1. What is JDK? JRE? JVM?

   JDK stands for Java Development Kit. It is a software development environment used to develop Java applications and applets.

   JRE stands for Java Runtime Environment. It is the implementation of JVM (Java Virtual Machine) and it is specially designed to provide an environment to execute Java programs.

   JVM stands for Java virtual machine and it is a virtual machine that enables a computer to run Java programs as well as programs written in other languages that are also compiled to Java bytecode.

2. What is java compiler?

   A Java compiler is a program that takes the text file work of a developer and compiles it into a platform-independent Java file.

3. Why is java platform independent?

   Java is platform independent because the Java compiler converts the source code to bytecode, which is Intermediate Language. Bytecode can be executed on any platform (OS) using JVM(Java Virtual Machine).

4. What is IDE? Why is it important for developers?

   An IDE, or Integrated Development Environment, enables programmers to consolidate the different aspects of writing a computer program. IDEs increase programmer productivity by combining common activities of writing software into a single application: editing source code, building executables, and debugging.

5. Is java case sensitive?

   Java is case sensitive

6. What do the following key words do?static, final, public, private, void, null, package, Class, new

   -keyword **static** means that the particular member belongs to a type itself, rather than to an instance of that type. This means we'll create only one instance of that static member that is shared across all instances of the class.

-The final keyword is a non-access modifier used for classes, attributes and methods, which makes them non-changeable (impossible to inherit or override).

-public is a Java keyword which declares a member's access as public. Public members are visible to all other classes.

-private is a Java keyword which declares a member's access as private. That is, the member is only visible within the class, not from any other class

-Void is used at method declaration and definition to specify that the method does not return any type, the method returns void

-null is a literal similar to true and false in Java. These are not keywords because these are the values of something

-package is a Java keyword. It declares a 'name space' for the Java class. It must be put at the top of the Java file, it should be the first Java statement line.

-class is a Java keyword which begins the declaration and definition of a class.

-new is a Java keyword. It creates a Java object and allocates memory for it on the heap. new is also used for array creation, as arrays are also objects.

7. What is primitive type and reference type?

   The basic difference is that primitive variables store the actual values, whereas reference variables store the addresses of the objects they refer to.

8. Is parameter passed by value or reference?

   Java is always pass-by-value.

9. What is the output: System.out.println(1 > 0 : “A”:”B”);

   A

10. How to define constants in java?

    To turn an ordinary variable into a constant, you have to use the keyword "final."

11. What is String? Is it primitive type?

    A String in Java is actually a non-primitive data type, because it refers to an object.

12. How to check if a String is representing a number?

    Perhaps the easiest and the most reliable way to check whether a String is numeric or not is by parsing it using Java's built-in methods like ParseInt

13.

```
import java.util.*;

public class lmsHW1 {

    public static void main(String[] args) {

        Scanner keyboard = new Scanner(System.in);



        while(true) {

            System.out.println("enter input");
            String x = keyboard.nextLine();

            if (x.equals("q") || x.equals("Q")) {
                System.out.println("Quit program");
                break;
            }

            int xInt = Integer.parseInt(x);

            if (xInt > 0) {
                for (int i = 0; i < xInt; i++) {
                    doSomething(xInt);
                }
            }

            else{
                System.out.println("Error. Please Enter valid input");
            }


        }
```

```
    }

    public static void doSomething(int x){
        int ans = 0;
        for(int i = 1; i<=x; i++){
            ans+=i;
        }
        System.out.println(ans);
    }
}
```

14.

```
public static void MergeIntArr() {

        int[] array1 = {0,1,2,3,4};
        int[] array2 = {5,6,7,8,9};

        int[] mergeArr = new int[array1.length + array2.length];
        System.arraycopy(array1, 0, mergeArr, 0, array1.length);
        System.arraycopy(array2, 0, mergeArr, array1.length, array2.length);

    }
```

15.

```
public static void getSecondHighest() {

int[] array = {3,52,1295,2,5,1020,48,2222};
int highest = Integer.MIN_VALUE;
int secondHighest = Integer.MIN_VALUE;


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

            if (array[i] > highest) {
                    secondHighest = highest;
                    highest = array[i];
            }
            else if (array[i] > secondHighest)
                    secondHighest = array[i];
            }
```

```
}
```