Assignment 2

1. snippet 1

Demo2.java | Voter.java | Grade.java | Main.java | Calculator.java | DiscountCal.java

```
1  public class Main {
2      public void main(String[] args) {
3          System.out.println("Hello, World!");
4      }
5  }
6
```

Problem:

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Error: Main method is not static in class Main, please define the main method as:
   public static void main(String[] args)
```

Solution:

Static keyword was missing.

The static is a keyword which we use in the main() method to define it as static. There is no object of the class available at the time of starting java runtime, and because of that, we have to define the main() method as static. By doing that, JVM can load the class into the main memory and call the main() method.

```
1  public class Main {
2      public static void main(String[] args) {
3          System.out.println("Hello, World!");
4      }
5  }
6
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Hello, World!
```

Snippet 2:

```
1  public class Main {
2      static void main(String[] args) {
3      System.out.println("Hello, World!");
4      }
5  }
6
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java

C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Error: Main method not found in class Main, please define the main method as:
   public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
```

Sol: Public keyword was missing.

It is not so complicated to understand. It is an access modifier of the main() method. We create main() method with public access specifier to execute it by any program. So, it is required to define main() method public

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Hello, World!
```

Snippet 3:

```
public class Main {
    public static int main(String[] args) {
        System.out.println("Hello, World!");
        return 0;
    }
```

Sol: The keyword void tells Java that the main method won't return a value. Other methods in other classes can receive and return values/variables, but main can't return anything.

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Error: Main method must return a value of type void in class Main, please
define the main method as:
   public static void main(String[] args)
```

Snippet 4:

```
public class Main {
    public static void main() {
        System.out.println("Hello, World!");
    }
}
```

Problem: runtime error

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Error: Main method not found in class Main, please define the main method as:
   public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
```

Sol:

Add String[] args

String args in Java is used to pass command-line arguments to a Java program. It allows external data to be passed into the program when it starts, making it more versatile and adaptable to various inputs.

```java
public class Main {
 public static void main(String[] args) {
  System.out.println("Hello, World!");
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Hello, World!
```

Snippet 5:

```java
public class Main {
 public static void main(String[] args) {
  System.out.println("Main method with String[] args");
 }
 public static void main(int[] args) {
  System.out.println("Overloaded main method with int[] args");
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java

C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Main method with String[] args
```

Sol: we can have more than one class in one java file but only the main method will be executed.

Snippet 6:

```java
public class Main {
 public static void main(String[] args) {
  int x = y + 10;
  System.out.println(x);
 }
}
```

Problem: variable y is not declared.

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:3: error: cannot find symbol
 int x = y + 10;
         ^
  symbol:   variable y
  location: class Main
1 error
```

Solution: The declaration of variable should be always done before using that variable as tell the compiler what is the meaning and purpose of that word or else it will throw an error. So the solution for this snippet can be solved by declaring and initializing y = 5.

```java
public class Main {
 public static void main(String[] args) {
 int y = 5;
 int x = y + 10;
 System.out.println(x);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
15
```

Snippet 7:

```java
public class Main {
 public static void main(String[] args) {
 int x = "Hello";
 System.out.println(x);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:3: error: incompatible types: String cannot be converted to int
 int x = "Hello";
         ^
1 error
```

Sol: Type safety means preventing type errors. Type error occurs when data type of one type is assigned to other type unknowingly and we get undesirable results.

Change the datatype of the x it can be a String.

```java
1  public class Main {
2   public static void main(String[] args) {
3   String x = "Hello";
4   System.out.println(x);
5   }
6  }
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Hello
```

Snippet 8:

```java
1  public class Main {
2   public static void main(String[] args) {
3   System.out.println("Hello, World!"
4   }
5  }
```

Problem: Its is missing one ‘)’ bracket which will cause compilation error.

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:3: error: ')' expected
 System.out.println("Hello, World!"
                                   ^
1 error
```

Sol:

Add ‘)’ and ‘;’ at line3

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Hello, World!
```

Snippet 9:

```java
public class Main {
 public static void main(String[] args) {
  int class = 10;
  System.out.println(class);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:3: error: not a statement
 int class = 10;
 ^
Main.java:3: error: ';' expected
 int class = 10;
    ^
Main.java:3: error: <identifier> expected
 int class = 10;
         ^
Main.java:4: error: <identifier> expected
 System.out.println(class);
                    ^
Main.java:4: error: illegal start of type
 System.out.println(class);
                    ^
Main.java:4: error: <identifier> expected
 System.out.println(class);
                         ^
Main.java:6: error: reached end of file while parsing
}
 ^
7 errors
```

Solution;

Identifiers are the reserved keywords which should not be used as a variable,class name as it makes it harder to differenciate between keyword and variable.

