1 Exam Questions

1. What is printed by the following code snippet?

```
1 int birb = 1 + 2 * 5>=2 ? 4 : 2;
2 int mammals = 3 < 3 ? 1 : 5>=5 ? 9 : 7;
3 System.out.print(birb+mammals+"");
```

   A. `49`

   B. `13`

   C. `18`

   D. `99`

   E. It does not compile

2. Which of the following statements about objects, reference types, and casting are correct?

   A. An object can be assigned to an inherited interface reference variable without an explicit cast.

   B. The compiler can prevent all explicit casts that lead to an exception at runtime.

   C. Casting an object to a reference variable does not modify the object in memory.

   D. An object can be assigned to a subclass reference variable without an explicit cast.

   E. An object can be assigned to a superclass reference variable without an explicit cast.

   F. An implicit cast of an object to one of its inherited types can sometimes lead to a `ClassCastException` at runtime.

3. What is the output of the following when run as `java WhatAClass seed flower plant`?

```
1 package unix;
2 import java.util.*;
3 public class WhatAClass {
4   public static void main(String[] args) {
5     int result = Arrays.binarySearch(args, args[0]);
6     System.out.println(result);
7   }
8 }
```

   A. `0`

   B. `1`

   C. `2`

   D. The code does not compile.

   E. The code compiles but throws an exception at runtime.

   F. The output is not guaranteed.

4. How many objects are eligible for garbage collection at the end of the `main()` method?

```
1  package store;
2  public class Primates {
3    static String primate1 = new String("oranghutan");
4    static String primate2 = new String("lemur");
5    public static void primateMethod() {
6      String primate3 = new String("gorilla");
7      primate2 = primate1;
8      primate1 = primate3;
9    }
10   public static void main(String... args) {
11     primateMethod();
12   }
13 }
```

A. `0`

B. `1`

C. `2`

D. `3`

E. The code does not compile

5. Fill in the blanks: The _____ keyword is used in method declarations, the _____ keyword is used to guarantee a statement will execute even if an exception is thrown, and the _____ keyword is used to throw an exception to the surrounding process.

A. `throw, finally, throws`

B. `throws, catch, throw`

C. `catch, finally, throw`

D. `finally, catch, throw`

E. `throws, finally, throw`

6. Which statements best describe the result of this code?

```
1  package asean;
2  public class FlyingCar {
3    public static void main(String... args) {
4      String[] aseanTourLoops = new String[] { "Malaysia", "
     Indonesia", "Philippines" };
5      String[] times = new String[] { "Day", "Night" };
6      for (int i = 0, j = 0; i < aseanTourLoops.length; i++, j++)
7        System.out.println(aseanTourLoops[i] + " " + times[j]);
8    }
9  }
```

A. The `println` causes one line of output.

B. The `println` causes two lines of output.

C. The `println` causes three lines of output.

D. The code terminates successfully.

E. The code throws an exception at runtime.

7. Fill in the blanks: Because of _____, it is possible to _____ a method, which allows Java to support _____.

   A. abstract methods, override, inheritance

   B. concrete methods, overload, inheritance

   C. virtual methods, overload, interfaces

   D. inheritance, abstract, polymorphism

   E. virtual methods, override, polymorphism.

8. What is the result of the following?

```
1  package calendar;
2  public class Seasons {
3    public static void seasons(String... names) {
4      int l = names[1].length(); // s1
5      System.out.println(names[l]); // s2
6    }
7    public static void main(String[] args) {
8      seasons("Summer", "Fall", "Winter", "Spring");
9    }
10 }
```

   A. `Fall`

   B. `Spring`

   C. The code does not compile

   D. The code throws an exception on line `s1`

   E. The code throws an exception on line `s2`

9. How many lines of the following application contain compilation errors?

```
1  package percussion;
2
3  interface BadInterface {}
4  abstract class Abstract implements BadInterface {
5      public Abstract(int stones) {}
6      public void throwRock() {}
7  }
8  public class Concrete extends Abstract {
9      public void throwRock(int count) {}
10     public void fight() {
11         super.throwRock(5);
12     }
13     public static void main(String[] stones) {
14         BadInterface mn = new Concrete();
```

```
15          mn.fight();
16      }
17  }
```

