Assessment

Q1. **What is the size of float and double in java?**
A. 32 and 64 B. 32 and 32 C. 64 and 64 D. 64 and 32

The sizes of the float and double data types. Here are the specifics:

Float : Size: 32 bits (4 bytes)

Double : Size: 64 bits (8 bytes)

So, the output is **A. 32 and 64**

Q2. Automatic type conversion is possible in which of the possible cases?
A. Byte to int B. Int to long C. Long to int D. Short to int

In Java, automatic type conversion is possible when converting a smaller data type to a larger data type without any explicit cast. This is because the larger data type can safely accommodate all possible values of the smaller type.

A. Byte to int - Yes, automatic type conversion is possible. A byte is 8 bits and can be automatically promoted to an int, which is 32 bits.

B. Int to long - Yes, automatic type conversion is possible. An int is 32 bits and can be automatically promoted to a long, which is 64 bits.

C. Long to int - No, automatic type conversion is not possible. A long is 64 bits and cannot be safely converted to an int, which is 32 bits, without potential data loss. This conversion requires explicit casting.

D. Short to int - Yes, automatic type conversion is possible. A short is 16 bits and can be automatically promoted to an int, which is 32 bits.

So, the output is :

- **Byte to int (A)**
- **Int to long (B)**
- **Short to int (D)**

Q3.Find the output of the following code.

```
int Integer = 24;
char String = 'I';
System.out.print(Integer);
System.out.print(String);
```

A. Compile error B. Throws exception C. I D. 24 I

int Integer = 24; declares an integer variable named Integer and initializes it with the value 24.
char String = 'I'; declares a character variable named String and initializes it with the character 'I'.

First, it print 24
Second, it print 'I'

So, the output is **D. 24 I**

Q4.Find the output of the following program.

```
public class Solution{
  public static void main(String[] args){
        short x = 10;
        x = x * 5;
        System.out.print(x);
  }
}
```

A. 50 B. 10 C. Compile error D. Exception

This line declares a variable x of type short and initializes it with the value 10.
Here, x is multiplied by 5. The multiplication operation involves promoting x (which is a short) to an int before performing the multiplication. This is because arithmetic operations in Java, by default, promote smaller data types like byte and short to int. After the multiplication, the result is of type int (since x was promoted to int). Java does not automatically convert int back to short because it might lead to data loss.

Hence, you will get a compile-time error because you are trying to assign an int value to a short variable without explicit casting.

So, the output is **C. Compile error**

Q5.Find the output of the following program.

```
public class Solution{
    public static void main(String[] args){
            byte x = 127;
            x++;
            x++;
            System.out.print(x);
}
    }
```

A. -127 B. 127 C. 129 D. 2

Declares a variable x of type byte and initializes it with the value 127. In Java, the byte data type is an 8-bit signed integer with a range from -128 to 127.
The ++ operator increases the value of x by 1 each time it is executed.
After the first x++, the value of x becomes 128. However, 128 is outside the valid range of the byte type.
In Java, when a byte value exceeds its maximum value (127), it wraps around to the minimum value (-128). This is due to the overflow behavior of signed integers.
So, the sequence of values for x is:

Initially: 127
After first x++: -128
After second x++: -127

So, the output is **A. -127**

Q6. Select the valid statement.

A. char[] ch = new char(5) **B. char[] ch = new char[5]**
C. char[] ch = new char() **D. char[] ch = new char[]**

- A. char[] ch = new char(5)

This statement is **invalid**. In Java, when you create an array, you use square brackets [] to specify the size, not parentheses ().

- B. char[] ch = new char[5]

  This statement is **valid**. It correctly creates a new character array of size 5.

- C. char[] ch = new char()

This statement is **invalid**. You cannot create an array without specifying the size within the square brackets or providing elements within curly braces.

- D. char[] ch = new char[]

This statement is **invalid**. If you are using the empty array initializer, you need to include elements or specify the size, like new char[5] or new char[]{'a', 'b', 'c'}.

So, the output is **B. char[] ch = new char[5]**

Q7. Find the output of the following program.

```
public class Solution{
  public static void main(String[] args){
      int[] x = {120, 200, 016};
      for(int i = 0; i < x.length; i++){
            System.out.print(x[i] + " ");
      }
  }
}
```

A. 120 200 016 B. 120 200 14 C. 120 200 16 D. None

- The first value 120 is a decimal integer.
- The second value 200 is also a decimal integer.
- The third value 016 is an octal (base 8) integer. In Java, numbers that start with 0 are interpreted as octal numbers. 016 in octal is equivalent to 1 * 8 + 6 = 14 in decimal.

the initialization:

- 120 remains 120.
- 200 remains 200.
- 016 is interpreted as octal 016, which is 14 in decimal.

So, the output is **B. 120 200 14**

Q8.When an array is passed to a method, what does the method receive?

A. The reference of the array **B. A copy of the array**
C. Length of the array **D. Copy of first element**

When an array is passed to a method in Java, the method receives the reference to the array, not a copy of the array itself or any other part of it. This means that any changes made to the array elements within the method will affect the original array.

