INSTRUCTIONS TO CANDIDATES

NAME OF CANDIDATE .......................... SEAT NO ......................

USUAL SIGNATURE ........................................

READ THE FOLLOWING CAREFULLY:

1. Each of the following questions comprise 5 statements, for which you should select the one most appropriate answer.

2. On the question paper, place a tick in the box to indicate your answer.

3. Enter your name and examination number IN PENCIL on the computer answer sheet according to the instruction on that sheet. Please note that all numbers MUST consist of three digits, e.g. 9 is entered as 009 and 25 as 025. The digits should be entered in the boxes under ‘Candidate Number’ and entered by means of horizontal lines in the appropriate boxes underneath, exactly as when answering questions.

4. When you have completed this question paper, read the instructions on the computer answer sheet carefully and transfer your answers. Use a HB pencil to mark the computer answer sheet and if you change your mind be sure to erase the mark you have made. You may then mark the alternative answer or mark the answer as don’t know.

5. At the end of the examination, be absolutely sure to hand in BOTH the answer sheet bearing the questions AND the computer answer sheet.

THIS PAPER MUST NOT BE REMOVED FROM THE EXAMINATION ROOM
Questions 1 and 2 refer to the following Java declarations.

```java
class APrinter
{
    APrinter() { System.out.print("A"); }
    APrinter(int i) { System.out.print("A"+i); }
}

class BPrinter extends APrinter
{
    BPrinter() { System.out.print("B"); }
    BPrinter(int i)
    {
        super(i);
        System.out.print("B");
    }
}

class CPrinter extends BPrinter
{
    CPrinter(int i) { System.out.print("C"+i); }
}
```

1. Consider the Java declarations above. What would you expect to see on standard output when an instance of BPrinter is created with `new BPrinter(1)`?

   A. AA1B  
   B. A1B  
   C. AB1B  
   D. B1B  
   E. BB

2. Look again at the Java code above. What would you expect to see on standard output when an instance of CPrinter is created with `new CPrinter(2)`?

   A. C2  
   B. BC2  
   C. B2C2  
   D. ABC2  
   E. A2BC2
3. Suppose the following two classes are declared in separate files in the same directory:

```java
class Point
{
    int xCoord, yCoord;

    Point(int x, int y) { xCoord = x; yCoord = y; }

    private void scale(int factor)
    {
        xCoord *= factor; yCoord *= factor;
    }
}

class ThreeDPoint extends Point
{
    int zCoord;

    ThreeDPoint(int x, int y, int z)
    {
        xCoord = x; yCoord = y; zCoord = z;
    }

    private void scale(int factor)
    {
        super.scale(factor); zCoord *= factor;
    }
}
```

Possible causes for compile-time errors might be:

1. the class `Point` is not public, and therefore cannot be subclassed by `ThreeDPoint`;
2. the fields `xCoord` and `yCoord` are not protected, and therefore not in scope in class `ThreeDPoint`;
3. class `Point` does not have a constructor with no arguments;
4. method `scale(int)` is private, and not in scope in class `ThreeDPoint`;
5. the keyword `super` can only be used to refer to constructors.

Which of these are real causes of compile-time errors?

A. 1 and 5 only.  [ ]
B. 2 and 4 only.  [ ]
C. 3 and 4 only.  [ ]
D. 2, 3 and 4 only.  [ ]
E. 3, 4 and 5 only  [ ]
4. Which of the following best describes the meaning of ‘encapsulation’?

A. ‘Encapsulation’ means that every field in a class has public access and modifier methods. ☐
B. ‘Encapsulation’ means that a class is part of a package. ☐
C. ‘Encapsulation’ means that all fields in a class are private. ☐
D. ‘Encapsulation’ means that a class implements an abstract data type and hides implementation details. ☐
E. ‘Encapsulation’ means that the local state of an instance has been saved in encoded form. ☐

5. For a given class and abstract data type, which of the following best describes the meaning of ‘adequacy of representation’?