A. None. The code compiles and runs without issue.

B. 1

C. 2

D. 3

E. 4

10. What is the output of the following code?

```
1  package fly;
2  public class Penguin {
3    public int adjustFlippers(int length, String[] type) {
4      length++;
5      type[0] = "LONG";
6      return length;
7    }
8    public static void main(String[] climb) {
9      final Penguin p = new Penguin();
10     int length = 5;
11     String[] type = new String[1];
12     length = p.adjustFlippers(length, type);
13     System.out.print(length+","+type[0]);
14   }
15 }
```

A. `5,LONG`

B. `6,LONG`

C. `5,null`

D. `6,null`

E. The code does not compile

F. The code compiles but throws an exception at runtime.

11. Examine the following code and select the correct statement (choose 1 option).

```
1  class StringBuilders {
2      public static void main(String... args) {
3          StringBuilder sb1 = new StringBuilder("eLion");
4          String jaud = null;
5          jaud = sb1.append("X").substring(sb1.indexOf("L"), sb1.
               indexOf("X"));
6          System.out.println(jaud);
7      }
8  }
```

A. The code will print `LionX`

B. The code will print `Lion`

C. The code will print `Lion` if line 5 is changed to the following:

```
jaud = sb1.append("X").substring(sb1.indexOf('L'), sb1.
   indexOf('X'));
```

D. The code will compile only when line 4 is changed to the following:

```
StringBuilder jaud = null;
```

12. Given the following code,

```
interface Jumpable {
  int height = 1;
  default void worldRecord() {
    System.out.print(height);
  }
}
interface Moveable {
  int height = 2;
  static void worldRecord() {
    System.out.print(height);
  }
}

class Kangaroo implements Jumpable, Moveable {
  int height = 3;
  Kangaroo() {
    worldRecord();
  }
  public static void main(String args[]) {
    Jumpable j = new Kangaroo();
    Moveable m = new Kangaroo();
    Chair c = new Kangaroo();
  }
}
```

what is the output? Select 1 option.

A. 111

B. 123

C. 333

D. 222

E. Compilation error

F. Runtime exception

13. Given the following code, which option, if used to replace `/* INSERT CODE HERE */`, will enable the class `Jungle` to determine whether the reference variable `animal` refers to an object of the class `Penguin` and print `1`? (Select 1 option.)

```
class Animal{ float age; }
class Penguin extends Animal { int beak;}
class Jungle {
  public static void main(String args[]) {
    Animal animal = new Penguin();
    /* INSERT CODE HERE */
    System.out.println(1);
  }
}
```

A. `if (animal instanceof Penguin)`

B. `if (animal instanceOf Penguin)`

C. `if (animal == Penguin)`

D. `if (animal = Penguin)`

14. Given that the file `Test.java`, which defines the following code, fails to compile, select the reasons for the compilation failure (choose 2 options).

```
class Human {
  Human(String value) {}
}
class Michael extends Human {}
class Test {
  public static void main(String args[]) {
    Michael m = new Michael();
  }
}
```

A. The class `Human` fails to compile.

B. The class `Michael` fails to compile.

C. The default constructor can call only a no-argument constructor of a base class.

D. The code that creates the object of the class `Michael` in the class `Test` did not pass a `String` value to the constructor of the class `Michael`.

15. Examine the following code and select the correct statements (choose 2 options).

```
class Bottle {
  void Bottle() {}
  void Bottle(MayonnaiseBottle w) {}
}
class MayonnaiseBottle extends Bottle {}
```

A. A base class can't pass reference variables of its defined class as method parameters in constructors.

B. The class compiles successfully—a base class can use reference variables of its derived class as method parameters.

C. The class `Bottle` defines two overloaded constructors.

D. The class `Bottle` can access only one constructor.

16. Given the following code, which option, if used to replace `/* INSERT CODE HERE */`, will cause the code print `110`? (Select 1 option.)

```
class Book {
  private int pages = 100;
}
class EBook extends Book {
  private int interviews = 2;
  private int totalPages() { /* INSERT CODE HERE */ }
  public static void main(String[] args) {
    System.out.println(new EBook().totalPages());
  }
}
```

A. `return super.pages + this.interviews*5;`

B. `return this.pages + this.interviews*5;`

C. `return super.pages + interviews*5;`

D. `return pages + this.interviews*5;`

E. None of the above

17. Given the following code,

```
class NoMoneyException extends Exception {}
class Pen{
  void write(String val) throws NoMoneyException {
    int c = (10 - 7)/ (8 - 2 - 6);
  }
  void article() {
    //INSERT CODE HERE
  }
}
```

which of the options, when inserted at `//INSERT CODE HERE`, will define a valid use of the method `write` in the method `article`? (Select 2 options.)

