Bünyamin AKTAŞ
17/04/2020

HOMEWORK 1 REPORT

The correct answers are coloured with red.

Q1) **A -** main() method should be called without creating an object. In order to call a method without creating an object, it should be made static. A method can be made static by using *static* keyword. So Java gives an error if the main() method does not contain *static* keyword in its signature as in this example.

B – Java Virtual Machine (JVM) always looks for a main() method that does contain a String array parameter. Because this option does not contain any parameters, JVM cannot recognize the main() method and gives an error.

C – JVM looks for the *main()* signature because it is predefined that way. So the main method should contain *main()* in its signature. Other than that, main() method should be public because it should be able to get called by JVM which is outside of the scope of the package. The method has a parameter that is a String type array as it should be. The name of the array is not significant for the validity of main() method. In this option, JVM cannot find the main() method due to that the method is not public at the first place and gives an error.

D – The method is public, has *static* keyword and a parameter which is a String array. In addition, it has *final* keyword that prevents methods to be overridden. Overriding main() method is allowed in Java and sometimes it is necessary to prevent main() method to be overridden. For example if a class is extended, then main() method can be overridden in subclasses and this may not be desired. In conclusion, the method in this option can be compiled and run without an error.

Q2) This diagram depicts inheritance in programming languages. *Metal* class is a superclass and *Gold* and *Silver* classes are subclasses of *Metal.* A subclass inherits all attributes from its superclass and can override them.

A – The platform independence means that Java compiled bytecode can run on all operating systems independently from the operating system in which the Java code is compiled. So this diagram is not about it.

B – Object-oriented design is a system of separate parts of an entity running together. This diagram shows us 3 different classes that runs together to give a desired output.

C – The *Gold* and *Silver* classes are subclasses of *Metal* class, so that they inherit all attributes from their superclass.

D – *Gold* and *Silver* classes are not connected directly so they cannot use each other's attributes.

Q3 **A –** *.java* filename extension is used in Java source code files.

B – There is no such thing as *.bytecode* as a filename extension.

C – *.class* is the determined filename extension by designers of Java in order to be used in Java bytecode compiled files.

D – Files using *.dll* filename extension are libraries used in Microsoft Windows and OS/2 operating systems.

Q4 **A –** Both packages imported on lines 1 and 2 having same class necessitates some tinkering but it doesn't necessarily cause any error.

B – Class *Date* exists in both packages imported so the compiler cannot distinguish which package's class is meant to be used here.

C – Line 5 states exactly which class to be used, so it does not cause a problem.

D – Line 4 causes issue.

Q5 Objects are the instances of classes. Objects can hold data via their variables. Multiple objects can be created from a single class and each object is unique with its own data which means they are grouped with their data. Objects can perform actions with their methods and can be cast to their own superclasses thank to inheritance. So the answer is **A.**

Q6 **A – Interface variables** are created in Java interfaces. By default these are public, static and final. So that they are available at all places and their value cannot be changed or one more instance cannot be created.

B – Class variables are static variables and accessible from all classes in a package by default.

C – Instance variables are created in a class. If they are not created as public they normally cannot be accessed but non-static methods can provide an access to these variables from outside.

D – Local variables are created in a method and they cannot be accessed from outside of the method.

Q7 **B – java.lang** package is imported by default because without this package it is impossible to code in Java.

The other packages; java.util, system.lang, java.system is not mandatory for coding.

Q8 Double slash before a line or lines encapsulated by /* - */ characters are interpreted as comments in Java but # is not a comment indicator for Java compiler. The answer is **C.**

Q9 A .java file must contain one public class at most and can contain private classes, too. The answer is **D.**

Q10 As explained in **Q1**, a main() method must be public static and void and must have a String type array parameter. It should be static because it needs to be called without creating an object. So main() method in the example should be corrected.

Class variables are static variables so they are accessible from all places. There are 2 static/class variables.

Instance variables are variable created in class and they are available at all places in the class in which they are created. Here we have two instance variables.

Local variables are created in methods and they can only be accessed within method.

P1 is in scope of the main() method and there is one variable in main() method. So there are 2 class variables, 2 instance variables and 1 local variable that can be accessed at P1. Answer is **B.**

Q11 Unused import statements does not cause any issue at compiling and running, so does not duplicate import statement. If a class contains an import statement that cannot be located, the compiler would throw out an error. Unused import statements can easily be removed. It would not cause an error because it is not used. The answer is **B.**

Q12 Instance variables can be accessed only by creating an object. So, in order to use *birds* variable we should create an object of class *ParkRanger*. This way, this code will not compile. The answer is **A.**

Q13 **Java** is an object-oriented programming language.

javac command compiles source code into the bytecode which would be run by JVM.

java command could execute only .class files before Java 11 but it can now execute single source-files with Java 11. **Source**: JEP 330: Launch Single-File Source-Code Programs.

So the answer is **A** according to this recent changes. For earlier Java versions than Java 11, none of the statements are true and the answer would be **D**.

