Java Interfaces and Packages

1 – Java Packages

1. Which of these keywords is used to define packages in Java?

a) pkg

b) Pkg

c) package

d) Package

Answer: c

Explanation: None.

2. Which of these is a mechanism for naming and visibility control of a class and its content?

a) Object

b) Packages

c) Interfaces

d) None of the Mentioned.

Answer: b

Explanation: Packages are both naming and visibility control mechanism. We can define a class inside a package which is not accessible by code outside the package.

3. Which of this access specifies can be used for a class so that its members can be accessed by a different class in the same package?

a) Public

b) Protected

c) No Modifier

d) All of the mentioned

Answer: d

Explanation: Either we can use public, protected or we can name the class without any specifier.

4. Which of these access specifiers can be used for a class so that its members can be accessed by a different class in the different package?

a) Public

b) Protected

c) Private

d) No Modifier

Answer: a

Explanation: None.

5. Which of the following is the correct way of importing an entire package ‘pkg’?

a) import pkg.

b) Import pkg.

c) import pkg.\*

d) Import pkg.\*

Answer: c

Explanation: Operator \* is used to import the entire package.

6. Which of the following is an incorrect statement about packages?

a) Package defines a namespace in which classes are stored

b) A package can contain other package within it

c) Java uses file system directories to store packages

d) A package can be renamed without renaming the directory in which the classes are stored

Answer: d

Explanation: A package can be renamed only after renaming the directory in which the classes are stored.

7. Which of the following package stores all the standard java classes?

a) lang

b) java

c) util

d) java.packages

Answer: a

Explanation: The lang package contains fundamental/standard classes that are essential for Java programming which includes object, string, system, maths and many more. The util package stores classes for various functionalities like collection, date/time, etc. Out the given choices, “lang” package is the most appropriate.

8. What will be the output of the following Java program?

package pkg;

class display

{

int x;

void show()

{

if (x > 1)

System.out.print(x + " ");

}

}

class packages

{

public static void main(String args[])

{

display[] arr=new display[3];

for(int i=0;i<3;i++)

arr[i]=new display();

arr[0].x = 0;

arr[1].x = 1;

arr[2].x = 2;

for (int i = 0; i < 3; ++i)

arr[i].show();

}

}

Note : packages.class file is in directory pkg;

a) 0

b) 1

c) 2

d) 0 1 2

Answer: c

Explanation: None.

9. What will be the output of the following Java program?

package pkg;

class output

{

public static void main(String args[])

{

StringBuffer s1 = new StringBuffer("Hello");

s1.setCharAt(1, x);

System.out.println(s1);

}

}

a) xello

b) xxxxx

c) Hxllo

d) Hexlo

Answer: c

Explanation: None.

10. What will be the output of the following Java program?

package pkg;

class output

{

public static void main(String args[])

{

StringBuffer s1 = new StringBuffer("Hello World");

s1.insert(6 , "Good ");

System.out.println(s1);

}

}

Note : Output.class file is not in directory pkg.

a) HelloGoodWorld

b) HellGoodoWorld

c) Compilation error

d) Runtime error

Answer: d

Explanation: Since output.class file is not in the directory pkg in which class output is defined, program will not be able to run.

Skip to content

Sanfoundry

Menu

2 – Java Interfaces – 1

1. Which of these keywords is used to define interfaces in Java?

a) interface

b) Interface

c) intf

d) Intf

Answer: a

Explanation: None.

2. Which of these can be used to fully abstract a class from its implementation?

a) Objects

b) Packages

c) Interfaces

d) None of the Mentioned

Answer: c

Explanation: None.

3. Which of these access specifiers can be used for an interface?

a) Public

b) Protected

c) private

d) All of the mentioned

Answer: a

Explanation: Access specifier of an interface is either public or no specifier. When no access specifier is used then default access specifier is used due to which interface is available only to other members of the package in which it is declared, when declared public it can be used by any code.

4. Which of these keywords is used by a class to use an interface defined previously?

a) import

b) Import

c) implements

d) Implements

Answer: c

Explanation: interface is inherited by a class using implements.

5. Which of the following is the correct way of implementing an interface salary by class manager?

a) class manager extends salary {}

b) class manager implements salary {}

c) class manager imports salary {}

d) none of the mentioned

Answer: b

Explanation: None.

6. Which of the following is an incorrect statement about packages?

a) Interfaces specifies what class must do but not how it does

b) Interfaces are specified public if they are to be accessed by any code in the program

c) All variables in interface are implicitly final and static

d) All variables are static and methods are public if interface is defined public

Answer: d

Explanation: All methods and variables are implicitly public if interface is declared public.

