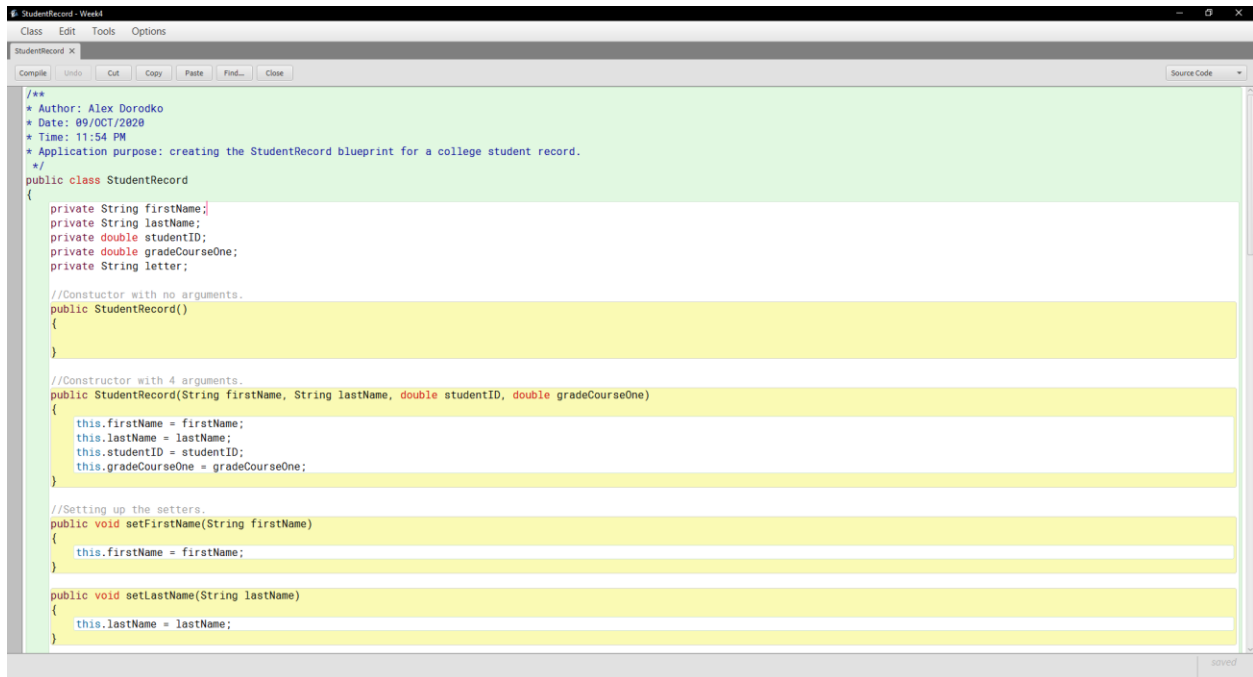


# StudentRecord.java

## Part 1



```
StudentRecord - Week4
Class Edit Tools Options
StudentRecord X
Compile Undo Cut Copy Paste Find... Close Source Code

/**
 * Author: Alex Dorodko
 * Date: 09/OCT/2020
 * Time: 11:54 PM
 * Application purpose: creating the StudentRecord blueprint for a college student record.
 */
public class StudentRecord
{
    private String firstName;
    private String lastName;
    private double studentID;
    private double gradeCourseOne;
    private String letter;

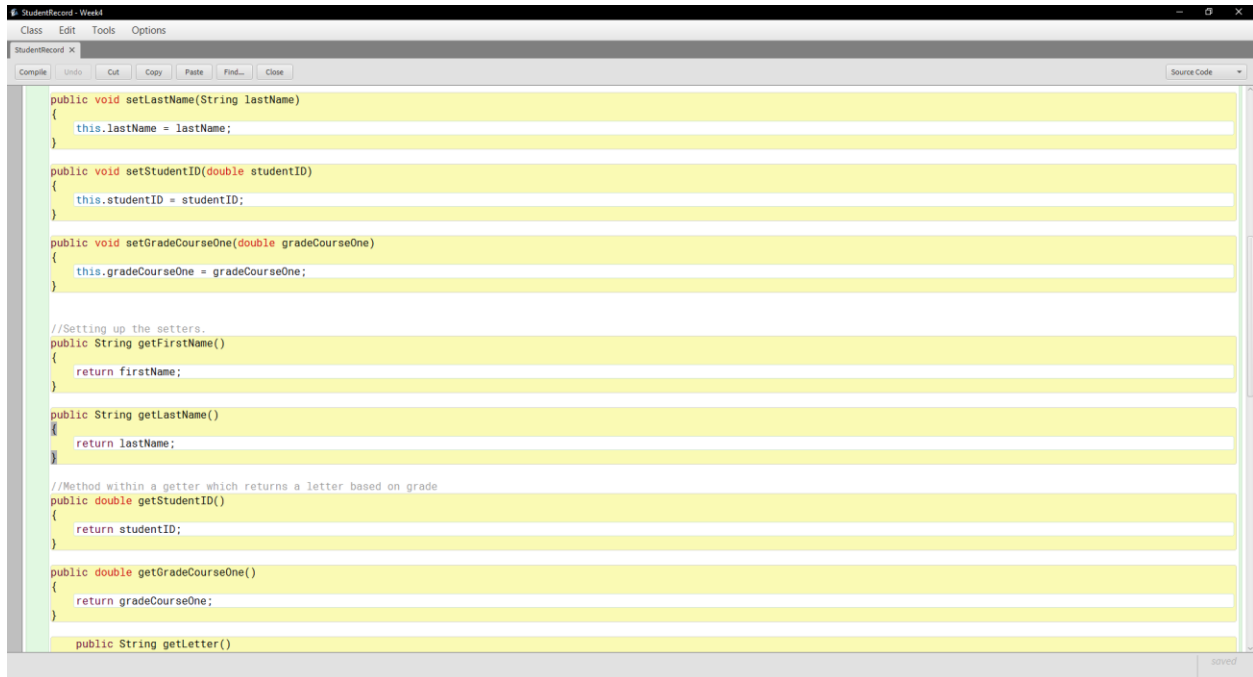
    //Constructor with no arguments.
    public StudentRecord()
    {
    }

    //Constructor with 4 arguments.
    public StudentRecord(String firstName, String lastName, double studentID, double gradeCourseOne)
    {
        this.firstName = firstName;
        this.lastName = lastName;
        this.studentID = studentID;
        this.gradeCourseOne = gradeCourseOne;
    }

    //Setting up the setters.
    public void setFirstName(String firstName)
    {
        this.firstName = firstName;
    }

    public void setLastName(String lastName)
    {
        this.lastName = lastName;
    }
}
```