A. Every value of the abstract data type is represented by an instance of the class, and every instance of the class represents a data value. ☒
B. Every operation in the abstract data type is implemented by a method in the class. ☐
C. Each public method in the class implements one of the operations in the abstract data type. ☐
D. The class has no public constructor, but provides public methods that create instances of the class. ☐
E. The class has a field that stores values of the abstract data type. ☐

6. The following fragment of Java code will cause a compile-time error.

```java
JTextArea ta = new JTextArea(20,20);
JButton clear = new JButton("Clear");
clear.addActionListener(
    new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            ta.setText(" ");
        }
    });
```

What is the cause of the compile-time error?

A. ActionListener is an interface and has no constructor. ☒
B. ActionListener should have only final methods. ☐
C. ta should be declared static. ☐
D. clear should be declared static. ☐
E. ta should be declared final. ☒
7. Consider the following code:

```java
interface Number
{
    public int getValue();
}

class Test
{
    class Zero implements Number
    {
        public int getValue()
        {
            return 0;
        }
    }

    public static final Number zero = new Zero();
}
```

This will raise a compile-time error; why?

A. Number is not public and therefore not in scope within Test.  
B. There is no body for the method getValue() in the interface Number.  
C. There should be a semicolon at the end of the declaration of Zero.  
D. The field zero is declared static, so the class Zero should be declared static.  
E. The field zero is declared final, so the class Zero should be declared final.

8. Consider the following fragment of Java code.

```java
class Outer{
    private class Inner1{
        int a;
        protected int b;
        private int c;
    }
    private class Inner2{ ...}
}
```

Which of the names Inner1, a, b and c are in scope within class Inner2?

A. All of them.  
B. Inner1, a and b only.  
C. Inner1 and a only.  
D. Inner1 only.  
E. None of them.
9. Consider the following declarations.

```java
static void printArray(int[] a, int i)
{
    System.out.println(a[i]); // print the ith element
    printArray(a, i+1);        // repeat for i+1
}

public static void main(String[] args)
{
    int[] is = new int[]{1,2,3};
    printArray(is, 0);
}
```

When the `main()` method is run, an exception is thrown; how many method calls are in the stack trace?

A. One.  
B. Two.  
C. Three.  
D. Four.  
E. Five.

10. When an exception is thrown, which of the following, in general, is the best course of action?

A. Always print out the entire stack-trace, so that the user has as much information as possible about the error.  
B. Try to catch the exception as deep down in the stack trace (closest to the top level) as possible, because this is where the most informative error message can be formulated.  
C. Try to catch the exception as high up in the stack trace (closest to the error) as possible, because this is where most information about the cause of the error is available.  
D. Never try to catch the error, because the Java interpreter does that anyway.  
E. Never report an error to the user, because that will only diminish their faith in the program.

11. Which of the following statements about concurrency is false?

A. Interference may arise when two or more threads access a shared resource.  
B. Monitors ensure mutual exclusion of critical sections.  
C. The `Thread.wait()` method allows a thread to surrender a monitor key.  
D. An instance of a class with synchronized methods has one monitor key for each synchronized method.  
E. Synchronized methods may lead to deadlock.
12. Which of the following is a true statement about time-slicing in Java?

A. Time-slicing requires a multiple-processor architecture.  
B. Time-slicing can only be ensured if every thread calls the `sleep()` method.  
C. Time-slicing can only be ensured if every thread calls the `wait()` and `notify()` methods.  
D. In time-slicing, a thread is allowed to run for a period of time that is specified by the programmer.  
E. In time-slicing, a thread is allowed to run for a period of time that is determined by the interpreter.

13. Which of the following best describes the effect of the `Thread.sleep()` method in a multithreaded program?

A. Calling `sleep()` causes execution of the thread to pause until another thread calls `sleep()`.  
B. Calling `sleep()` causes execution of the thread to pause until another thread calls the `wake()` method.  
C. Calling `sleep()` causes execution of the thread to end.  
D. Calling `sleep()` causes execution of a thread to pause for a specified amount of time, after which the thread is ready to run.  
E. Calling `sleep()` causes execution of a thread to pause for a specified amount of time, after which execution of the thread immediately resumes.

14. A thread can be in any of the following states: Ready, Running, Waiting, Blocked, and Dead. Which of the following statements is false?