7. What will be the output of the following Java program?

interface calculate

{

void cal(int item);

}

class display implements calculate

{

int x;

public void cal(int item)

{

x = item \* item;

}

}

class interfaces

{

public static void main(String args[])

{

display arr = new display;

arr.x = 0;

arr.cal(2);

System.out.print(arr.x);

}

}

a) 0

b) 2

c) 4

d) None of the mentioned

Answer: c

Explanation: None.

8. What will be the output of the following Java program?

interface calculate

{

void cal(int item);

}

class displayA implements calculate

{

int x;

public void cal(int item)

{

x = item \* item;

}

}

class displayB implements calculate

{

int x;

public void cal(int item)

{

x = item / item;

}

}

class interfaces

{

public static void main(String args[])

{

displayA arr1 = new displayA;

displayB arr2 = new displayB;

arr1.x = 0;

arr2.x = 0;

arr1.cal(2);

arr2.cal(2);

System.out.print(arr1.x + " " + arr2.x);

}

}

a) 0 0

b) 2 2

c) 4 1

d) 1 4

Answer: c

Explanation: class displayA implements the interface calculate by doubling the value of item, where as class displayB implements the interface by dividing item by item, therefore variable x of class displayA stores 4 and variable x of class displayB stores 1.

9. What will be the output of the following Java program?

interface calculate

{

int VAR = 0;

void cal(int item);

}

class display implements calculate

{

int x;

public void cal(int item)

{

if (item<2)

x = VAR;

else

x = item \* item;

}

}

class interfaces

{

public static void main(String args[])

{

display[] arr=new display[3];

for(int i=0;i<3;i++)

arr[i]=new display();

arr[0].cal(0);

arr[1].cal(1);

arr[2].cal(2);

System.out.print(arr[0].x+" " + arr[1].x + " " + arr[2].x);

}

}

a) 0 1 2

b) 0 2 4

c) 0 0 4

d) 0 1 4

Answer: c

Explanation: None.

3 – Java Interfaces – 2

1. Which of the following access specifiers can be used for an interface?

a) Protected

b) Private

c) Public

d) Public, protected, private

Answer: c

Explanation: Interface can have either public access specifier or no specifier. The reason is they need to be implemented by other classes.

2. Which of the following is the correct way of implementing an interface A by class B?

a) class B extends A{}

b) class B implements A{}

c) class B imports A{}

d) None of the mentioned

Answer: b

Explanation: Concrete class implements an interface. They can be instantiated.

3. All methods must be implemented of an interface.

a) True

b) False

Answer: a

Explanation: Concrete classes must implement all methods in an interface. Through interface multiple inheritance is possible.

4. What type of variable can be defined in an interface?

a) public static

b) private final

c) public final

d) static final

Answer: d

Explanation: variable defined in an interface is implicitly final and static. They are usually written in capital letters.

5. What does an interface contain?

a) Method definition

b) Method declaration

c) Method declaration and definition

d) Method name

Answer: b

Explanation: Interface contains the only declaration of the method.

6. What type of methods an interface contain by default?

a) abstract

b) static

c) final

d) private

Answer: a

Explanation: By default, interface contains abstract methods. The abstract methods need to be implemented by concrete classes.

7. What will happen if we provide concrete implementation of method in interface?

a) The concrete class implementing that method need not provide implementation of that method

b) Runtime exception is thrown

c) Compilation failure

d) Method not found exception is thrown

Answer: c

Explanation: The methods of interfaces are always abstract. They provide only method definition.

8. What happens when a constructor is defined for an interface?

a) Compilation failure

b) Runtime Exception

c) The interface compiles successfully

d) The implementing class will throw exception

Answer: a

Explanation: Constructor is not provided by interface as objects cannot be instantiated.

9. What happens when we access the same variable defined in two interfaces implemented by the same class?

a) Compilation failure

b) Runtime Exception

c) The JVM is not able to identify the correct variable

d) The interfaceName.variableName needs to be defined

Answer: d

Explanation: The JVM needs to distinctly know which value of variable it needs to use. To avoid confusion to the JVM interfaceName.variableName is mandatory.

10. Can “abstract” keyword be used with constructor, Initialization Block, Instance Initialization and Static Initialization Block.

a) True

b) False

Answer: b

Explanation: No, Constructor, Static Initialization Block, Instance Initialization Block and variables cannot be abstract.