## Part 2



```
StudentRecord - Week4
Class Edit Tools Options
StudentRecord X
Compile Undo Cut Copy Paste Find... Close Source Code

public void setLastName(String lastName)
{
    this.lastName = lastName;
}

public void setStudentID(double studentID)
{
    this.studentID = studentID;
}

public void setGradeCourseOne(double gradeCourseOne)
{
    this.gradeCourseOne = gradeCourseOne;
}

//Setting up the getters.
public String getFirstName()
{
    return firstName;
}

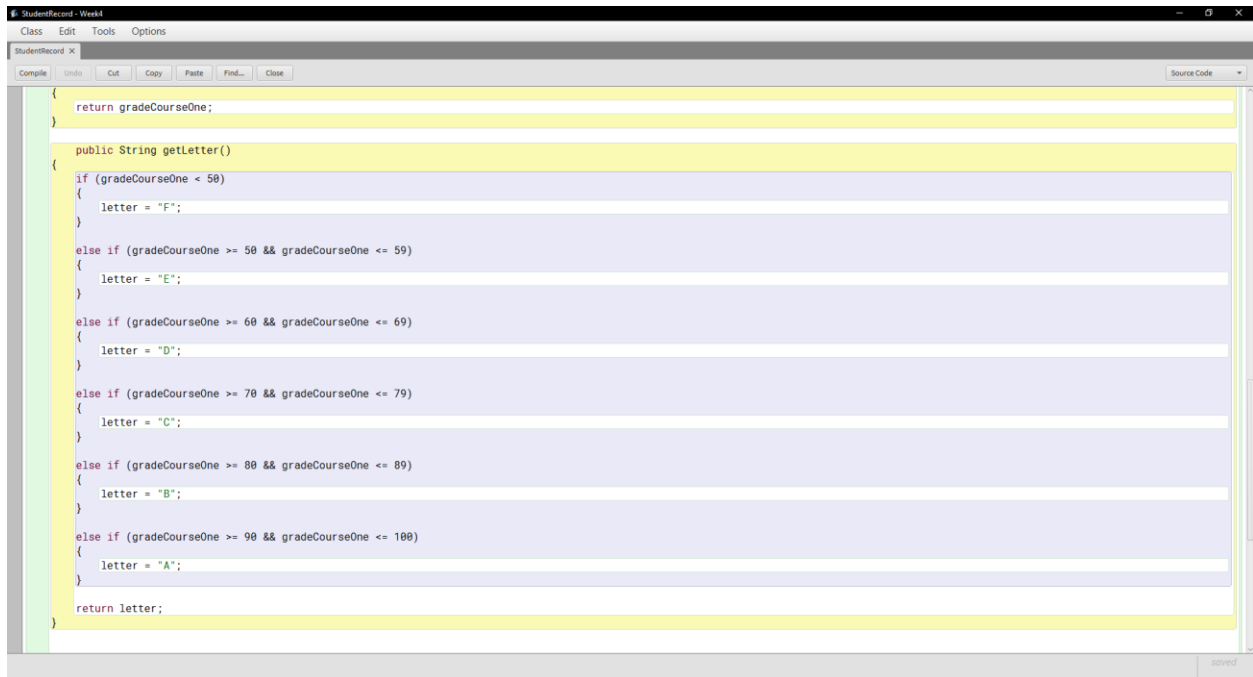
public String getLastName()
{
    return lastName;
}

//Method within a getter which returns a letter based on grade
public double getStudentID()
{
    return studentID;
}

public double getGradeCourseOne()
{
    return gradeCourseOne;
}

public String getLetter()
{
}
```

