number is 10
I must be going!
```

(D)
```
Hi there!  - your number is 9
I must be going!
```

(E)
```
Hi there!  - your number is 10
```



3. Given the following class definitions, which make up a random investment portfolio, with 50% probability of adding a bond, and 50% of adding a stock:

```
public abstract class Asset { }

public class Bond extends Asset { }

public class Stock extends Asset { }


public class Test
{
   public static void main(String [] args)
   {
      Asset[] list = new Asset[5];
      for (int i = 0; i < list.length; i++)
         if (Math.random() >= 0.5)
            list[i] = new Bond();
         else
            list[i] = new Stock();

      int count = 0;
      for (int i = 0; i < list.length; i++)
      {
         // INSERT LINE HERE

         if (item instanceof Bond)
            count++;
      }

      System.out.println(count+" bonds");
   }
}
```

Which one of the following lines, inserted at '`// INSERT LINE HERE`', will correctly extract each item from the array and allow us to count how many bonds are in the portfolio?

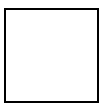
(A) `Object item = (Bond)list[i];`

(B) `Asset item = (Bond)list[i];`

(C) `Asset item = (Asset)list[i];`

(D) `Bond item = (Bond)list[i];`

(E) `Bond item = (Asset)list[i];`



4. Consider the following program:

```
import java.util.*;

public class MyClass
{
  public static void main(String [] args)
  {
    Integer [] myArray = {3, 1, 2};                          /*A*/
    List<Integer> list = Arrays.<Integer>asList(myArray);
    Set<Integer> set1 = new HashSet<Integer>();
    TreeSet<Integer> set2 = new TreeSet<Integer>(list);
    set1.addAll(set2);                                       /*B*/
    set1.add(4);                                             /*C*/
    set2.remove(5);                                          /*D*/
    String result = "";
    Iterator<Integer> myIterator = set2.iterator();
    while (myIterator.hasNext())
      result += myIterator.next();
    System.out.println("result: " + result);                 /*E*/
  }
}
```

Which one of the following statements about this program is TRUE?

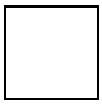
(A) Line A will fail to compile because `myArray` is an array with elements of type `Integer`, whereas the literal array `{3, 1, 2}` has elements of type `int`.

(B) Line B will cause a run-time error because `set1` is empty.

(C) Line C will add an `Integer` with value 4 to `set1`.

(D) Line D will cause a run-time error because `set2` does not contain an `Integer` with value 5.

(E) Line E will produce '`result:  1234`' on the standard output.



5. Which one of the following statements about the Collections Framework is TRUE?

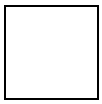
   (A) It is usually best to use `TreeSet<V>` rather than `HashSet<V>` as the default implementation for `Map<K,V>`.

   (B) All elements to be added to Collections must belong to classes that implement the `Comparable` interface.

   (C) `TreeMap<T>` should be used instead of `TreeSet<T>` as the implementation of choice for the `Collections<T>` interface.

   (D) An advantage of the Collections Framework is that, in user applications, there is no need to implement *any* methods necessary to sort a group of user-defined objects, since these methods are all provided in the standard collections implementations.

   (E) The Collections Framework provides a uniform way of manipulating groups of objects.



6. Consider the following program fragment:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.PrintWriter;
import java.io.FileWriter;
import java.io.IOException;

public class Test
{
  public static void main(String [] args) throws IOException
  {
    //    ...
    BufferedReader input =
        new BufferedReader(new FileReader(args[0]));
    PrintWriter output =
        new PrintWriter(new FileWriter(args[1]));
    //    ...
    output.println("The first line of the file " + args[0] + " is:\n"
                  + input.readLine());
    input.close();
    output.close();
  }
}
```

Which one of the following statements is TRUE when the command line call

```
java Test firstname secondname
```

is executed?

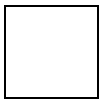
(A) If `firstname` is the name of a write protected file then an `IOException` is thrown when the program attempts to open the file for reading.

(B) If `firstname` is the name of a directory then `input.readLine()` returns a string corresponding to the first line of the directory's listing.

(C) If `secondname` is the name of a directory then a `FileNotFoundException` is thrown when the program attempts to print to the file.

(D) If `secondname` is the name of a directory then a `FileNotFoundException` is thrown when the program attempts to open the file for writing via the constructor `FileWriter`.

