Option B is the correct answer.
When we create multi-dimensional arrays, we need to give first dimension size. So, when you create a two-dimensional array, we must give the number of rows when we initialize it. So, here options A and E are incorrect. Option B is correct since it creates a two-dimensional array without passing the size for columns which is legal, then we can define a different number of columns for each row.

Option B is the correct answer.
Here we have created a two-dimensional array with three rows and each of row will have three columns, so nums array can hold total 9 values. But then we have changed the first row of the two-dimensional array to refer array which can hold only two values. So, now total elements allowed will be 8. Hence, option B is correct.

Option D is the correct answer.
Following is the correct syntax for creating the anonymous array.

type[][elements separated by comma.]

Remember that you do not specify a size when using anonymous array creation syntax. The size is derived from the number of items (comma-separated) between the curly braces. So, option D is correct.
As explained above options A and C incorrect.
Option B is incorrect as [] is missing.

Option C is the correct answer.
At line 4, we have defined an integer array with final, but it doesn't mean that all values in the array are final. It means only the reference is final, so at line 6, there won't be any error. All array elements are initialized to their default values when array initialized, hence all elements of the array will be 0, at line 6 second element will be increased by one. So, when using enhanced for loop, 010 will be printed. Hence, option C is correct.

Option E is the correct answer.
This code fails to compile due to an error at line 7, because the array constants can only be used in initializers. So, we can't use "{1,2,3}" at line 7.

Option D is the correct answer.
Methods in the Arrays class throw a NullPointerException, if the specified array reference is null. So, here we haven't initialized the ints[1], so it is null. Hence, passing it to Arrays sort method throws a NullPointerException. So, option D is correct.

Option A is the correct answer.
Arrays class has binary Search method which can use to search elements in a sorted array.

public static int binarySearch(int[] a,int key)

This method searches the specified array of ints for the specified value using the binary search algorithm. The array must be sorted (as by the sort(int[]) method) prior to making this call. If it is not sorted, the results are undefined.

Since we have sorted the array, binary search will function properly. Since the array is sorted, index position of -1 will be 0, so the binary search will return 0. Hence, option A is correct.

Option D is the correct answer.
The method meth() is declared as an exception, so when we call that method within another method, the calling method must handle or declare the exception. Otherwise, compilation will not succeed.

Option D is the correct answer.
If we write catch block for checked exceptions and if there is no chance of that checked exception from the try block , we get compile time error. But this rule is not applicable for Exception class even though it is checked exception . There will be no exception occurred in try block , so catch clause won't run but finally runs as always. Hence, finally will be printed; so option D is correct

Option D is the correct answer.
Option D is correct since the NegativeArraySizeException is thrown when we tried to initialize an array with negative size.
The NullPointerException is thrown by the JVM when there is a null reference where an object is required. So, option A is incorrect.
The NumberFormatException thrown by the programmer when an attempt is made to convert a string to a numeric type but the string doesn't have an appropriate format. So option B is incorrect.
Option C is incorrect since the IllegalArgumentException thrown by the programmer to indicate that a method has been passed an illegal or inappropriate argument.

Option E is the correct answer.
You can't have multiple finally clauses. In this code, the first finally clause causes the end of the try clause. So, other catch clause appeared like a catch clause without a try clause, so compilation fails.

Option C is the correct answer.
Here, we try to cast a superclass reference to lower class reference, but superclass reference refers to the superclass object. So, casting will cause a ClassCastException.

Option B is the correct answer.
When we compile the java source we get the bytecode. Java bytecode is the instruction set of the Java virtual machine. Each bytecode is composed by one, or in some cases two, bytes that represent the instruction (opcode), along with zero or more bytes for passing parameters. So, option B is correct.

Option A is the correct answer.
The compiler is invoked by the javac command. When compiling a java class, you must include the filename with .java extension. Hence, the option A is correct.
To execute java program we can use java command but can't use it for compiling.

Option B is the correct answer.
In the printing statement at line 5, we have used the string value first, so the + will work as the concatenation operator in "x+y" hence 1012 will be printed. So, option B is correct.

Option C is the correct answer.
The correct order of the statements should be

package statement
import statement

class statement

Except above comments can be appeared on any place. So option C is correct.

Option D is incorrect since we can place constructors before or after other statements, also middle of other statements.