Q14 A Java class file can start with an import statement if there is not a specific package, with a comment line and package statement. But variable definitions cannot be made in first lines. The answer is **D.**

Q15 **A –** There is no need for a package declaration for every class because Java puts classes in the default package if there is no package declared.

B – There is no such a file as *.init* in Java. Packages can be created as folders that classes of this package would be in those folders.

C – The reason of having a package declared is to restrict access to classes in the package from outside.

D – It is the purpose of packages to restrict access, so this is not true.

Q16 *javac* command needs full name of the source file with its extension. *java* command needs only the name of *.class* file. So the only true answer is **B.**

Q17 **A – Platform independence** means that a java bytecode can run on all operating systems.

B – Object orientation means that separate units of a program can run together as one entity.

C – Inheritance is a mechanism that one object can take all properties and attributes of its parent object.

D – Encapsulation is the mechanism of wrapping data and methods together as a single unit. So it ensures that the instance variables of an object can be accessed only by the instance's methods.

Q18 *height* variable is local variable which is in if-statement block. So it cannot be accessed from where *System.out.print()* method is placed and the code does not compile. So the answer is **D**.

Q19 **A –** Java bytecode files can be run on any computer by JVM because JVM is the one that is responsible compile this bytecode to machine code. So the correct answer is **A**.

B – As explained in option **A**, Java is platform independent so java bytecode files can run on all computers.

C – Bytecodes is not readable by humans.

D – JVM can run bytecode file without needing its source file.

Q20 In Java, a semicolon (;) terminates the statements. A colon (:), an end-of-line character (\ n) and a tab character does not have a special meaning in Java. The answer is **D.**

Q21 Although the static variable *yesterday* should be accessed in a static way due to that it can cause a problem when there is ambiguity such as in cases the instance is of a subclass, this does not prevent the variable from being read. So it compiles and outputs 31. The answer is **C.**

Q22 **Line 1** is missing *class* keyword.

Line 2 has two variable type keywords, double and int. Two different variable type keywords cannot be used for single variable.

Line 3 has a valid method definition. So the answer is **C.**

Line 4 has an invalid variable type, void. And *private* keyword cannot be put after variable type keyword.

Q23 **A – Encapsulation** is the mechanism of wrapping data and methods together as a single unit. So it ensures that the instance variables of an object can be accessed only by the instance's methods.

B – Object orientation means that separate units of a program can run together as one entity.

C – Inheritance is a mechanism that one object can take all properties and attributes of its parent object.

D – Platform independence means that a java bytecode can run on all operating systems.

Q24 **Java Virtual Machines** has a garbage collector which releases memory occupied by objects that are not used any more. So a JVM manages memory. JVM translates Java bytecode to machine instructions so that Java programs can run on all computers independently from their operating systems. This is called as **platform independence.** But a JVM does not prevent a bytecode from being decompiled. The answer is **A.**

Q25 **A –** There is no such thing as package variables.

B – Class variables are static variables and accessible from everywhere.

C – Instance variables are available only in the instance of the class.

D – Local variables are only in the scope of the method or block in which they are defined.

Q26 **A –** Importing a package does not import its sub-packages because packages are hierarchical only in their name. A package and its sub-package actually are separate packages. So here, *television.actor.recurring* is a different package from *television.actor* and is not imported. *television.actor.recurring.Marie* is not included.

B – *movie.director* is imported but here plural *directors* word is used.

C – *television.actor* package is imported so the class *Package* that belongs to this package is included.

D – *movie* is not imported. It is a different package from *movie.director.*

Q27 In a Java class file, if there is a package **package statement** comes first. Then **import statement** and finally **class statement** comes. The answer is **D.**

Q28 Because only *stars.Blackhole* class is used all import statements other than this class can be discarded. These are three import statements, *java.lang.** (which is imported by default in every Java program), *stars* (which is unnecessary because *stars.Blackhole* is imported) and *java.lang.Object.* The answer is **D.**

Q29 There is no variable defined as *theInput* so, the program does not compile. If that statement is changed to *...deerParams[2],* it would compile. Then, in order to get *White-tailed* output we should enter such a command, *java package.Class arg1 arg2 White-tailed* and so on.

A – *White-tailed* must be the third argument. In this option the third argument is *"White-tailed deer"*. So this option does not give the desired output.

B – Third argument in this option is *3*. So this option is not correct, too.

C – Third argument in this option is *White-tailed.* So this option is correct.

D – This option does have 2 arguments and the code seeks at least 3 arguments. So it will compile but throws out a runtime error, *ArrayIndexOutOfBoundsException*.

Q30 **A –** *java* command executes bytecode files which has *.class extension.*

B – *javac* command compiles *.java* files into *.class* files. *java* executes *.class* files.

C – *java* command does not compile anything as it is explained in **Q30-A**.

D – *javac* command does not compile *.class* files, it compiles *.java* files.

Q31 **A – Procedural programming** uses procedures to operate on data structures. **Object-oriented programming** bundles the procedures and data structures as an "object", which is an instance of a class, operates on its "own" data structure. Java is an object-oriented programming, not a procedural programming.