A.
```
try {
    new Pen().write("story");
  } catch (NoMoneyException e) {}
```

B.
```
try {
    new Pen().write("story");
  } finally {}
```

C.
```
try {
    write("story");
  } catch (Exception e) {}
```

D.
```
try {
    new Pen().write("story");
  } catch (RuntimeException e) {}
```

18. What is the output of the following code? (Select 1 option.)

```
class JammyJellyfish {
  static String name = "m1";
  void ubuntuName() {
    String name = "m2";
    if (8 > 2) {
      String name = "m3";
      System.out.println(name);
    }
  }
  public static void main(String[] args) {
    JammyJellyfish m1 = new JammyJellyfish();
    m1.ubuntuName();
  }
}
```

A. `m1`

B. `m2`

C. `m3`

D. The code fails to compile

19. What is the output of the following code? (Select 1 option.)

```
class JaBowl {
  public static void main(String args[]) {
    String jasFood = "Corn";
    System.out.println(jasFood);
    mix(jasFood);
    System.out.println(jasFood);
  }
  static void mix(String foodIn) {
    foodIn.concat("A");
    foodIn.replace('C', 'B');
  }
}
```

A.
```
Corn
BornA
```

B.
```
Corn
CornA
```

C. `Corn`
`Born`

D. `Corn`
`Corn`

20. What is the output of the following code? (Select 1 option.)

```
class SwJava {
  public static void main(String args[]) {
    String[] shapes = {"Circle", "Square", "Triangle"};
    switch (shapes) {
      case "Square": System.out.println("Circle"); break;
      case "Triangle": System.out.println("Square"); break;
      case "Circle": System.out.println("Triangle"); break;
    }
  }
}
```

A. The code prints `Circle`

B. The code prints `Square`

C. The code prints `Triangle`

D. The code prints
`Circle`
`Square`
`Triangle`

E. The code prints
`Triangle`
`Circle`
`Square`

F. The code fails to compile

21. Given the following definition of the classes `Human`, `Father`, and `Home`, which option, if used to replace `//INSERT CODE HERE`, will cause the code to compile successfully? (Select 3 options.)

```
class Human {}
class Father extends Human {
  public void dance() throws ClassCastException {}
}
class Home {
  public static void main(String args[]) {
    Human p = new Human();
    try {
      ((Father)p).dance();
    }
    //INSERT CODE HERE
  }
}
```

A.
```
catch (NullPointerException e) {}
catch (ClassCastException e) {}
catch (Exception e) {}
catch (Throwable t) {}
```

B.
```
catch (ClassCastException e) {}
catch (NullPointerException e) {}
catch (Exception e) {}
catch (Throwable t) {}
```

C.
```
catch (ClassCastException e) {}
catch (Exception e) {}
catch (NullPointerException e) {}
catch (Throwable t) {}
```

D.
```
catch (Throwable t) {}
catch (Exception e) {}
catch (ClassCastException e) {}
catch (NullPointerException e) {}
```

E. `finally`

22. What is the output of the following code? (Select 1 option.)

```
import java.time.*;
class Camera {
  public static void main(String args[]) {
    int hours;
    LocalDateTime now = LocalDateTime.of(2020, 10, 01, 0 , 0);
    LocalDate before = now.toLocalDate().minusDays(1);
    LocalTime after = now.toLocalTime().plusHours(1);
    while (before.isBefore(after) && hours < 4) {
      ++hours;
    }
    System.out.println("Hours:" + hours);
  }
}
```

A. The code prints `Camera:null`.

B. The code prints `Camera:Adjust settings manually`

C. The code prints `Camera:`.

D. The code will fail to compile.

23. The output of the class `TestJaJavaCourse`, defined as follows, is `300`:

```
class Course {
  int enrollments;
}
class TestJaJavaCourse {
```

```
  public static void main(String args[]) {
    Course c1 = new Course();
    Course c2 = new Course();
    c1.enrollments = 100;
    c2.enrollments = 200;
    System.out.println(c1.enrollments + c2.enrollments);
  }
}
```

What will happen if the variable `enrollments` is defined as a `static` variable? (Select 1 option.)