(E) The call of the `close` method on the `PrintWriter output` will produce a compiler error because `close()` is not defined in `PrinterWriter`.



7. Which one of the following statements about the `Comparable` interface is TRUE?

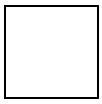
   (A) It implements a generic *polymorphic* sorting method which can be used by any interface that extends it.

   (B) If a user-defined class called `MyClass` implements `Comparable<MyClass>`, then `MyClass` must implement a method with specification '`public int compareTo(MyClass o)`'.

   (C) It defines the total ordering on elements in an implementing class.

   (D) Any classes implementing `Comparable` will fail to compile unless they provide consistent implementations for methods '`public boolean equals(Object o)`' and '`public int hashCode()`'.

   (E) If a user-defined class called `MyClass` implements the `Comparable<MyClass>` interface, then `MyClass` must implement a method with specification '`public int compareTo(Object o)`'.



8. Which one of the following names found in the Java Standard API is an interface?

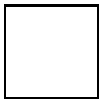
   (A) `actionPerformed`

   (B) `TreeMap`

   (C) `Throwable`

   (D) `InputStreamReader`

   (E) `Set`



9. In the COMP16212 lectures, you saw a multiple recursive implementation of a dice combinations method. Which of the following statements is TRUE?

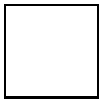
   (A) Unlike the vowel movement method seen in the COMP16212 lectures, the dice combinations method does not strictly use multiple recursion, because there is actually only one recursive call in the code.

   (B) Whilst the multiple recursive implementation is more flexible than the fixed loop version found in COMP16121, it was obvious when you saw it run that the recursive implementation is not significantly less efficient than the fixed nested loop version.

   (C) The approach taken in the multiple recursive implementation is essentially the same as the fixed nested loop version found in COMP16121, except that the actual number of nested loops executed is determined at run time, and the Java virtual machine rewrites the method to have that many loops in it just before executing it.

   (D) Whilst the multiple recursive implementation is more flexible than the fixed nested loop version found in COMP16121, it was obvious when you saw it run that the recursive implementation is by far the slower of the two.

   (E) It is just as easy to write an iterative version of a dice combinations method which has the same flexibility as the multiple recursive implementation, and the iterative version would be much faster.



10. Consider the following class definitions:

```
public interface A
{
    void method1();
}

public class B implements A
{
    public void method1() { }
    public void method2() { }
}

public class C<T extends A>   /* (a) */
{
    public T elt;             /* (b) */

    public C(T requiredElt)   /* (c) */
    {
      elt = requiredElt;
    }

    public void apply()
    {
       elt.method1();         /* (d) */
       elt.method2();         /* (e) */
    }

    public static void main(String [] args)
    {
       C<B> val = new C<B>(new B());
       val.apply();
    }
}
```

Which one of the following statements is TRUE?

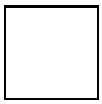
(A) Line (a) in class `C` fails to compile and should be replaced with `public class C<T>`.

(B) Line (b) fails to compile because it uses type parameter `T` rather than type `A`.

(C) Line (c) fails to compile because it uses type parameter `T` rather than type `A`.

(D) Line (d) in class `C` fails to compile because `T` is a generic type, so the compiler cannot determine `method1()` is defined for objects of type `T`.

(E) Line (e) in class `C` fails to compile because `T` is a generic type, so the compiler cannot determine `method2()` is defined for objects of type `T`.



11. Here is a method you have seen in lectures, as part of the OBT implementation.

```
private void buildFromList(OBTInt tree, int size)
{
  if (size > 0)
  {
    tree.setValue(-1); // First set to an arbitrary value.
    int leftSize = (size - 1) / 2;
    buildFromList(tree.left, leftSize);
    tree.value
      = ((Integer) buildElementsList
                     .get(buildElementsIndex))
            .intValue();
    buildElementsIndex++;
    int rightSize = size - 1 - leftSize;
    buildFromList(tree.right, rightSize);
  } // if
} // buildFromList
```

How many times is `buildFromList` called if `size` has the value $n-1$, where n is a whole power of 2, greater than 0? (You should assume that `buildElementsList` contains enough data from position `buildElementsIndex` onwards.)