Option C is the correct answer.
We can have following comments in java
/* text */ - The compiler ignores everything from /* to */.
// text - The compiler ignores everything from // to the end of the line.
/** documentation */
This is a documentation comment and in general, it's called doc comment. The JDK javadoc tool uses doc comments when preparing automatically generated documentation.
Option C is correct since it doesn't affect the complete statement. Option A is incorrect since using / for comments is invalid in java. Option B is incorrect since because of // the statement will be incomplete.

Option B is the correct answer.
In the given code, we have defined a static variable x with value 2 hence at line 5 if block will be executed, and increment the value of x by one, at line 7, defining new integer with the same name will shadow the class variable x, but the scope of that variable is limited to the if block. Hence, the value of the class variable will be printed which is 3. So, option B is correct.
The integer variable x at line 10 is declared after the printing statement, so it won't affect any statement above the line 10, but if there was more code after line 10, then that x would shadow the class variable x.

Option C is the correct answer.
At line 10, we have defined new integer with value 3, it shadows the class variable x. Hence, the value of z will be 3.
So, 2+3 will print 5 as the output. Hence, option C is correct.

Option C is the correct answer.
To store the number of books in the library, you can't use an instance variable. Because Library class uses a static method to add a book in the library. The static method can't access the instance variable. So, you must take static variable to store the number of books in the library. The static variable in Library class will be incremented when static method of Library class gets called. when you create an object for Book class, Book class constructor calls Library class static method which adds a book to the library and updates the number of the books. To know the number of books in library, just call static variable of Library class which gives the number of books. So, option C is correct.

Option E is the correct answer.
Correct syntax for static import is
import static [class member/s]
Option A is incorrect since it is non-static import as we haven't used static there.
Option B is incorrect since to import a specific static method it is illegal to use parentheses, a just name should be used.
Options C and D are incorrect since we have used static before import there, which makes them invalid.
Option E is correct since we have followed correct syntax there.

Option E is the correct answer.
When the value of i is 2, the "if statement" will call continue. After that, every time the value of i will be 2 as it can't reach to i-- statement, where it cause a never ending loop.

Option B is the correct answer.
This is a nested loop. When j is equal to two, current iteration stops as if calls continue. So, there will be 123 on first and third iteration process.

Option B is the correct answer.
Option B is correct as it will print all the elements of the array. As every array has three elements we should break the inner loop as soon as k becomes 3, as then it will cause an exception.
Option A is incorrect as the output will only contain elements of the array[0] because it breaks the outer loop.
Options C and E are incorrect as they cause an ArrayIndexOutOfBoundsException.
Option D is incorrect as the output will only contain the first elements for each array.

Option E is the correct answer.
Option E is correct as "for" loops has two forms, for and enhanced for. The later one is designed to iterate through elements of a collection or array.
Option A is incorrect because to execute a certain section of code then we have to use "if" of "switch".
Option B is incorrect as we can't use loops to select different execution paths, for that we need to use control statements like if, switch.
Option C is incorrect as "while" and "do-while" are not same.
Option D is incorrect since the "while" loop evaluates its expression at the top of the loop.

Option C is the correct answer.
Using "while(true)" at line 6, will create an endless loop, but in its first iteration, all the elements of the array will be printed in descending order since the initial value of the variable "i" is 3.
Option C is correct as the output will be 4321 followed by endless loop because of "while(true)".
Options A and B are incorrect since a never ending loop is created.
Option D is incorrect as "4321" is produced in the first iteration of the first while loop.
Option E is incorrect as the code compiles successfully.

Option B is the correct answer.
At line 6, we have used parentheses to change the value of x. Inside the parentheses, we have used assignment operator which will result in x multiplied by 3. But the condition of the if block won't be true since the value of y, is 10. Hence, else block executes and print 3, so option B is correct.

Option A is the correct answer.
If we use a variable for the case, it needs to be a compile time constant, so options B and C are incorrect.
Option D is incorrect as there is a case with value 2, so if we use 1 as the value of x then there will be two cases with value 2, which is illegal.
Option A is correct since there if we use -1 then the case x+1 will be 0, so there won't be any duplicate cases.

Option E is the correct answer.
Option E is correct as the code fails to compile due to line case statements. When using a primitive or reference type variables as case constants, we should keep in mind that they should be compile-time constant. But in given code, we have used variables from the primitive array, even the array is final, the elements of the array are not final hence code fails to compile.