A. No change in output. `TestJaJavaCourse` prints `300`.

B. Change in output. `TestJaJavaCourse` prints `200`.

C. Change in output. `TestJaJavaCourse` prints `400`.

D. The class `TestJaJavaCourse` fails to compile.

24. What is the output of the following code? (Select 1 option.)

```
String jaudStr[] = new String[][]{{null},new String[]{"a","b","c
   "},{new String()}}[0] ;
String jaudStr1[] = null;
String jaudStr2[] = {null};
System.out.println(jaudStr[0]);
System.out.println(jaudStr2[0]);
System.out.println(jaudStr1[0]);
```

A.
```
null
NullPointerException
```

B.
```
null
null
NullPointerException
```

C.
```
NullPointerException
```

D.
```
null
null
null
```

25. Examine the following code and select the correct statement (choose 1 option).

```
1 import java.util.*;
2 class Human {}
3 class Emp extends Human {}
4 class TestArrayList {
5   public static void main(String[] args) {
6     ArrayList<Object> list = new ArrayList<>();
7     list.add(new String("1234")); //LINE1
8     list.add(new Human()); //LINE2
9     list.add(new Emp()); //LINE3
```

```
10      list.add(new String[]{"abcd", "xyz"}); //LINE4
11      list.add(LocalDate.now().plus(1)); //LINE5
12    }
13  }
```

A. The code on line 1 won't compile.

B. The code on line 2 won't compile.

C. The code on line 3 won't compile.

D. The code on line 4 won't compile.

E. The code on line 5 won't compile.

F. None of the above.

G. All the options from (A) through (E).

26. Examine the following code and select the correct statement (choose 1 option).

```
public class If2 {
  public static void main(String args[]) {
    int a = 10; int b = 20; boolean c = false;
    if (b > a) if (++a == 10) if (c!=true) System.out.println(1);
    else System.out.println(2); else System.out.println(3);
  }
}
```

A. `1`

B. `2`

C. `3`

D. No output

2 Programming Exercises

1. Create a Java program that takes a user-inputted string and performs the following tasks:
    1. Reverse the string: Reverse the order of characters in the string.
    2. Check for Palindrome: Determine if the string is a palindrome (reads the same backward as forward).
    3. Count Vowels: Count the number of vowels (a, e, i, o, u) in the string.
    4. **(OPTIONAL)** Extract Email Addresses: Extract all email addresses from the string using regular expressions.
2. Create a Java program to simulate a restaurant management system. The system should have the following classes:
    1. `MenuItem`: This class should represent a menu item with properties like `name`, `price`, and `category` (e.g., appetizer, main course, dessert).
    2. `Order`: This class should represent a customer's order, including a list of menu items and the total cost.
    3. `Waiter`: This class should represent a waiter who can take orders, process payments, and deliver food.
    4. `Kitchen`: This class should simulate the kitchen, receiving orders from waiters, preparing food, and notifying waiters when orders are ready.

    Tasks

    1. Create Menu: Implement the MenuItem class and create a menu with various items, categorized as appetizers, main courses, and desserts.
    2. Take Orders: The `Waiter` class should allow customers to order items from the menu.
    3. Process Orders: The `Waiter` should send orders to the `Kitchen` class.
    4. Prepare Food: The `Kitchen` class should simulate food preparation time and notify the `Waiter` when an order is ready.
    5. Deliver Orders: The `Waiter` should deliver the prepared food to the customer.
    6. Calculate Bill: The `Waiter` should calculate the total bill for the customer's order, including taxes and any discounts.

    Optional:

    1. Error Handling: Implement error handling for situations like out-of-stock items or invalid orders.
    2. User Interface: Create a simple text-based user interface to interact with the system.
    3. Data Structures: Use appropriate data structures like `ArrayList`s or `HashMaps` to store menu items, orders, and customer information.
    4. Concurrency: Consider using threads to simulate concurrent order processing and food preparation.
3. Create a Java program that reads a text file containing a list of words, one word per line. The program should then:
    1. Count word frequencies: Count the frequency of each word in the file.
    2. Sort words by frequency: Sort the words by their frequency, descending order.
    3. Write the sorted word frequencies to a new file.

4. Create a Java program that simulates a library catalog. The program should store a list of books with their titles and ISBN numbers. The user should be able to:
    1. Add new books to the catalog
    2. Search for a book by title or ISBN using binary search
    3. Display a list of all books in the catalog