## Part 3



The screenshot shows a Java IDE window titled "StudentRecord - Week4". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". The code editor displays the following Java code:

```
{
    return gradeCourseOne;
}

public String getLetter()
{
    if (gradeCourseOne < 50)
    {
        letter = "F";
    }

    else if (gradeCourseOne >= 50 && gradeCourseOne <= 59)
    {
        letter = "E";
    }

    else if (gradeCourseOne >= 60 && gradeCourseOne <= 69)
    {
        letter = "D";
    }

    else if (gradeCourseOne >= 70 && gradeCourseOne <= 79)
    {
        letter = "C";
    }

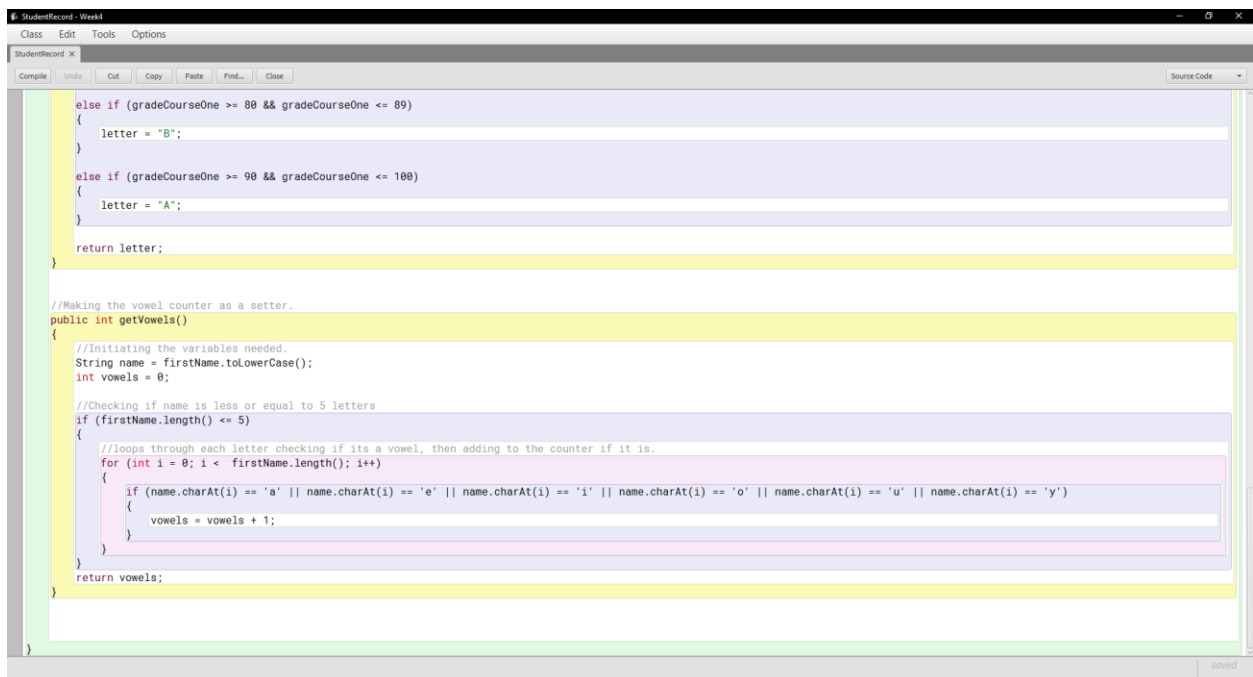
    else if (gradeCourseOne >= 80 && gradeCourseOne <= 89)
    {
        letter = "B";
    }

    else if (gradeCourseOne >= 90 && gradeCourseOne <= 100)
    {
        letter = "A";
    }

    return letter;
}
```

The code is color-coded: keywords are in blue, strings in red, and comments in green. The IDE status bar at the bottom right shows "saved".

## Part 4



The screenshot shows the same Java IDE window, now displaying the implementation of the `getVowels()` method. The code is as follows:

```
    else if (gradeCourseOne >= 80 && gradeCourseOne <= 89)
    {
        letter = "B";
    }

    else if (gradeCourseOne >= 90 && gradeCourseOne <= 100)
    {
        letter = "A";
    }

    return letter;
}

//Making the vowel counter as a setter.
public int getVowels()
{
    //Initiating the variables needed.
    String name = firstName.toLowerCase();
    int vowels = 0;

    //Checking if name is less or equal to 5 letters
    if (firstName.length() <= 5)
    {
        //loops through each letter checking if its a vowel, then adding to the counter if it is.
        for (int i = 0; i < firstName.length(); i++)
        {
            if (name.charAt(i) == 'a' || name.charAt(i) == 'e' || name.charAt(i) == 'i' || name.charAt(i) == 'o' || name.charAt(i) == 'u' || name.charAt(i) == 'y')
            {
                vowels = vowels + 1;
            }
        }
    }

    return vowels;
}
```

The code continues with the same color-coding as Part 3. The IDE status bar at the bottom right shows "saved".