4 – Core Java API Packages

1. Which of these package is used for graphical user interface?

a) java.applet

b) java.awt

c) java.awt.image

d) java.io

Answer: b

Explanation: java.awt provides capabilities for graphical user interface.

2. Which of this package is used for analyzing code during run-time?

a) java.applet

b) java.awt

c) java.io

d) java.lang.reflect

Answer: d

Explanation: Reflection is the ability of a software to analyze itself. This is provided by java.lang.reflect package.

3. Which of this package is used for handling security related issues in a program?

a) java.security

b) java.lang.security

c) java.awt.image

d) java.io.security

Answer: a

Explanation: java.security handles certificates, keys, digests, signatures, and other security functions.

4. Which of these class allows us to get real time data about private and protected member of a class?

a) java.io

b) GetInformation

c) ReflectPermission

d) MembersPermission

Answer: c

Explanation: The ReflectPermission class allows reflection of private or protected members of a class. This was added after java 2.0 .

5. Which of this package is used for invoking a method remotely?

a) java.rmi

b) java.awt

c) java.util

d) java.applet

Answer: a

Explanation: java.rmi provides capabilities for remote method invocation.

6. What will be the output of the following Java program?

import java.lang.reflect.\*;

class Additional\_packages

{

public static void main(String args[])

{

try

{

Class c = Class.forName("java.awt.Dimension");

Constructor constructors[] = c.getConstructors();

for (int i = 0; i < constructors.length; i++)

System.out.println(constructors[i]);

}

catch (Exception e)

{

System.out.print("Exception");

}

}

}

a) Program prints all the constructors of ‘java.awt.Dimension’ package

b) Program prints all the possible constructors of class ‘Class’

c) Program prints “Exception”

d) Runtime Error

Answer: a

Explanation: None.

7. What will be the output of the following Java program?

import java.lang.reflect.\*;

class Additional\_packages

{

public static void main(String args[])

{

try

{

Class c = Class.forName("java.awt.Dimension");

Field fields[] = c.getFields();

for (int i = 0; i < fields.length; i++)

System.out.println(fields[i]);

}

catch (Exception e)

{

System.out.print("Exception");

}

}

}

a) Program prints all the constructors of ‘java.awt.Dimension’ package

b) Program prints all the methods of ‘java.awt.Dimension’ package

c) Program prints all the data members of ‘java.awt.Dimension’ package

d) program prints all the methods and data member of ‘java.awt.Dimension’ package

Answer: c

Explanation: None.

8. What is the length of the application box made in the following Java program?

import java.awt.\*;

import java.applet.\*;

public class myapplet extends Applet

{

Graphic g;

g.drawString("A Simple Applet",20,20);

}

a) 20

b) Default value

c) Compilation Error

d) Runtime Error

Answer: c

Explanation: To implement the method drawString we need first need to define abstract method of AWT that is paint() method. Without paint() method we cannot define and use drawString or any Graphic class methods.

9. What will be the output of the following Java program?

import java.lang.reflect.\*;

class Additional\_packages

{

public static void main(String args[])

{

try

{

Class c = Class.forName("java.awt.Dimension");

Method methods[] = c.getMethods();

for (int i = 0; i < methods.length; i++)

System.out.println(methods[i]);

}

catch (Exception e)

{

System.out.print("Exception");

}

}

}

a) Program prints all the constructors of ‘java.awt.Dimension’ package

b) Program prints all the methods of ‘java.awt.Dimension’ package

c) Program prints all the data members of ‘java.awt.Dimension’ package

d) program prints all the methods and data member of ‘java.awt.Dimension’ package

Answer: b

Explanation: None.

5 – Java Type Interface

1. Why are generics used?

a) Generics make code more fast

b) Generics make code more optimised and readable

c) Generics add stability to your code by making more of your bugs detectable at compile time

d) Generics add stability to your code by making more of your bugs detectable at runtime

Answer: c

Explanation: Generics add stability to your code by making more of your bugs detectable at compile time.

2. Which of these type parameters is used for a generic class to return and accept any type of object?

a) K

b) N

c) T

d) V

Answer: c

Explanation: T is used for type, A type variable can be any non-primitive type you specify: any class type, any interface type, any array type, or even another type variable.

3. Which of these type parameters is used for a generic class to return and accept a number?

a) K

b) N

c) T

d) V

Answer: b

Explanation: N is used for Number.

