|  |  |  |
| --- | --- | --- |
| **Skill** | **Subskill** | **Competency** |
| Abstraction | Abstract Classes | Abstract class features |
| Abstraction & Interfaces | Interface Features | Interface capabilities |
| Arrays | Declaration and Properties | Understanding array behaviour |
| Indexing | Array bounds |
| Classes & Objects | Constructors | Object initialization |
| Control Structures | Loops | Loop behaviour |
| Data Types | Primitive Types | Understanding data type ranges |
| Encapsulation | Access Modifiers | Controlling access |
| OOP Principles | Data hiding |
| Inheritance | Class Hierarchies | Understanding inheritance limitations |
| Inheritance & Polymorphism | Method Overriding and Variable Hiding | Understanding dynamic method dispatch |
| Method Overloading | Compile-time Polymorphism | Method signature differentiation |
| Signature Matching | Valid overloads |
| Access Control | Preventing method overriding |
| Methods & Functions | Return Types | Method declaration |
| Operators | Arithmetic and String Concatenation | Operator precedence and type conversion |
| Polymorphism | Runtime Polymorphism | Method overriding |
| Static vs Instance Members | Static Methods | Understanding static context |
| Static vs Instance Members | Static Variables | Shared class-level data |
| Strings | Immutability | String manipulation |

**Multiple Choice Question 1**

Which of the following data types can store the largest positive value in Java?   
Options:   
A. int  
B. long  
C. float  
D. double  
**Skill:** Data Types  
**Subskill:** Primitive Types  
**Competency:** Understanding data type ranges  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  double can store larger values than float, int, or long due to its 64-bit floating-point representation.
* **Incorrect Options Feedback:**
  + A. int is 32-bit and smaller than long.
  + B. long is 64-bit but stores integers only.
  + C. float is 32-bit floating-point, smaller than double.

**Multiple Choice Question 2**

What is the result of the following expression: 10 + 20 + "30"?   
Options:   
A. 60  
B. 102030  
C. 3030  
D. 30  
**Skill:** Operators  
**Subskill:** Arithmetic and String Concatenation  
**Competency:** Operator precedence and type conversion  
**Difficulty Level:** Intermediate  
**Bloom Level:** Application

* **Correct Answer Reason:**  
  The expression evaluates left to right: 10 + 20 = 30, then 30 + "30" results in "3030" due to string concatenation.
* **Incorrect Options Feedback:**
  + A. 60 assumes all are integers.
  + B. 102030 assumes all are strings.
  + D. 30 is only the string part.

**Multiple Choice Question 3**

Which control structure is best suited for executing a block of code at least once regardless of the condition?   
Options:   
A. for loop  
B. while loop  
C. do-while loop  
D. switch  
**Skill:** Control Structures  
**Subskill:** Loops  
**Competency:** Loop behavior  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  do-while executes the loop body at least once before checking the condition.
* **Incorrect Options Feedback:**
  + A & B. for and while check the condition before execution.
  + D. switch is not a loop.

**Multiple Choice Question 4**

What is the output of the following code?

String s = "Java";

s.concat(" Programming");

System.out.println(s);

Options:   
A. Java Programming  
B. Java  
C. Programming  
D. Compilation error  
**Skill:** Strings  
**Subskill:** Immutability  
**Competency:** String manipulation  
**Difficulty Level:** Intermediate  
**Bloom Level:** Application

* **Correct Answer Reason:**  
  Strings are immutable; concat() returns a new string, but s is not reassigned.
* **Incorrect Options Feedback:**
  + A. Would be correct if s = s.concat(...) was used.
  + C. Incorrect, only "Programming" is not printed.
  + D. No compilation error.

**Multiple Choice Question 5**

Which of the following is true about Java arrays?   
Options:   
A. Arrays can grow dynamically  
B. Arrays can store different data types  
C. Arrays are objects  
D. Arrays are primitive types  
**Skill:** Arrays  
**Subskill:** Declaration and Properties  
**Competency:** Understanding array behavior  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  Arrays in Java are objects, even if they store primitive types.
* **Incorrect Options Feedback:**
  + A. Arrays have fixed size.
  + B. Arrays are homogeneous.
  + D. Arrays are not primitive types.

**Multiple Choice Question 6**

Which keyword is used to define a method that does not return any value?

Options:   
A. void  
B. null  
C. return  
D. empty  
**Skill:** Methods & Functions  
**Subskill:** Return Types  
**Competency:** Method declaration  
**Difficulty Level:** Intermediate  
**Bloom Level:** Remembering

* **Correct Answer Reason:**  
  void indicates the method does not return a value.
* **Incorrect Options Feedback:**
  + B. null is a value, not a return type.
  + C. return is a statement, not a type.
  + D. empty is not a valid keyword.