A. The `notify()` method may change a thread’s state from Waiting to Ready.  
B. A thread becomes Dead when its `run()` method terminates.  
C. The `Thread.start()` method puts a thread into the Running state.  
D. The `Thread.sleep()` method changes a thread’s state from Running to Blocked.  
E. Time-slicing can change a thread’s state from Running to Ready.

15. Why does Java no longer provide the `Thread.stop()` method?

A. Calling the method usually resulted in deadlock.  
B. The method was incompatible with the `synchronized` keyword.  
C. The method compromised security in Applets.  
D. To encourage programmers to write explicit routines to shut down threads.  
E. To encourage programmers to use the `synchronized` keyword to ensure that all threads ended at the same time.
16. How is execution of a thread begun?
A. By calling the Thread() constructor.  
B. By creating a subclass of Thread() and calling the constructor for that class.  
C. By calling the Thread.start() method.  
D. By calling the Thread.init() method.  
E. By calling the Thread.run() method.

17. In the context of multithreaded programs, what is the purpose of a ‘monitor’?
A. A monitor ensures that at most one thread is running a critical section at any time.  
B. A monitor implements time-slicing.  
C. A monitor is code that is invoked whenever deadlock occurs.  
D. A monitor ensures that all threads are allowed to run for the same amount of time.  
E. A monitor prevents threads calling methods in other threads.

18. Thread-safety can be ensured in Java by qualifying a method with the keyword synchronized. This ensures that synchronized method cannot be called concurrently in different threads. Notionally, the interpreter associates a ‘key’ with the synchronized method, and any object that calls the synchronized method must first obtain the key. Concerning the number of keys, which of the following is true?
A. Each instance of the class has one key for each synchronized method in the class.  
B. Each instance of the class has one key, regardless of the number of synchronized methods in the class.  
C. There is one key for each synchronized method in the class, and these keys are shared by all instances of the class.  
D. There is one key, shared by all instances of the class, regardless of the number of instances of the class.  
E. Each instance of each class that can call the synchronized method has its own key.

19. Which of the following best describes the effect of calling the wait() method inside a synchronized method?
A. When wait() is invoked in a thread, that thread is put in the pool of waiting threads, keeping the monitor key for the synchronized method.  
B. When wait() is invoked in a thread, that thread is put in the pool of waiting threads, and releases the monitor key for the synchronized method; when the thread is next invoked, any other thread that has the monitor key must release it and give it back to the thread.  
C. When wait() is invoked in a thread, that thread is put in the pool of waiting threads, and releases the monitor key for the synchronized method; when the thread is next invoked, it must reacquire the monitor key before it can proceed.  
D. When wait() is invoked in a thread, that thread is given highest priority, and all other threads must wait for that thread to terminate.  
E. When wait() is invoked in a thread, all other threads must wait until execution of the synchronized method terminates.
20. Which of the following best describes ‘deadlock’?
   A. Deadlock arises when two or more threads concurrently access a shared resource.
   B. Deadlock arises when two or more threads are given higher priority, and so prevent other threads from being executed.
   C. Deadlock arises when two or more threads each control some resource, and so prevent the other threads from making progress.
   D. Deadlock arises when a class contains a synchronized method, and so a lock is put upon the code in that method.
   E. Deadlock arises when a thread’s `sleep()` method is invoked while that thread is in possession of a monitor key.

21. Which of the following is not characteristic of direct manipulation?
   A. Objects in the application domain are represented visually.
   B. Results of actions are immediately visible.
   C. Actions are rapid and reversible.
   D. Interaction takes place through pointing and selecting.
   E. Actions are controlled by a command language.

22. Consider the following features of user interfaces:
   1. use of a command language
   2. interaction through form fill-in
   3. support for macro definition
   4. informative feedback provided on each task accomplished.
   Which of these are particularly suited to expert users?
   A. 1 only.
   B. 1 and 3 only.
   C. 2 and 3 only.
   D. 1, 3 and 4 only.
   E. 2, 3 and 4 only.