Option C is the correct answer.
While using if, we need to remember when there is no curly braces to group the if statements. The if consider only the next line as statement related to the if. So, in given code, first if consider next if condition as its statement, and that if at line 6 consider if at line 7 as its statement to execute.
At line 5, if test becomes true hence next if at line 6 is tested and it becomes false, so its else block at line 10 is executed and concat 3 to string. So, option C is correct.

Option E is the correct answer.
This code fails to compile since we can't include other statements between if and else. So, line 9 printing statement included between if and else results in a compile time error.

Option C is the correct answer.
This code compiles successfully and prints C as the output so option C is correct. But if you have a look carefully you can see that logic of using if else here is incorrect since any value greater than 40 will end up printing C which it is not supposed to. To fix this we need to use check for greater values first.

Option A is the correct answer.
The correct syntax of using the ternary operator is

[condition]?[statement to execute when true]:[statement to execute when false]

So, option A is correct since it follows the correct syntax.
Options B and D are incorrect since we have used incorrect syntax for the ternary operator.
Option C is incorrect since we are not allowed to use if inside a printing statement.

Option C is the correct answer.
In given code, we need to use parentheses to change the operator's precedence.
Option A is incorrect since (z/y) will result in 1 and then z*2 will result in 6, so 7 will be printed.
Option B is incorrect since (y+z) will happen first and then it will divide z which returns 0. So, the final output will be 0.
Option C is correct since it will print 8. In statement(z / y + z)*2, (z / y + z) will result 4, (1+3) and then it will be multiplied by 2 which result 8.

Option D is the correct answer.
While using equals method of wrappers, it checks the wrapper type and the wrapper primitive value. Hence, line 8 will result in false since i and d are not the same types. Comparison of primitives using '==' will check the actual value of the variable refer not the type. Hence, at line 9, true will be printed. Hence, option D is correct.

Option D is the correct answer.
We have passed the anonymous array to the iterator method which uses a for loop to iterate through array elements and print them. In given for loop, we have used the printing statement in update part and we have done the increment part inside the loop. So, in the first iteration, it will increase the value of x and then print the second element. But when it does two iteration value of x will become 3, so in final iteration trying to access element will index position 3, will result in an exception. Hence, option D is correct.

Option D is the correct answer.
The first one is the correct way of using for loop to iterate the array elements and the second one is called an enhance for loop. It can also be used to iterate the array elements. The third one is the incorrect use of the enhance for loop since the variable declaration should come before the collection inside for.

Option D is the correct answer.
When creating Strings without using "new" keyword, compiler search for equal String literal in the String pool, if equal found then the reference will refer to it without creating new String. So, in this case with the statement I, new String won't be created, so both "s1" and "s2" refer to the same String object. Therefore with the statement I, we will have "true true" as result. So, options B and C are incorrect.
When using "new" to create a String, a new String object will be created even there is an equal String literal in the pool, so here "==" check will return false since "s1" and "s2" refer to two different objects. But the "equals()" method will return true since the String literal are equal. So, the option A is incorrect and D is correct.

Option A is the correct answer.
Option A is correct since the switch is efficient than if, as it will directly execute the statements based on the input, where ifs have to check one by one till match found.
Option B is incorrect as in a switch all cases should be unique.
Option C is incorrect as the switch uses equals method when working with Strings.
Option D is incorrect since we can have nested switches.

Option E is the correct answer.
To override a method, that method needs to inherited. So, in this case, we have defined the method run as private hence we can't override the method run at line 8. Hence, option E is correct.

Option A is the correct answer.
Here at line 20, we have created Dog instance and have used a superclass Animal reference, so we can invoke any method of superclass Animal even the actual object is Dog. When invoking the overridden method at run time, the overridden version of the actual object will be invoked, so sound and run method of Dog class will be invoked. So, option A is correct.

Option E is the correct answer.
If a method is overridden but we use a polymorphic (super type) reference to refer to the subtype object with the overriding method, the compiler assumes we are calling the supertype version of the method. If the supertype version declares a checked exception but the overriding subtype method does not, the compiler still thinks you are calling a method that declares an exception.
So, above code fails to compile due to line 12 since compiler sees calling the eat method of superclass and fails to catch or declare the exception.