4. Which of these is an correct way of defining generic class?

a) class name(T1, T2, …, Tn) { /\* … \*/ }

b) class name<T1, T2, …, Tn> { /\* … \*/ }

c) class name[T1, T2, …, Tn] { /\* … \*/ }

d) class name{T1, T2, …, Tn} { /\* … \*/ }

Answer: b

Explanation: The type parameter section, delimited by angle brackets (<>), follows the class name. It specifies the type parameters (also called type variables) T1, T2, …, and Tn.

5. Which of the following is an incorrect statement regarding the use of generics and parameterized types in Java?

a) Generics provide type safety by shifting more type checking responsibilities to the compiler

b) Generics and parameterized types eliminate the need for down casts when using Java Collections

c) When designing your own collections class (say, a linked list), generics and parameterized types allow you to achieve type safety with just a single class definition as opposed to defining multiple classes

d) All of the mentioned

Answer: c

Explanation: None.

6. Which of the following reference types cannot be generic?

a) Anonymous inner class

b) Interface

c) Inner class

d) All of the mentioned

Answer: a

Explanation: None.

7. What will be the output of the following Java program?

public class BoxDemo

{

public static <U> void addBox(U u, java.util.List<Box<U>> boxes)

{

Box<U> box = new Box<>();

box.set(u);

boxes.add(box);

}

public static <U> void outputBoxes(java.util.List<Box<U>> boxes)

{

int counter = 0;

for (Box<U> box: boxes)

{

U boxContents = box.get();

System.out.println("Box #" + counter + " contains [" + boxContents.toString() + "]");

counter++;

}

}

public static void main(String[] args)

{

java.util.ArrayList<Box<Integer>> listOfIntegerBoxes = new java.util.ArrayList<>();

BoxDemo.<Integer>addBox(Integer.valueOf(10), listOfIntegerBoxes);

BoxDemo.outputBoxes(listOfIntegerBoxes);

}

}

a) 10

b) Box #0 [10]

c) Box contains [10]

d) Box #0 contains [10]

Answer: d

Explanation: None.

8. What will be the output of the following Java program?

public class BoxDemo

{

public static <U> void addBox(U u,

java.util.List<Box<U>> boxes)

{

Box<U> box = new Box<>();

box.set(u);

boxes.add(box);

}

public static <U> void outputBoxes(java.util.List<Box<U>> boxes)

{

int counter = 0;

for (Box<U> box: boxes)

{

U boxContents = box.get();

System.out.println("[" + boxContents.toString() + "]");

counter++;

}

}

public static void main(String[] args)

{

java.util.ArrayList<Box<Integer>> listOfIntegerBoxes = new java.util.ArrayList<>();

BoxDemo.<Integer>addBox(Integer.valueOf(0), listOfIntegerBoxes);

BoxDemo.outputBoxes(listOfIntegerBoxes);

}

}

a) 0

b) 1

c) [1]

d) [0]

Answer: d

Explanation: None.

9. What will be the output of the following Java program?

import java.util.\*;

public class genericstack <E>

{

Stack <E> stk = new Stack <E>();

public void push(E obj)

{

stk.push(obj);

}

public E pop()

{

E obj = stk.pop();

return obj;

}

}

class Output

{

public static void main(String args[])

{

genericstack <String> gs = new genericstack<String>();

gs.push("Hello");

System.out.print(gs.pop() + " ");

genericstack <Integer> gs = new genericstack<Integer>();

gs.push(36);

System.out.println(gs.pop());

}

}

a) Error

b) Hello

c) 36

d) Hello 36

Answer: d

Explanation: None.

10. What will be the output of the following Java program?

public class BoxDemo

{

public static <U> void addBox(U u,

java.util.List<Box<U>> boxes)

{

Box<U> box = new Box<>();

box.set(u);

boxes.add(box);

}

public static <U> void outputBoxes(java.util.List<Box<U>> boxes)

{

int counter = 0;

for (Box<U> box: boxes)

{

U boxContents = box.get();

System.out.println("Box #" + counter + " contains [" + boxContents.toString() + "]");

counter++;

}

}

public static void main(String[] args)

{

java.util.ArrayList<Box<Integer>> listOfIntegerBoxes = new java.util.ArrayList<>();

BoxDemo.<Integer>addBox(Integer.valueOf(10), listOfIntegerBoxes);

BoxDemo.outputBoxes(listOfIntegerBoxes);

}

}

a) 10

b) Box #0 [10]

c) Box contains [10]

d) Box #0 contains [10]

Answer: d

Explanation: None.