B – Method overloading means that a class has multiple methods that have same names but different parameters so that they perform different tasks. Java allows this.

C – Operator overloading, which means giving different definition to operators, is not allowed in Java.

D – Java does not allow direct access to object in memory as they are under control of JVM and available only to Java program.

Q32 The static method has a *return* statement so that this method have a return type. *null* and *void* cannot be return types. So **A** and **B** is wrong. Class definition cannot contain *int*. So **C** is wrong. The answer is **D.**

Q33 The code compiles because the syntax is correct. The constructor parameter *x*'s value is ignored as it is however equated to 4. So *end* is 4, too. e*nd* is 4, *start* is 2; so the first line is four minus two, "2". *distance* is 5, so the second line is "5". The answer is **2 5, A**.

Q34 **Inheritance** allows developers to avoid duplicate code by providing access to superclasses methods and attributes from subclasses. Inheritance does not require that the superclass and its subclasses must be in same package.

Method signature changes in parent classes may break subclasses and program uses extra time and resources when inheritance is implemented but these are counter arguments. The question is the reason of using inheritance. So the answer is **D**.

Q35 Valid comment lines in Java starts with double slash (//) or starts with (/*) and ends with (*/). So the option **A** is correct.

Q36 A valid *main()* method must have in its signature *public*, *static*, *void;* may have *final* and must have a String type array parameter. *(String… arguments)* is also valid for the parameter. **B** has not an array in its parameter. So it is not valid.

Q37 The place of ***line a1*** is not allowed for variable declarations because it is outside of class.

The place of ***line a2*** and ***line a4*** is in class, so there can be instance variables declared.

The place of ***line a3*** is in a method, a local area. Only local variable can be declared there and they cannot have access modifiers such as *public*. The answer is **B.**

Q38 One class declaration makes a java class file valid. Package and import statements are optional and a class has not to be public. The answer is **A.**

Q39 *.jav* and *.source* are not used in Java. *.class* is the bytecode file. Source file extension of Java is *.java*. The answer is **D.**

Q40 *java.lang* package, which contains class *Math*, is imported by default in Java. There is a class called Math in pocket.complex, too. So when using class *Math*, it should be stated explicitly. Line 6 causes ambiguity for this reason and causes the code not to compile. Line 2 and 3 are import statements and the syntax is okay so, they don't cause a problem. The answer is **C.**

Q41 *java.lang* package is imported by default in Java. Packages *dog* and *dog.puppy* is imported so the classes which are in those packages can be accessed without using their full package name. But package *dog.puppy.female* is not imported so, in order to access to a class which is in that package, the full package name must be written. The answer is **A.**

Q42 **A – Encapsulation** is the mechanism of wrapping data and methods together as a single unit. So it ensures that the instance variables of an object can be accessed only by the instance's methods.

B – Object orientation means that separate units of a program can run together as one entity.

C – Platform independence means that a java bytecode can run on all operating systems.

D – Polymorphism is the ability of an object to get into many forms. The most common use of polymorphism occurs when a parent class reference is used to refer to a child class object.

Q43 Apple class is in *food.fruit*, Broccoli class is in *food.vegetables* and Date class is in *java.util* packages. All these classes are used so none of the import statements cannot be discarded. The answer is **A.**

Q44 *numLock* is an instance variable that must be accessed via an object so an object of the class must be created and through that object *numLock* should be accessed in

System.out.print(numLock+" "+capLock);. For that the code cannot recognize the instance variable the code does not compile. So the answer is **C.**

Q45 Although the static variable *wheels* should be accessed in a static way due to that it can cause a problem when there is ambiguity such as in cases the instance is of a subclass, this does not prevent the variable from being read. So it compiles and outputs 20 (feet = 4 + *local* tracks = 15 + *static* wheels = 1). The answer is **D.**

Q46 There is no syntax error in the code so, it compiles. *printColor()* method prints "purple" no matter what it takes as argument. So the answer is **B.**

Q47 *javac* command converts *.java* files into *.class* files by compiling them. *java* use a period (.) to separate packages, not slash (/). So the answer is **C.**

Q48 main() method is missing String type array parameter so the code compiles but throws an error at runtime due to that JVM could not find a main() method to start the execution.. The answer is **D.**

Q49 The class diagram shows us that there is a class called *Book* which has two attributes, one class variable and one method. So the implementation must contain one class definition and one instance variable and one method definition in scope of that class. The option **C** meets this condition.

Q50 **A –** JVM always looks for the methods, objects, threads, etc. and if they stop for a while, then it claims the memory used by them. JVM garbage collector(GC) can also be triggered by being reached to the maximum size of memory. So JVM GC does not work on a schedule, rather it is triggered by the behaviour of the program.

B – JVM is not responsible for the application's termination.

C – JVM needs a main() method to start the application. So this option is true.

D – A Java compiled code needs a JVM installed on a computer to be run.