```java
public class Main {
 public static void main(String[] args) {
  int a = 10;
  System.out.println(a);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
10
```

Snippet 10:

```java
public class Main {
 public void display() {
  System.out.println("No parameters");
 }
 public void display(int num) {
  System.out.println("With parameter: " + num);
 }
 public static void main(String[] args) {
  display();
  display(5);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:9: error: non-static method display() cannot be referenced from a static context
 display();
 ^
Main.java:10: error: non-static method display(int) cannot be referenced from a static context
 display(5);
 ^
2 errors
```

Sol: Static Was missing. Method Loading is allowed in java.

```java
public class Main {
 public static void display() {
  System.out.println("No parameters");
 }
 public static void display(int num) {
  System.out.println("With parameter: " + num);
 }
 public static void main(String[] args) {
  display();
  display(5);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
No parameters
With parameter: 5
```

Snippet 11:

```java
public class Main {
 public static void main(String[] args) {
  int[] arr = {1, 2, 3};
  System.out.println(arr[5]);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java

C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 3
        at Main.main(Main.java:4)
```

Sol: The arr was asked to print what was not there so we make it so that it asks for the values/indexes according to the array size.

```java
public class Main {
 public static void main(String[] args) {
  int[] arr = {1, 2, 3};
  System.out.println(arr[1]);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
2
```

Snippet 12:

```java
public class Main {
 public static void main(String[] args) {
  while (true) {
   System.out.println("Infinite Loop");
  }
  }
 }
```

```
Infinite Loop
Infinite Loop
Infinite Loop
Infinite Loop
Infinite Loop
Infinite Loop
Infinite Loop
Infinite Loop
Infinite Loop
Infinite Loop
Infinite Loop
Infinite Loop
^C
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>
```

Sol:

In the above snippet the program leads to a infinite loop as it does not have a terminal statement to terminate the loop.

To avoid infinite while loops in Java, ensure that the loop's condition eventually becomes false. Here are a few strategies:

- Update Loop Control Variable: Increment/Decrement: If your loop relies on a counter, make sure it's incremented or decremented within the loop body so that it eventually meets the exit condition.
- Use a Boolean FlagSet a flag: Use a boolean variable to control the loop's execution. Change the flag's value within the loop to exit when necessary.
- Check for Exit Condition at the BeginningPlace the condition in the while statement: Ensure that the condition is evaluated at the beginning of each iteration. If the condition is false from the start, the loop won't execute at all.
- Break StatementExit the loop prematurely: Use the break statement to exit the loop when a specific condition is met.
- Limit the Number of IterationsSet a maximum number of iterations: If appropriate, add a counter to limit the number of times the loop executes.

Snippet 13:

```java
public class Main {
 public static void main(String[] args) {
  String str = null;
  System.out.println(str.length());
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java

C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Exception in thread "main" java.lang.NullPointerException: Cannot invoke "String.length()" because "<local1>" is null
        at Main.main(Main.java:4)
```

Sol:

In this snippet the String i.e str is given null so me assign as string “hello” which has a length of 5.

```java
public class Main {
 public static void main(String[] args) {
  String str = "hello";
  System.out.println(str.length());
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
5
```

Snippet 14:

```java
public class Main {
 public static void main(String[] args) {
 double num = "Hello";
 System.out.println(num);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:3: error: incompatible types: String cannot be converted to double
 double num = "Hello";
              ^
1 error
```

Sol:

In this the datatype of num is wrong as it should be a String Datatype.

```java
public class Main {
 public static void main(String[] args) {
 String num = "Hello";
 System.out.println(num);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Hello
```

Snippet no 15

```java
public class Main {
 public static void main(String[] args) {
 int num1 = 10;
 double num2 = 5.5;
 int result = num1 + num2;
 System.out.println(result);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:5: error: incompatible types: possible lossy conversion from double to int
 int result = num1 + num2;
                   ^
```

Sol:

In this the result variable is declared as a int while the variable as different datatypes i.e num1 is a int while num2 is double so to may it work we have to do typecasting.

```java
public class Main {
 public static void main(String[] args) {
 int num1 = 10;
 double num2 = 5.5;
 double result = (double)num1 + num2;
 System.out.println(result);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
15.5
```

Snippet no 16

```java
public class Main {
 public static void main(String[] args) {
  int num = 10;
  double result = num / 4;
  System.out.println(result);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
2.0
```

Sol:

As expected the result is in double as double is considered higher then int datatype so it automatically follows.

Snippet no.17

```java
public class Main {
 public static void main(String[] args) {
  int a = 10;
  int b = 5;
  int result = a ** b;
  System.out.println(result);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:5: error: illegal start of expression
 int result = a ** b;
                ^
1 error
```

Sol: Java does not have a built-in exponentiation operator.

Snippet 18

```java
public class Main {
 public static void main(String[] args) {
  int a = 10;
  int b = 5;
  int result = a + b * 2;
  System.out.println(result);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
20
```

Sol: The output of the program is 20. The precedence of * is higher than that of the + so the first calculation to take place is b * 2 = 10. After that a will be added making it 20.

Snippet 19

