Part 1: Vocabulary

1. **While Loop**: allows the program to repeat a set of instructions while a condition is true. If the condition is false or when the condition becomes false, it skips the block of code to be repeated and runs the rest of the code.

   Syntax:

   ```
   while <condition> {
         // block of codes to be repeated
         // condition must be changed to avoid infinite loop
   }
   // rest of code
   ```

   Example: Prints out the sum of the first 3 consecutive integers

   ```
   int term = 1, sum = 0;
   while (term <= 3) {     // term = 1            2           3           4 (Stop)
         sum = sum + term; // sum = 0 + 1 = 1     1 + 2 = 3   3 + 3 = 6
         term = term + 1;  // term = 2            3           4
   }
   System.out.println(sum);      // prints sum = 6
   ```

2. …

Part 2: Key Ideas

Lesson 12: While and Do-While Loop

- While loops are same as for loops except initializing and step statements are not included in while loops.
- Do-While loops are exactly the same as while loops except their control expression is at the bottom of the loop. In other words, the program runs the block of codes inside the do-while loop and then checks the condition.

Lesson 13: …

Part 3: Exercise Problems / Quiz Corrections

Lesson 13, Problem 8: What does the following code do?

```
char c;
for (int j = 97; j <= 122; j++) {
c = (char)(j – 32);
      System.out.print(c);
}
```

Original Response: It prints out abcdef…xyz. ← Thinking error.

Correct Response: It print out ABCDEF…XYZ instead. I thought (char)65 printed ‘a’ but it actually prints ‘A’; ASCII prints uppercase letters before lowercase.

Quiz 3,Problem 1: …

Part 4: Project Problems

Lesson 7: What's My Name?

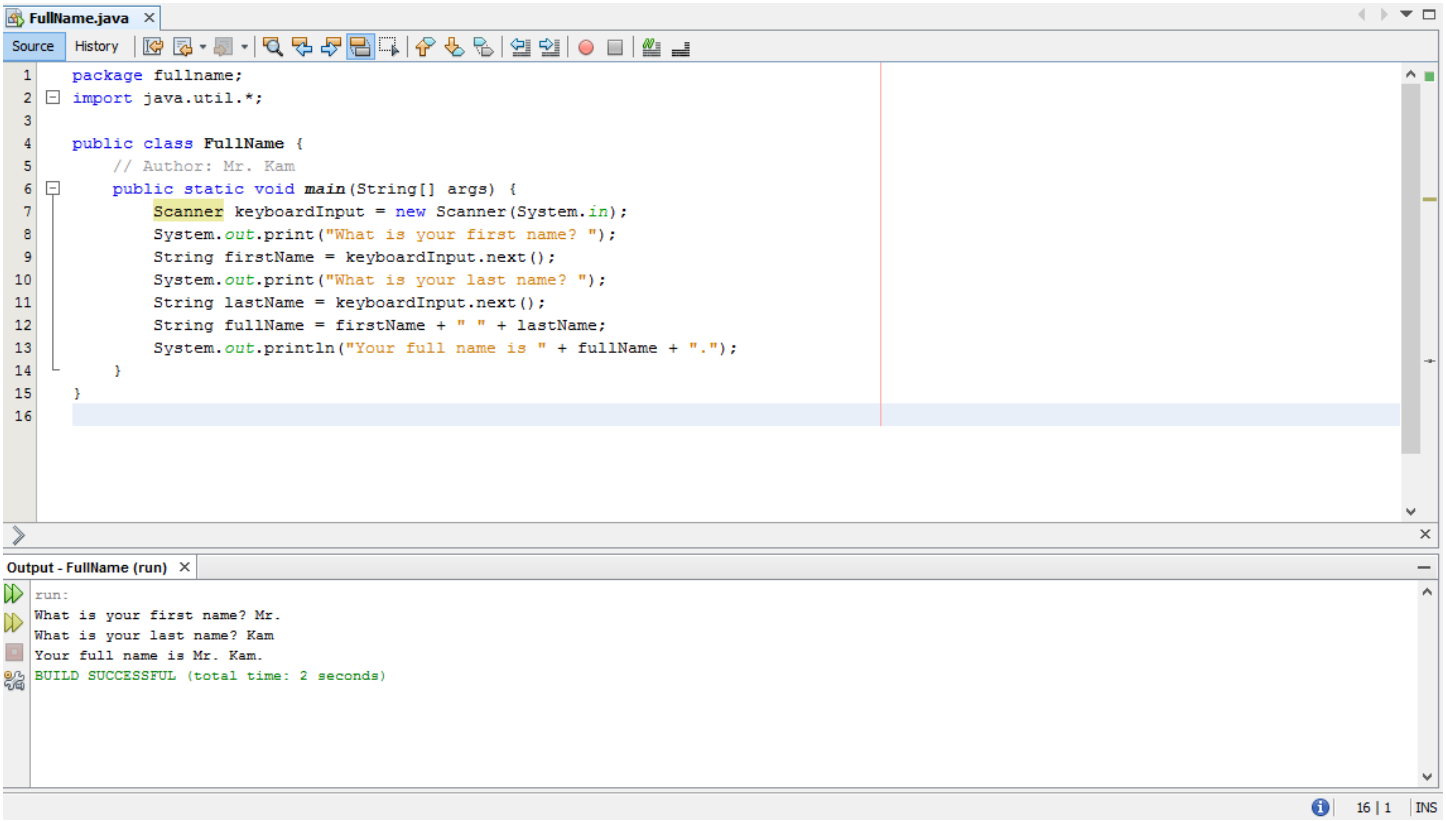
Problem: For the keyboard enter your first and then your last name, each with its own prompt. Store each in a separate String and then concatenate them to show your first name. Call both the project and class FullName.