Option C is the correct answer.
Option A is incorrect since we can't use more restrictive access level when overriding methods so using private is incorrect.
Option B is incorrect since we can't change the return type of overriding method except for covariant types. (Subtype of superclass method return type)
Option C is correct since there we have used Integer as return type which is a subtype of Number, and also it is legal to throw a new or broader unchecked exception in overriding method. Hence, option C is correct.
While overriding, we can't change the argument list. Hence, option D is incorrect.

Option D is the correct answer.
Inheritance can also make application code more flexible to change because classes that inherit from a common superclass can be used interchangeably. If the return type of a method is superclass
Reusability -- facility to use public methods of base class without rewriting the same
Extensibility -- extending the base class logic as per business logic of the derived class
Overriding-- With inheritance, we will be able to override the methods of the base class so that meaningful implementation of the base class method can be designed in the derived class.

Option D is the correct answer.
At line 9, we have created a Student instance with reference Person, so it is impossible to invoke the read method using that reference directly even the actual object is Student compiler tries to find the read method of reference type. Hence, the option A is incorrect. Options B and C will do the same thing since the "." operator has higher precedence if we use them then it will try to see the return value of the read method and tries to cast it which is illegal.
Option D is correct since we have correctly cast the object to Student using correct order of parentheses.

Option B is the correct answer.
Option B is correct as we have used the keyword "this" to invoke the "Cat(String s)" constructor. And the print statement in that constructor produces the expected output.
Option A is incorrect as the superclass has only one constructor which can take String as the argument so calling "super()" with empty parameters causes a compile time error.
Option C is incorrect as the expected output can only be achieved by invoking the "Cat(String s)" constructor so using super will skip the needed constructor and will invoke the super class constructor. Therefore, the expected output won't produce.
Option D will cause compile time error as we need to have "Cat()" constructor which doesn't take any parameter as we used "new Cat()" at line 10.

Option D is the correct answer.
Option A is incorrect as only interfaces can have default method.
Option B is incorrect as abstract classes can't be final, they meant to be extended and concreted, so if we could mark them final then there will be no way to concrete them.
Option C is incorrect as every class abstract or not should have a constructor.
Option D is correct, it is illegal to instantiate abstract classes.

Option E is the correct answer.

From java se 8, we can have default and static non-abstract methods in interfaces but there are some rules, such a rule is we can't override Object class methods using default method, it will result in a compile time error. Hence at line 2 trying to override equals method of the Object class results into a compile time error.

Option A is the correct answer.
From java se 8, we can have default and static non-abstract methods in interfaces but there are some rules. Both static and default methods in interfaces are public. At line 15, we have initialized an instance of type I. So, using it we can invoke the default method, which prints "I". Then we can invoke the static method using the interface name or using the object reference, here we have to use Interface name to invoke its static method, which will print "Static". Hence, option A is correct.

Option A is the correct answer.
Using L suffix will make literal long, as long as it is valid non-fraction number hence option D is incorrect. Also, we can use decimal, hexadecimal, octal or binary literals, so option A is correct.
Option B is a valid integer literal but it can be assigned to long since long can store integers.

Option E is the correct answer.
Java handles deallocation of memory automatically, we do not need to explicitly clear an element. The GC only occurs during execution of the program. When no references to the object exist, that object is assumed to be no longer needed, and the memory occupied by the object can be reclaimed.

Option E is the correct answer.
We can use various literal types for numeric type variables, i.e.
binary literal – 0b11
Octal literal – 012
Decimal literal – 11
Hexadecimal literal – 0xff
At line 5, we have used octal literal, for an octal literal, we can use only 0-7 digits, so using 8 will result in compile time error. Hence option E is correct.

Option E is the correct answer.
At line 3, we have defined an integer array. Then at line 4, we have tried to assign array at line 3 to double type array so code fails to compile. So, option E is correct.

Option C is the correct answer.
At line 8, we have created a wrap instance, in that wrapper, there are two objects, one is an array of ints and other is a Double. So, making w reference null will result in Wrap object unreachable and also its enclosing two objects. Hence, total three objects eligible for GC. Hence, option C is correct.

Option B is correct.
We can invoke a static method using a class name or via an instance of the class. So, here option B is correct since we have used an object reference to invoke the p2 method by passing an integer as the argument.
Option A is incorrect since the p2 method expects an integer as the argument.
Option C is incorrect since print returns nothing so using it in printing statement is invalid.
The print method is an instance method so we can't invoke it using a class name. So, option D is incorrect.