```java
public class Main {
 public static void main(String[] args) {
  int a = 10;
  int b = 0;
  int result = a / b;
  System.out.println(result);
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at Main.main(Main.java:5)
```

Sol: The runtime error in above snippet is ArithmeticException because division by zero with any number is undefined.

Snippet 20:

```java
public class Main {
 public static void main(String[] args) {
  System.out.println("Hello, World")
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:3: error: ';' expected
 System.out.println("Hello, World")
                                   ^
1 error
```

Sol: The above program is missing a ';' semicolon. In Java, semicolons are used to separate statements and indicate the end of one logical entity. Without semicolons, it would be difficult for the computer to determine where one statement ends and another begins, which could lead to errors in the code.

```java
public class Main {
 public static void main(String[] args) {
  System.out.println("Hello, World");
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Main
Hello, World
```

Snippet 21:

```java
public class Main {
 public static void main(String[] args) {
  System.out.println("Hello, World!");
  // Missing closing brace here
 }

```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:5: error: reached end of file while parsing
}
 ^
1 error
```

Sol: In Java, the error message "Reached End of File While Parsing" can occur when a closing curly bracket for a block of code is missing.

Snippet 22:

```java
public class Main {
 public static void main(String[] args) {
 static void displayMessage() {
  System.out.println("Message");
 }
 }
}

```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Main.java
Main.java:3: error: illegal start of expression
 static void displayMessage() {
 ^
Main.java:7: error: class, interface, or enum expected
}
^
2 errors
```

Sol: java does not support directly declaring a method inside another method.

Snippet 23:

```java
public class Confusion {
 public static void main(String[] args) {
 int value = 2;
 switch(value) {
 case 1:
 System.out.println("Value is 1");
 case 2:
 System.out.println("Value is 2");
 case 3:
 System.out.println("Value is 3");
 default:
 System.out.println("Default case");
 }
 }
}
```

Sol: To avoid printing default after “Value is 2” we can write a terminate statement i.e break after 2

```java
public class Confusion {
 public static void main(String[] args) {
 int value = 2;
 switch(value) {
 case 1:
 System.out.println("Value is 1");
 case 2:
 System.out.println("Value is 2");
 break;
 case 3:
 System.out.println("Value is 3");
 default:
 System.out.println("Default case");
 }
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Confusion
Value is 2
```

Snippet 24

```java
public class MissingBreakCase {
 public static void main(String[] args) {
  int level = 1;
  switch(level) {
  case 1:
  System.out.println("Level 1");
  case 2:
  System.out.println("Level 2");
  case 3:
  System.out.println("Level 3");
  default:
  System.out.println("Unknown level");
  }
  }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java MissingBreakCase
Level 1
Level 2
Level 3
Unknown level
```

Sol: To avoid printing default after “"Level 1", "Level 2", "Level 3", and "Unknown level"” we can write a terminate statement i.e break after every case.

```java
public class MissingBreakCase {
 public static void main(String[] args) {
  int level = 1;
  switch(level) {
  case 1:
  System.out.println("Level 1");
  break;
  case 2:
  System.out.println("Level 2");
  break;
  case 3:
  System.out.println("Level 3");
  break;
  default:
  System.out.println("Unknown level");
  break;
  }
  }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java MissingBreakCase
Level 1
```

Snippet no.25

```java
public class Switch {
 public static void main(String[] args) {
  double score = 85.0;
  switch(score) {
  case 100:
  System.out.println("Perfect score!");
  break;
  case 85:
  System.out.println("Great job!");
  break;
  default:
  System.out.println("Keep trying!");
  }
  }
 }
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Switch.java
Switch.java:4: error: incompatible types: possible lossy conversion from double to int
 switch(score) {
        ^
1 error
```

Sol: The code does not work as switch does not support double datatype along with float. We can use type casting on course which is entering in the switch.

```java
public class Switch {
 public static void main(String[] args) {
  double score = 85.0;
  switch((int)score) {
  case 100:
  System.out.println("Perfect score!");
  break;
  case 85:
  System.out.println("Great job!");
  break;
  default:
  System.out.println("Keep trying!");
  }
  }
 }
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>java Switch
Great job!
```

Snippet no.26

```java
public class Switch {
 public static void main(String[] args) {
  int number = 5;
 switch(number) {
  case 5:
  System.out.println("Number is 5");
  break;
  case 5:
  System.out.println("This is another case 5");
  break;
  default:
  System.out.println("This is the default case");
 }
 }
}
```

```
C:\Users\ASUS\Desktop\Aug24CDAC\day2cdacpre>javac Switch.java
Switch.java:8: error: duplicate case label
 case 5:
 ^
1 error
```

Sol: if two cases in a switch statement have the same label, the second case will never be executed**.**