Solution:

```
package fullname;
import java.util.*;

public class FullName {
    public static void main(String[] args) {
        Scanner keyboardInput = new Scanner(System.in);
        System.out.print("What is your first name? ");
        String firstName = keyboardInput.next();
        System.out.print("What is your last name? ");
        String lastName = keyboardInput.next();
        String fullName = firstName + " " + lastName;
        System.out.println("Your full name is " + fullName + ".");
    }
}
```

Screenshot:



Part 5: Peer Problem

Problem: Write a program in which the user enters a valid DNA sequence and it prints the complementary sequence. A valid DNA sequence only contains A, T, C or G and its complementary bases are T, A, G and C respectively (i.e. A ⇔ T and C ⇔ G). For example, if the DNA sequence is "TAGCCGAT", then its complementary sequence is "ATCGGCTA".

Solution:

```
package dna;
import java.util.*;

public class DNA {
    public static void main(String[] args) {
        Scanner keyboardInput = new Scanner(System.in);
        System.out.print("Enter a valid DNA sequence: ");
        String dnaInput = keyboardInput.next();
        String dnaOutput = "";
        String basePair;
        for (int k = 0; k < dnaInput.length(); k++) {
            if (dnaInput.charAt(k) == 'A')
                basePair = "T";
            else if (dnaInput.charAt(k) == 'T')
                basePair = "A";
            else if (dnaInput.charAt(k) == 'C')
                basePair = "G";
            else if (dnaInput.charAt(k) == 'G')
                basePair = "C";
            else
                break; // invalid sequence
            dnaOutput = dnaOutput + basePair;
        }
        if (dnaInput.length() == dnaOutput.length())
            System.out.println("Its complementary sequence: " + dnaOutput);
        else
            System.out.println("The DNA sequence is invalid.");
    }
}
```

Evaluation: 4 marks

Expectations/Criteria:

- Code finds and prints the complementary sequence.
- Code considers invalid DNA sequence.
- Code has little or no syntax errors.
- Code is clean and easy to read.

Grading Rubric:

- Fully Meeting (4): no logical error; all criteria are satisfied.
- Mostly Meeting (3): few minor logical errors; at least one criteria is not satisfied.
- Minimally Meeting (2): some minor logical errors; at least two criteria is not satisfied.
- Developing (1): major logical errors; at least three criteria is not satisfied.
- Missing (0): blank or non-sensible solution.