**Multiple Choice Question 7**

What is the purpose of a constructor in a Java class?

Options:

A. To destroy objects  
B. To initialize object state  
C. To call other methods  
D. To inherit from another class  
**Skill:** Classes & Objects  
**Subskill:** Constructors  
**Competency:** Object initialization  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  Constructors initialize the state of an object when it is created.
* **Incorrect Options Feedback:**
  + A. Destructors are not used in Java.
  + C. Methods can be called inside constructors, but that’s not their main purpose.
  + D. Inheritance is handled via extends.

**Multiple Choice Question 8**

Which access modifier allows visibility only within the same package?

Options:   
A. private  
B. protected  
C. public  
D. default (no modifier)  
**Skill:** Encapsulation  
**Subskill:** Access Modifiers  
**Competency:** Controlling access  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  Default access (no modifier) restricts visibility to the same package.
* **Incorrect Options Feedback:**
  + A. private restricts to the same class.
  + B. protected allows access in subclasses and same package.
  + C. public allows access everywhere.

**Multiple Choice Question 9**

Which of the following is true about method overloading?

Options:   
A. It allows multiple methods with the same name and same parameters  
B. It allows methods with the same name but different return types only  
C. It allows methods with the same name but different parameter lists  
D. It overrides methods from the superclass  
**Skill:** Method Overloading  
**Subskill:** Compile-time Polymorphism  
**Competency:** Method signature differentiation  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  Overloading allows methods with the same name but different parameter lists.
* **Incorrect Options Feedback:**
  + A. Same parameters cause conflict.
  + B. Return type alone is not sufficient.
  + D. That’s overriding.

**Multiple Choice Question 10**

Which of the following is **not** a feature of inheritance in Java?

Options:   
A. Code reusability  
B. Method overriding  
C. Multiple inheritance with classes  
D. Polymorphism  
**Skill:** Inheritance  
**Subskill:** Class Hierarchies  
**Competency:** Understanding inheritance limitations  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  Java does not support multiple inheritance with classes.
* **Incorrect Options Feedback:**
  + A, B, D. All are valid features of inheritance.

**Multiple Choice Question 11**

Which of the following statements about interfaces in Java is **true**?

Options:   
A. Interfaces can have constructors  
B. Interfaces can have private instance variables  
C. Interfaces can have static and default methods  
D. Interfaces cannot have any methods  
**Skill:** Abstraction & Interfaces  
**Subskill:** Interface Features  
**Competency:** Interface capabilities  
**Difficulty Level:** Advanced  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  Since Java 8, interfaces can have static and default methods.
* **Incorrect Options Feedback:**
  + A. Interfaces cannot have constructors.
  + B. Interfaces cannot have instance variables.
  + D. Interfaces can have abstract, default, and static methods.

**Multiple Choice Question 12**

What will be the output of the following code?

class A {

int x = 10;

void print() {

System.out.println("A");

}

}

class B extends A {

int x = 20;

void print() {

System.out.println("B");

}

}

public class Test {

public static void main(String[] args) {

A obj = new B();

System.out.println(obj.x);

obj.print();

}

}

Options:   
A. 10 and A  
B. 20 and B  
C. 10 and B  
D. 20 and A  
**Skill:** Inheritance & Polymorphism  
**Subskill:** Method Overriding and Variable Hiding  
**Competency:** Understanding dynamic method dispatch  
**Difficulty Level:** Advanced  
**Bloom Level:** Analysis

* **Correct Answer Reason:**  
  Method print() is overridden, so B is printed. But x is hidden, not overridden, so A's x is accessed.
* **Incorrect Options Feedback:**
  + A. Incorrect method output.
  + B. Incorrect variable access.
  + D. Incorrect variable and method access.

**Multiple Choice Question 13**

Which keyword is used to prevent a method from being overridden?

Options:   
A. static  
B. final  
C. private  
D. abstract  
**Skill:** Method Overriding  
**Subskill:** Access Control  
**Competency:** Preventing method overriding  
**Difficulty Level:** Intermediate  
**Bloom Level:** Remembering

* **Correct Answer Reason:**  
  final prevents a method from being overridden in subclasses.
* **Incorrect Options Feedback:**
  + A. static methods can't be overridden but not due to final.
  + C. private methods are not inherited.
  + D. abstract methods must be overridden.