Option B is the correct answer.
Option B is correct since it returns true. Comparison of Wrapper and primitive using "==" will unwrap the wrapper to primitive and compare, hence "d==l" return true.

Option A is incorrect since it results in a compile-time error because we can't use "==" with two different wrappers.

Other options are incorrect since equals method of the wrapper will first check if the wrapper is of the same type, hence others will return false.

Option B is the correct answer.
Integer wrapper as a static field called MAX_VALUE which represents the maximum value of Integer type can hold.
Hence, option B is correct.

Option D is the correct answer.
While assigning literals to wrapper we need to assign wrapper or primitive in wrapper type since assignment can't unwrap and then wide the literal. So, here at line 3, we have tried to assign an integer primitive literal to the Double wrapper which results in a compile time error. So, option D is correct.

Option B is the correct answer.
Primitive char type is int-compatible, so at line 4 using increment operator will change it is ASCII value by one, which will result in the next character. Hence, D will be printed, so option B is correct.

Option D is the correct answer.
This code shows usage of BYTES, SIZE are the constant fields in Double class.

BYTES : The number of bytes used to represent a double value which is 8 .
SIZE : The number of bits used to represent a double value.

Option E is the correct answer.
When compiling the code, Compiler looks only at the reference and sees that A doesn't have a print() method that takes a String. The compiler doesn't care that the actual object might be a B at runtime. So, it will result in a compile-time error, so option E is correct.

Option C is the correct answer.
According to the given requirement, the method should be a default method since the method supposed to be defined in an interface and it should be a non-abstract and an instance method. Since we used the method in an interface it is implicitly public, so option C is correct.

Option A is the correct answer.

The signature of a method is used to uniquely identify a method. A signature consists of:

- Method name
- Number of parameters
- Type of the parameters
- Order of the parameter

Even a return type is not part of the method signature. The return type is a must for a method, if a method doesn't expect to return anything then we need to use void. So, option A is correct.

Option D is the correct answer.
We can't access non-static content from static content, it will result in a compile-time error. So, at line 6, trying to access variable x in main method results compile-time error, since the variable x is not static.

Option D is the correct answer.
Class A in package one has default access level so when the code tried to extends class A and to create an instance of class A in class B of package two, it cause compilation failure since class A can be only accessed within the one package. So option D is correct.
Options A is incorrect as the code fails to compile, if we mark class A as public then class A can be accessed within package two, so then the code will compile fine.

Option A is the correct answer.
The statement I is correct as the private members can be only accessed by the members of the enclosing class.
Statement III is incorrect . Let us consider Test1 is in package one, Test2 is in package two and Test3 is in package three. Test2 is a subclass of Test1 and Test3 is a subclass of Test2. Here both Test2, Test3 can access Test1 class protected members. Here statement III is not valid.
Statement II is incorrect since protected members are inherited by subclasses for same package as well as other packages.
Option A is correct since only the statement I is correct.

Option B is the correct answer.
When you are passing a primitive variable, you are passing a copy of the bits representing the variable. Here we have passed the variable x to change method which will pass the bit patter of 1, not the reference, so the value of x in the main method will remain unchanged. So, the output will be 3, hence option B is correct.

Option A is the correct answer.
At line 6, we have used one of the overloaded version of the append method.
public StringBuilder append(char[] str, int offset, int len)
Appends the string representation of a subarray of the char array argument to this sequence. Characters of the char array str, starting at index offset, are appended in order to the contents of this sequence. The length of this sequence increases by the value of len.

So at line 6, the content of the builder will be "1Z0-8". Then in next two lines, we have added 0 and 8, which makes it "1Z0-808". Hence, option A is correct.

Option C is the correct answer.
As the length of the String "WhizLabs" is eight, so the length method will return 8. And we have used StringBuilder class' "public StringBuilder(String str)" constructor, when we construct a string builder initialized to the contents of the specified string. The initial capacity of the string builder is 16 plus the length of the string argument. So, here we have initially passed "Whiz" to constructor and length of it; is 4. Therefore, the initial capacity is 20. So, 20 will be returned when capacity method invoked and output will be 28. Hence, option C is correct.

Option C is the correct answer.
Strings are immutable, so at line 4 concatenating "0" will not add "0" to s, instead it will return new String with value "1Z0". At line 5, using assignment operator will add "1" to s, so output will be 1Z1-808. Hence, option C is correct.