So, the output is **A. The reference of the array**

Q9.Find the value of A[1] after execution of the following program.

```
int[] A = {0,2,4,1,3};
for(int i = 0; i < a.length; i++){
        a[i] = a[(a[i] + 3) % a.length];
 }
```

A. 0 B. 1 C. 2 D. 3

Initial Array : {0,2,4,1,3}
Loop Execution:
For i = 0:

- A[0] = A[(0+3)%5] = A[3%5] = A[3} = 1

For i = 1:

- A[1] = A[(2 + 3) % 5] = A[5 % 5] = A[0] = 1

For i = 2:

- A[2] = A[(4 + 3) % 5] = A[7 % 5] = A[2] = 4

For i = 3:

- A[3] = A[(1 + 3) % 5] = A[4 % 5] = A[4] = 3

For i = 4:

- A[4] = A[(3 + 3) % 5] = A[6 % 5] = A[1] = 1

New array: {1, 1, 4, 3, 1}
After the execution of the loop, the value of A[1] becomes 1.
So, the output is **B. 1**

Q10. When is the object created with a new keyword?

A. At run time B. At compile time C. Depends on the code D. None

In Java, when an object is created using the new keyword, the object is created at runtime.
So, the output is **A. At run time**

Q11. Identify the corrected definition of a package.

A. A package is a collection of editing tools

B. A package is a collection of classes
C. A package is a collection of classes and interfaces
D. A package is a collection of interfaces

Packages in Java are used to group related classes and interfaces together. This helps in organizing the code in a modular fashion, making it easier to manage and use. Additionally, packages provide access protection and namespace management.

So, the output is **C. A package is a collection of classes and interfaces**

Q12.Identify the keyword among the following that makes a variable belong to a class,rather than being defined for each instance of the class.

A. final B. static C. volatile D. abstract

In Java, the static keyword is used to declare class-level variables.
static is the keyword that makes a variable belong to a class, rather than being defined for each instance of the class.

So, the output is **B. static**

Q13.Identify what can directly access and change the value of the variable res.

```
Package com.mypackage;
Public class Solution{
   Private int res = 100;
}
```

A. Any class **B. Only Solution class**
C. Any class that extends Solution **D. None**

The visibility modifier private used before the declaration of the variable res in the Solution class restricts its access and modification to only within the Solution class itself.

So, the output is **B. Only Solution class**

Q14.In which of the following is the toString() method defined?

A. java.lang.Object B. java.lang.String C. java.lang.util D. None

The toString() method is defined in the java.lang.Object class.

So, the output is **A. java.lang.Object**

Q15.Identify the output of the following program.

```
String str = “abcde”;
System.out.println(str.substring(1, 3));
```

A. abc B. bc C. bcd D. cd

The substring(int beginIndex, int endIndex) method is called on the str object. This method returns a new string that is a substring of the original string, starting from the beginIndex (inclusive) to the endIndex (exclusive). In this case, beginIndex is 1 and endIndex is 3
The substring method extracts characters from index 1 to index 2 which corresponds to the characters 'b' and 'c'.

So, the output is **B. bc**

Q16.Identify the output of the following program.

```
String str = “Hellow”;
System.out.println(str.indexOf(‘t));
```

A. 0 B. 1 C. true D. -1

The indexOf(char ch) method is called on the str object. This method returns the index within the string of the first occurrence of the specified character ch. In this case, the specified character is 't'.
Since there is no 't' character in the string "Hellow", the indexOf method returns -1.

So, the output is **D. -1**

Q17. Identify the output of the following program.

```
public class Test{
   public static void main(String args[]){
         String str1 = “one”;
         String str2 = “two”;
         System.out.println(str1.concat(str2));
    }
}
```

A. one B. two C. onetwo D. twoone

The concat method is called on str1 with str2 as an argument. This method concatenates the specified string to the end of the invoking string.

So, the output is **C. onetwo**

Q18. How many objects will be created in the following?

```
String a = new String("FlipRobo");
String b = new String("FlipRobo");
String c = "FlipRobo";
String d = "FlipRobo";
```

A. 2 B. 3 C. 4 D. None

String a = new String("FlipRobo");

- This line explicitly creates a new String object with the value "FlipRobo" using the new keyword, so one object is created.

String b = new String("FlipRobo");

- Similarly, this line also explicitly creates another new String object with the value "FlipRobo" using the new keyword, so another object is created.

String c = "FlipRobo";

- This line creates a String object with the value "FlipRobo", but since the string literal "FlipRobo" already exists in the string pool, it doesn't create a new object. Instead, it reuses the existing object from the string pool. So, no new object is created here.

String d = "FlipRobo";

- This line also creates a String object with the value "FlipRobo", and it also reuses the existing object from the string pool. Again, no new object is created here.

There are 2 objects created. So, the output is **A. 2**

Q19.Find the output of the following code.

```
int ++a = 100;
System.out.println(++a);
```

A. 101 B. Compile error as ++a is not valid identifier C. 100 D. None

The code you provided has a syntax error. The prefix increment operator ++ cannot be directly applied to a variable during its declaration. It should be used separately as a statement or within an expression.