# StudentRecordTestHarness.java

## Part 1

```
StudentRecordTestHarness - Week4
Class Edit Tools Options
StudentRecordTestHarness X
Compile Undo Cut Copy Paste Find... Close Source Code

/**
 * Author: Alex Dorodko
 * Date: 18/OCT/2020
 * Time: 5:52 PM
 * Application purpose: Testing out the student record class, and using input to fill a student record.
 */
import java.util.Scanner;

public class StudentRecordTestHarness
{
    public static void main(String[] args)
    {
        //Initiating the scanner to read user input
        Scanner sc = new Scanner(System.in);

        //Instigating the first record using random values for percentage and student number
        StudentRecord firstRecord = new StudentRecord("Johnny", "Guitar", Math.random() * (10000 - 0 + 1) + 0, Math.random() * (100 - 0 + 1) + 0);

        //Outputting the first student record
        System.out.println("\nStudent Record #1\n\nFirst Name: " + firstRecord.getFirstName() + "\nLast Name: " + firstRecord.getLastName() + "\nStudent Number: " + Math.round(firstRecord.getStudentID()));

        //Outputting the number of vowels, but it will only count them if the name is less than 5 characters long.
        System.out.println("The number of vowels in the first name is " + firstRecord.getVowels() + ".\n");

        //Creating the second student record with no arguments.
        StudentRecord secondRecord = new StudentRecord();

        //Asking the user for the first name for the second record and storing it
        System.out.println("Creating the second record:\n\nEnter the first name: ");
        secondRecord.setFirstName(sc.nextLine());

        //Asking the user for the last name for the second record and storing it
        System.out.println("Enter the last name: ");
        secondRecord.setLastName(sc.nextLine());

        //Asking the user for the student number for the second record and storing it
        System.out.println("Enter the student number: ");
    }
}
```

## Part 2

```
StudentRecordTestHarness - Week4
Class Edit Tools Options
StudentRecordTestHarness X
Compile Undo Cut Copy Paste Find... Close Source Code

//Asking the user for the student number for the second record and storing it
System.out.println("Enter the student number: ");
secondRecord.setStudentID(sc.nextDouble());
//Making sure the student number is between 1 and 10,000.
while (secondRecord.getStudentID() <= 0 || secondRecord.getStudentID() > 10000)
{
    System.out.println("Invalid student number. Value must be between 1 - 10,000. Enter the student number: ");
    secondRecord.setStudentID(sc.nextDouble());
}

//Asking the user for the percent grade which the student recieved for the second record and storing it
System.out.println("Enter the percent grade:");
secondRecord.setGradeCourseOne(sc.nextDouble());
//Making sure the input is between 0 and 100
while (secondRecord.getGradeCourseOne() < 0 || secondRecord.getGradeCourseOne() > 100)
{
    System.out.println("Invalid percent. Value must be between 0 and 100. Enter the percent grade:");
    secondRecord.setGradeCourseOne(sc.nextDouble());
}

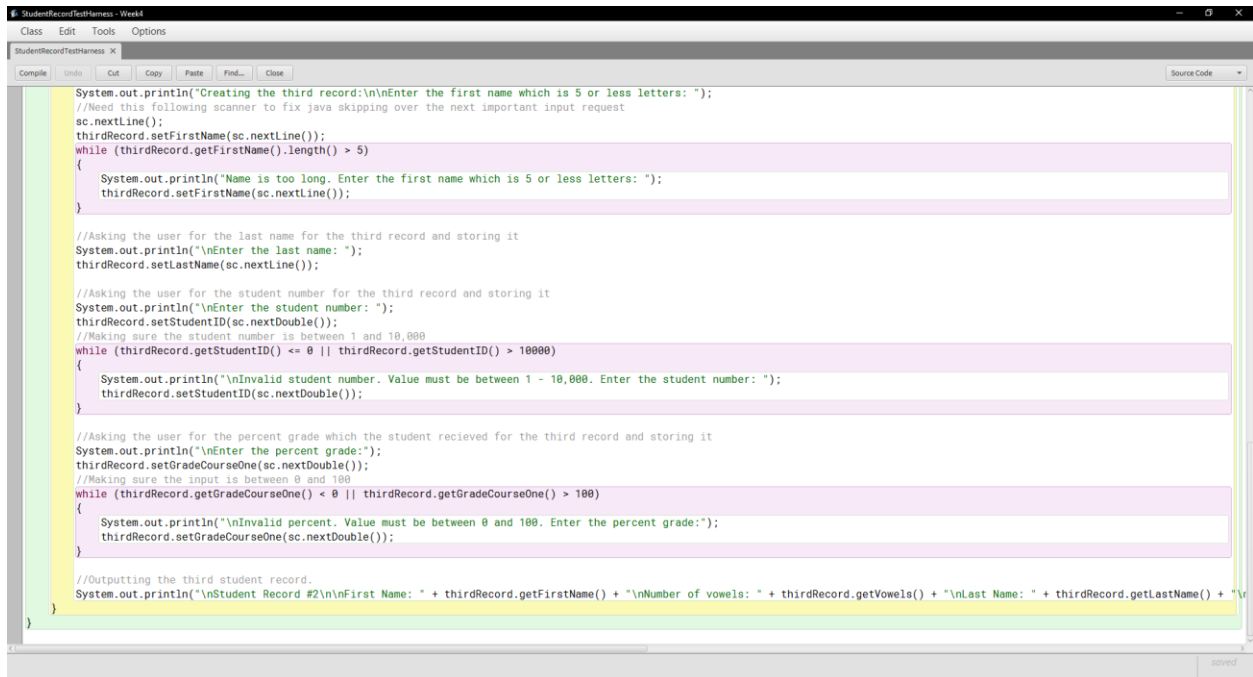
//Outputting the second student record.
System.out.println("\nStudent Record #2\n\nFirst Name: " + secondRecord.getFirstName() + "\nLast Name: " + secondRecord.getLastName() + "\nStudent Number: " + Math.round(secondRecord.getStudentID()));
System.out.println("The number of vowels in the first name is " + secondRecord.getVowels() + ".\n");

//3rd student record for the last challenge where we ask the user for 5 letter name, and print the number of vowels in the name along with other data

//Creating the third student record with no arguments.
StudentRecord thirdRecord = new StudentRecord();

//Asking the user for the first name for the third record and storing it
System.out.println("Creating the third record:\n\nEnter the first name which is 5 or less letters: ");
//Need this following scanner to fix java skipping over the next important input request
sc.nextLine();
thirdRecord.setFirstName(sc.nextLine());
while (thirdRecord.getFirstName().length() > 5)
{
}
```