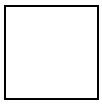
(A) $n+1$

(B) n

(C) $2n-1$

(D) $n-1$

(E) $log_2 n$



12. Here is a (possibly buggy) implementation of the vowel movement method you saw in the lectures.

```
private static void outputVowelMovements(int scanPosition)
{
  if (scanPosition >= inputStringBuffer.length())
    System.out.println(inputStringBuffer);
  else if (inputStringBuffer.charAt(scanPosition) != '*')
    outputVowelMovements(scanPosition + 1);
  else
  {
    inputStringBuffer.setCharAt(scanPosition, 'a');
    outputVowelMovements(scanPosition + 1);
    inputStringBuffer.setCharAt(scanPosition, 'e');
    outputVowelMovements(scanPosition + 1);
    inputStringBuffer.setCharAt(scanPosition, 'i');
    outputVowelMovements(scanPosition + 1);
    inputStringBuffer.setCharAt(scanPosition, 'o');
    outputVowelMovements(scanPosition + 1);
    inputStringBuffer.setCharAt(scanPosition, 'u');
    outputVowelMovements(scanPosition + 1);
  } // if
} // outputVowelMovements
```

If `inputStringBuffer` is a `StringBuffer` object containing the string `"**"`, how many output lines will be produced when the following code is executed?

```
outputVowelMovements(0);
```

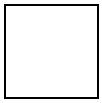
(A) 25 – it works perfectly and prints out all combinations of two vowels.

(B) None – because the method is a class method, and only instance methods can be recursive.

(C) 9 – it has a bug in it which makes it miss out one or more of the combinations.

(D) 5 – it has a bug in it which makes it miss out one or more of the combinations.

(E) 10 – it has a bug in it, so it produces 2 times 5 outputs rather than 5 to the power 2 outputs.



13. Given the following class and interface definitions

```
public interface InterW
{
}

public interface InterX<T>
{
}

public class ClassY
{
}

public class ClassZ
{
}
```

which one of the following would compile successfully without error

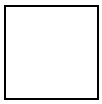
(A)
```
public class ClassA extends ClassY, ClassZ
                    implements InterW, InterX
{
}
```

(B)
```
public class ClassB<T extends InterX<ClassY>>
  extends ClassZ
  implements InterW
{
}
```

(C)
```
public interface InterC extends ClassY
                        implements InterW
{
}
```

(D)
```
public interface InterD implements InterW, InterX
{
}
```

(E)
```
public class ClassE<T implements InterW> extends ClassZ
{
}
```



14. Given the following class definitions

```
public class ClassP
{
  public String printInfo()
  { return "This is ClassP;";  }

  public String printAll()
  { return "In ClassP: " + printInfo(); }
}

public class ClassQ extends ClassP
{
  public String printInfo()
  { return "This is ClassQ;"; }

  public String printAll()
  { return "In ClassQ: " + printInfo() + " and: " + super.printAll(); }
}

public class TestPQ
{
  public static void main(String [] args)
  {
    ClassP myP = new ClassP();
    ClassP myQ = new ClassQ();

    System.out.println("myP: " + myP.printAll());
    System.out.println("myQ: " + myQ.printAll());
  }
}
```

what will be displayed on standard output as a result of running `TestPQ`?

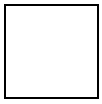
(A)
```
myP: In ClassP: This is ClassP;
    myP: In ClassP: This is ClassP;
```

(B)
```
myP: In ClassP: This is ClassP;
    myP: In ClassP: This is ClassP; and: In ClassP: This is ClassP;
```

(C)
```
myP: In ClassP: This is ClassP;
    myQ: In ClassQ: This is ClassQ; and: In ClassP: This is ClassP;
```

(D)
```
myP: In ClassP: This is ClassP;
    myQ: In ClassQ: This is ClassQ; and: In ClassP: This is ClassQ;
```

(E)
```
myP: In ClassP: This is ClassP;
    myQ: In ClassQ: This is ClassQ;
```



COMP16212

Mock Exam Questions: ANSWERS

WARNING

DO NOT EVEN THINK ABOUT LOOKING at ANY of these answers, until you have completed the whole of the Mock Exam under exam conditions!!!!

Clearly, to do so would seriously reduce the benefits you would get from attempting the Mock.

1. Answer: E	5. Answer: E	9. Answer: B	13. Answer: B
2. Answer: A	6. Answer: D	10. Answer: E	14. Answer: D
3. Answer: C	7. Answer: B	11. Answer: C	
4. Answer: C	8. Answer: E	12. Answer: C	