23. In the Model-View-Controller architecture, what does ‘model’ refer to?
   A. The user’s semantic model of the system
   B. The programmer’s conceptual model of the system
   C. The data that is manipulated by the system
   D. The data that is output by the system
   E. The best possible graphical representation of the system
24. Which of the following best describes how AWT events are handled in Java?

A. There is a special thread that handles all AWT events.  
B. Each AWT event causes a new thread to be created to handle that event.  
C. All AWT events are handled in the main thread.  
D. The programmer must explicitly create one thread to handle all AWT events.  
E. For each AWT event, the programmer must explicitly create a thread to handle that event.

25. Which of the following is the most accurate statement about thread-saftey for graphical user-interfaces in Java?

A. An interface is guaranteed to be thread-safe if it uses only javax.swing components.  
B. An interface is guaranteed to be thread-safe if it uses only java.awt components.  
C. An interface is guaranteed to be thread-safe if it uses only the SwingEvent thread.  
D. An interface is guaranteed to be thread-safe if it uses only the EventDispatch thread.  
E. An interface is guaranteed to be thread-safe if a separate thread is created for each event.

26. An instance of which event class is created by a user entering a return character in a TextField component?

A. TextFieldEvent  
B. TextComponentEvent  
C. TextChangedEvent  
D. ActionEvent  
E. ReturnEvent

27. What is the immediate superclass of javax.swing.JComponent?

A. java.lang.Object  
B. java.AWT.Component  
C. java.AWT.Container  
D. java.AWT.Window  
E. javax.swing.Object

28. What is the best way of setting up the graphical user-interface for an applet?

A. Create and place all GUI components in the Applet constructor.  
B. Create and place all GUI components using the appropriate tags in the HTML document.  
C. Override the Panel constructor and invoke super().  
D. Override the Applet.init() method.  
E. Override the Applet.start() method.
29. Consider the following statements about modal dialogs in Java.

1. A modal dialog always has a Frame or another Dialog as owner.
2. A modal dialog blocks user interaction with its owner when it is displayed.
3. When a thread displays a modal dialog, execution of that thread is blocked until the dialog is closed.

Which of these statements are true?

A. 2 only. □
B. 2 and 3 only. □
C. 1 and 3 only. □
D. 1 and 2 only. □
E. All of them. □

30. Consider the following statements about applets.

1. An applet can open network connections to the HTTP server from which it was downloaded.
2. An applet can read files from the local filestore.
3. An applet can redirect the browser in which it is embedded to any URL.
4. An applet can execute processes in the native operating system.

Which of these statements are true for an applet downloaded from a remote HTTP server?

A. 1 only. □
B. 2 only. □
C. 1 and 3 only. □
D. 2 and 4 only. □
E. 3 and 4 only. □

31. Consider the following Java command:

```java
BufferedReader br =
    new BufferedReader(
        new InputStreamReader(System.in)
    );
```

Which of the following statements is false?

A. The command allows lines of text to be read from standard input. □
B. The command allows keyboard input to be read efficiently by storing characters in a buffer. □
C. The command allows keyboard input to be written to a file. □
D. The command converts the byte-oriented standard input stream to a character stream. □
E. The command is an example of the use of wrapper classes. □
32. Which of the following statements best describes the difference between a byte stream and a character stream.

A. A byte stream contains only bytes, while a character stream contains only characters.
B. A byte stream allows only single bytes to be written and read, while a character stream stores bytes in a buffer until there are enough to make up a character.
C. A byte stream is an input stream, while a character stream is an output stream.
D. A byte stream is used to read and write to memory, while a character stream is used to read keyboard input.
E. A byte stream allows only single bytes to be written and read, while a character stream allows characters, integers and strings to be written and read.

33. Which of the following describes the effect of the flush() method in the java.io.BufferedOutputStream class?

A. The method is abstract and has no effect.
B. The method blocks any attempt to read from the stream until the buffer is full.
C. The method forces any bytes in the buffer to be written.
D. The method empties the buffer, losing any bytes that were in it.
E. The method closes the output stream.

