## 2. Explain the difference between method declaration and method body.

Method Declaration defines the method's name, return type, and parameters (the method's "header").
Example:

```
public static void greet(String name)
```

- This part tells the compiler what the method is and what it needs.

Method Body contains the statements or code that run when the method is called.
Example:

```
{
   System.out.println("Hello " + name);
}
```

- The body is everything inside the { }.

## 3. What type of keyword is used to change the access level of a method?

- Access modifiers are the keywords used to change the access level of a method.

Examples:

- public
- private
- protected
- (default — no modifier)

## 4. What is another word used for describing the access level of a method?

- Another word is visibility.
  It describes how visible or accessible a method is to other classes.

## 5. Explain the scope of each of the variables in the code below:

```
public class ScopeExample {
   public static void main(String[] args) {
      int var1;
      for (int var2 = 0; var2 < 5; var2++) {
         method1();
      }
   }
```

```
    public static void method1() {
        int var3;
        for (int var4 = 0; var4 < 2; var4++) {
            var3 += 1;
        }
    }
}
```

Variable Scopes:

- var1 → Declared in the main method.
  ➢ Scope: Entire main method.
  It can be accessed anywhere inside main, but not outside it.

- var2 → Declared inside the for loop in main.
  ➢ Scope: Inside the for loop only.
  It cannot be used after the loop ends.

- var3 → Declared inside method1.
  ➢ Scope: Entire method1 method.
  It cannot be used in main.

- var4 → Declared inside the for loop in method1.
  ➢ Scope: Inside that for loop only.

(Note: In this example, var3 += 1; will cause an error because var3 is never initialized before being used.)

## 6. Write a method declaration for each of the following descriptions:

a) A class method named getVowels that can be called by any other method, requires a String parameter, and returns an integer value.
public static int getVowels(String text)

b) A class method named extractDigit that can be called by any other method, requires an integer parameter, and returns an integer value.
public static int extractDigit(int number)

c) A class method named insertString that can be called by any other method, requires a String parameter and an integer parameter, and returns a String value.
public static String insertString(String text, int position)

## 7.

a) How does the compiler distinguish one method from another?

- The compiler uses the method signature, which includes the method name and the parameter list (types, number, and order of parameters).

b) Can two methods in the same class have the same name? Explain.

- Yes, this is called method overloading.
  It is allowed as long as each method has a different parameter list (different number or types of parameters).
  The return type alone does not distinguish methods.

a) Breaking a task down into methods is called procedural abstraction.

## 11. True/False Questions

True
Breaking a large task into smaller, reusable methods is known as procedural abstraction.

b) A method call consists of the method declaration in an assignment statement.

False
A method call is when you invoke a method (e.g., methodName();).
A method declaration is when you define the method (e.g., public void methodName() { }).

c) A void method must return a value.

False
A void method does not return a value. It just performs an action.

d) An access modifier declares the return type of a method.

False
An access modifier (like public, private) controls visibility, not return type.
The return type is declared after the access modifier.

e) The keyword static declares a method is a class method.

True
A static method belongs to the class, not to a specific object. It can be called using the class name.

f) Method parameters are enclosed by braces ({}).

False
Method parameters are enclosed by parentheses ( ), not braces { }.

g) Local variables can be used by any method in a class.

False
Local variables only exist inside the method where they’re declared. Other methods can’t access them.

h) The value of an argument passed to a method can be changed in an assignment statement in the method.

True (with explanation)
If the argument is a primitive type (like int), a copy is passed — changing it won’t affect the original.
If it’s an object, the method can modify the object’s contents.

i) Method overloading means that an application contains more than 10 methods.

False
Method overloading means having multiple methods with the same name but different parameter lists — not a specific number of methods.

j) The return statement is used to send a value back to the calling statement.

True
The return statement sends a value from a method back to the part of the program that called it.

k) The precondition of a method states the data types of the method’s parameters.

False
A precondition describes what must be true before a method runs (e.g., valid input values).
The data types of parameters are part of the method’s declaration, not its precondition.

l) The postcondition of a method describes the way the method accomplishes its task.

False
A postcondition describes what will be true after the method runs (the result or outcome), not how it accomplishes the task.