So the output is **B. Compile error as ++a is not valid identifier**

Q20.Find the output of the following code.

```
if(1 + 1 + 1 + 1 + 1 == 5){
System.out.print("TRUE");
```

```
}
else{
System.out.print("FALSE");
}
```
A. TRUE B. FALSE C. Compile error D. None

The given code checks whether the sum of five 1's is equal to 5. Since 1+1+1+1+1 equals 5, the condition evaluates to true, and the statement inside the if block is executed

So, the output is .**A. TRUE**

Q21.Find the output of the following code.

```
Public class Solution{
    Public static void main(String args[]){
        Int x = 5;
        x * = (3 + 7);
        System.out.println(x);
```

A. 50 B. 22 C. 10 D. None

There are two issues in the given code:

1. Java is case-sensitive, so Int should be int.
2. There should not be a space between * and = in the compound assignment operator *=.

Correct code is :

```
public class Solution {
  public static void main(String[] args) {
    int x = 5;
    x *= (3 + 7);
    System.out.println(x);
}
 }
```

The compound assignment operator *= is used to multiply the value of x by the result of (3 + 7), which is 10.
So, x becomes 5 * 10, which equals 50.

So, the output is **A. 50**

Q22.Identify the return type of a method that does not return any value.

A. int B. void C. double D. None

In Java, the void keyword is used as the return type of a method to indicate that the method does not return any value. When a method's return type is void, it means that the method performs some operations but does not produce a result to be used in expressions.

So, the output is **B. void**

Q23.Output of Math.floor(3.6)?
A. 3 B. 3.0 C. 4 D. 4.0

The Math.floor() method in Java returns the largest (closest to positive infinity) double value that is less than or equal to the argument and is equal to a mathematical integer.

So, when you use Math.floor(3.6), it returns 3.0, because 3.0 is the largest double value that is less than or equal to 3.6 and is equal to a mathematical integer.

So, the output is **B. 3.0**

Q24. Identify the modifier which cannot be used for constructor.
A. public B. protected C. private D. static

Constructors are automatically called when an object of the class is created, and they are used to initialize the object's state.
Declaring a constructor as static would not make sense because static methods or variables belong to the class itself, not to individual instances of the class. Constructors, on the other hand, are used to initialize object-specific state.

So, the output is **D. static**

Q25. What are the variables declared in a class for the use of all methods of the class called?
A. Object B. Instance variables C. Reference variable D. None

Instance variables are declared within a class but outside of any method, constructor, or block.
They are associated with objects (instances) of the class and are accessible to all methods of the class.
Each instance of the class (i.e., each object) has its own copy of instance variables.

So, the output is **B. Instance variables**

Q26. Find the output of the following code.

```
Public class Solution{
    Public static void main(String args[]){
        Int i;
        for(i = 1; i < 6; i++){
            if(i > 3) continue;
        } System.out.println(i);
    }
}
```

A. 3 B. 4 C. 5 D. 6

A for loop is used to iterate from 1 to 5.

- Inside the loop, there's an if statement checking if i is greater than 3. If it is, the continue statement is executed, which skips the remaining code inside the loop and continues with the next iteration.
- When i becomes 4, the condition i > 3 is true, so the continue statement is executed. This means that the value of i will not be incremented beyond 3 in this loop.
- After the loop, the value of i is printed.

Since the loop exits when i becomes 4, the value of i remains 4 when the loop finishes.

So, the output is **B. 4**

Q27. Identify the infinite loop.

A. for(; ;) B. for(int i = 0; i < 1; i--) C. for(int i = 0; ;i++) D. All of the above

A. for(;;): This is a classic example of an infinite loop. It has no initialization, no condition, and no increment/decrement. It will keep iterating indefinitely.

B. for(int i=0;i<1;i--): This loop will not be an infinite loop because i is initialized to 0, and i will always be less than 1 initially. However, i-- will keep decrementing i, and eventually, i will underflow, causing the loop to exit. So, this option does not create an infinite loop.

C. for(int i=0;;i++): This loop will be an infinite loop because there is no termination condition (i will keep incrementing without any check). It will continue indefinitely.

So, the output is **C. for(int i = 0; ;i++)**

Q28.Exception created by try block is caught in which block

A. catch B. throw C. final D. none

Exceptions created by a try block are caught in the catch block.
So, the output is **A. catch**

Q29.Which of the following exception is thrown when divided by zero statement is executed? A. NullPointerException B. NumberFormatException
C. ArithmeticException D. None

In Java, dividing an integer or floating-point number by zero results in an ArithmeticException.
This exception is thrown at runtime when an attempt is made to perform an arithmetic operation that is not mathematically defined, such as division by zero.
So, the output is **C. ArithmeticException**

Q30.Where is System class defined?
A. java.lang.package B. java.util.package C. java.io.package D. None

The System class is a part of the Java standard library and is used to access system resources, such as standard input, standard output, and error output streams.
Since it is a fundamental part of the Java language and is commonly used in Java programs, it is included in the java.lang package, which is automatically imported into every Java program.

So, the output is **A. java.lang.package**