**Multiple Choice Question 14**

What is the result of the following code?

public class Test {

static int count = 0;

public Test() {

count++;

}

public static void main(String[] args) {

new Test();

new Test();

System.out.println(Test.count);

}

}

Options:   
A. 0  
B. 1  
C. 2  
D. Compilation error  
**Skill:** Static vs Instance Members  
**Subskill:** Static Variables  
**Competency:** Shared class-level data  
**Difficulty Level:** Intermediate  
**Bloom Level:** Application

* **Correct Answer Reason:**  
  count is static and shared across all instances. Two objects increment it to 2.
* **Incorrect Options Feedback:**
  + A. Would be true if constructor didn’t increment.
  + B. Only one object created.
  + D. No compilation error.

**Multiple Choice Question 15**

Which of the following best describes encapsulation?

Options:   
A. Hiding implementation details and exposing only functionality  
B. Inheriting from a superclass  
C. Overriding methods in a subclass  
D. Using multiple classes in a program  
**Skill:** Encapsulation  
**Subskill:** OOP Principles  
**Competency:** Data hiding  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  Encapsulation hides internal state and requires all interaction through methods.
* **Incorrect Options Feedback:**
  + B. Describes inheritance.
  + C. Describes polymorphism.
  + D. Not specific to encapsulation.

**Multiple Choice Question 16**

Which of the following is **not** a valid method signature for overloading?   
Options:   
A. void print(int a)  
B. void print(int a, int b)  
C. int print(int a)  
D. void print(int x)  
**Skill:** Method Overloading  
**Subskill:** Signature Matching  
**Competency:** Valid overloads  
**Difficulty Level:** Intermediate  
**Bloom Level:** Analysis

* **Correct Answer Reason:**  
  D is not a valid overload because it has the same parameter type and count as A; parameter names don’t matter.
* **Incorrect Options Feedback:**
  + A, B, C. All have different parameter lists or return types (though return type alone doesn’t differentiate overloads).

**Multiple Choice Question 17**

Which of the following is **true** about abstract classes in Java?

Options:   
A. They can be instantiated directly  
B. They cannot have constructors  
C. They can have both abstract and non-abstract methods  
D. They must implement all methods of an interface  
**Skill:** Abstraction  
**Subskill:** Abstract Classes  
**Competency:** Abstract class features  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  Abstract classes can have both abstract and concrete methods.
* **Incorrect Options Feedback:**
  + A. Abstract classes cannot be instantiated.
  + B. They can have constructors.
  + D. Only concrete subclasses must implement all interface methods.

**Multiple Choice Question 18**

What is the output of the following code?

public class Test {

public static void main(String[] args) {

int[] arr = {1, 2, 3};

System.out.println(arr[3]);

}

}

Options:   
A. 3  
B. ArrayIndexOutOfBoundsException  
C. null  
D. Compilation error  
**Skill:** Arrays  
**Subskill:** Indexing  
**Competency:** Array bounds  
**Difficulty Level:** Intermediate  
**Bloom Level:** Application

* **Correct Answer Reason:**  
  Index 3 is out of bounds for an array of size 3 (0–2), causing an exception.
* **Incorrect Options Feedback:**
  + A. Index 3 doesn’t exist.
  + C. Not applicable to primitive arrays.
  + D. Code compiles fine.

**Multiple Choice Question 19**

Which of the following statements about static methods is **false**?

Options:   
A. Static methods can be called without creating an object  
B. Static methods can access instance variables directly  
C. Static methods belong to the class, not instances  
D. Static methods can call other static methods  
**Skill:** Static vs Instance Members  
**Subskill:** Static Methods  
**Competency:** Understanding static context  
**Difficulty Level:** Intermediate  
**Bloom Level:** Understanding

* **Correct Answer Reason:**  
  Static methods cannot access instance variables directly; they need an object reference.
* **Incorrect Options Feedback:**
  + A, C, D. All are true about static methods.

**Multiple Choice Question 20**

Which of the following best demonstrates polymorphism in Java?

Options:   
A. Using final methods  
B. Overloading methods with different parameters  
C. Overriding a method in a subclass  
D. Declaring variables as private  
**Skill:** Polymorphism  
**Subskill:** Runtime Polymorphism  
**Competency:** Method overriding  
**Difficulty Level:** Intermediate  
**Bloom Level:** Application

* **Correct Answer Reason:**  
  Method overriding is a key example of runtime polymorphism.
* **Incorrect Options Feedback:**
  + A. final prevents overriding.
  + B. Overloading is compile-time polymorphism.
  + D. Related to encapsulation.

**Assessment Name on Prism: Cigna QE Automation Week 2 Assessment**