34. Consider the following fragment of Java code:

```java
try {
    FileInputStream fis = ...;
    FileOutputStream fos = ...;
    ...
} catch (IOException ioe){}
finally {
    fis.close();
    fos.close();
}
```

Which of the following statements describes the effect of the finally-block?

A. The code will close the file I/O streams, whether or not an exception is thrown.
B. The code will close the file I/O streams only if an exception is thrown.
C. The code will close the file I/O streams only if an exception is thrown that is not of type IOException.
D. The code will close the file I/O streams only if no exceptions are thrown.
E. The code will not compile because finally is not a keyword in Java.
35. Consider the following statements about the classes **Socket** and **ServerSocket** in the package **java.net**.

1. **ServerSocket** creates **Socket** instances.
2. A **Socket** instance provides byte-oriented input and output streams.
3. A **ServerSocket** will accept connections only from remote Java Virtual machines.
4. **ServerSocket.accept()** blocks a thread until a connection is made.

Which of these statements are true?

A. 1, 3 and 4 only.
B. 1, 2 and 4 only.
C. 2, 3 and 4 only.
D. 1 and 3 only.
E. 2 and 4 only.

36. In Remote method Invocation, what is meant by ‘marshalling’?

A. Marshalling is the process of synchronizing all parameters to the method being invoked, in order to prevent data corruption arising from the concurrent method call.
B. Marshalling is the process of sending a copy of the remote object to the client JVM in order to invoke its method locally.
C. Marshalling is the process of sending the state of the client JVM on a byte stream to the remote object in order that the JVM can be started as a separate thread on the remote host.
D. Marshalling is the process of suspending the current thread in the client JVM until a result is obtained from the remote method invocation.
E. Marshalling is the process of sending the state of all parameter instances on a byte stream to the remote object.

37. Which of the following best describes the role of stub objects in Remote Method Invocation?

A. A stub object is a local object that implements the same interface as the remote object; when one of its methods is called, it sends the parameters to the remote object, and receives the result of the corresponding method call in the remote object.
B. A stub object is a local object that implements the same interface as the remote object; when one of its methods is called, it obtains the source code of the remote object, and compiles and runs the corresponding method.
C. A stub object is a local copy of the state of the remote object that is sent across the network when one of the remote object’s methods is called.
D. A stub object is the socket that is created to the remote machine on which the remote object is located.
E. A stub object is the encoded result of the remote method invocation that is sent across the network.
38. Consider the following fragment of Java code.

```java
interface Comparable<A> {
    public A greater(A x, A y);
}

class Comparison {
    public <A, B extends A, C extends Comparable<A>>
    A test(A x, B y, C z) {
        return z.greater(x, y);
    }
}
```

Which of the following statements is true?

A. The code will cause a compile-time error because A cannot be used as a return type.
B. The code will cause a compile-time error because the parameter type B cannot extend the parameter type A.
C. The code will cause a compile-time error because greater requires its arguments to be of the same type.
D. The third argument to the method test() must be an instance of some class that implements the Comparable interface, where Aaa is the type of the first argument of test().
E. The third argument to the method test() must be an instance of a class that explicitly extends a class that implements the Comparable interface by adding a method that takes two arguments of different types.

39. Java implements generic types by erasure: what does this mean?

A. The compiler removes references to parameter types and inserts casts wherever return types refer to parameter types.
B. The programmer must comment out references to parameter types before code is compiled.
C. The compiler removes any casts that may be type-unsafe.
D. The compiler removes any references to parameter types and inserts references to the closest superclass that will make expressions type-safe.
E. Any type-unsafe casts to parameter types will be ignored by the interpreter at run-time.
40. Consider the following class declaration.

```java
class Adder<A, B extends List<A>>
{
    Adder(A elt, B list) { ... }
}
```

Given an instance \(e\) of class `Integer` and \(l\) of class `List<Integer>`, which of the following is a correct call of this class’s constructor?

A. `<Integer, List<Integer>>Adder(e,l)`
B. `<A,B extends List<A>>Adder(e,l)`
C. `<Integer>Adder(e,l)`
D. `Adder<Integer, List<Integer>>(e,l)`
E. `Adder<Integer, B extends List<Integer>>(e,l)`