## Part 3



```
System.out.println("Creating the third record:\nEnter the first name which is 5 or less letters: ");
//Need this following scanner to fix java skipping over the next important input request
sc.nextLine();
thirdRecord.setFirstName(sc.nextLine());
while (thirdRecord.getFirstName().length() > 5)
{
    System.out.println("Name is too long. Enter the first name which is 5 or less letters: ");
    thirdRecord.setFirstName(sc.nextLine());
}

//Asking the user for the last name for the third record and storing it
System.out.println("\nEnter the last name: ");
thirdRecord.setLastName(sc.nextLine());

//Asking the user for the student number for the third record and storing it
System.out.println("\nEnter the student number: ");
thirdRecord.setStudentID(sc.nextDouble());
//Making sure the student number is between 1 and 10,000
while (thirdRecord.getStudentID() <= 0 || thirdRecord.getStudentID() > 10000)
{
    System.out.println("\nInvalid student number. Value must be between 1 - 10,000. Enter the student number: ");
    thirdRecord.setStudentID(sc.nextDouble());
}

//Asking the user for the percent grade which the student recieved for the third record and storing it
System.out.println("\nEnter the percent grade:");
thirdRecord.setGradeCourseOne(sc.nextDouble());
//Making sure the input is between 0 and 100
while (thirdRecord.getGradeCourseOne() < 0 || thirdRecord.getGradeCourseOne() > 100)
{
    System.out.println("\nInvalid percent. Value must be between 0 and 100. Enter the percent grade:");
    thirdRecord.setGradeCourseOne(sc.nextDouble());
}

//Outputting the third student record.
System.out.println("\nStudent Record #2\nFirst Name: " + thirdRecord.getFirstName() + "\nNumber of vowels: " + thirdRecord.getVowels() + "\nLast Name: " + thirdRecord.getLastName() + "\n");
}
```

# Output

## Part 1

```
Blue: Terminal Window - Week4
Options

Student Record #1

First Name: Johnny
Last Name: Guitar
Student Number: 1543
Percent Grade: 62
Letter Grade: D

The number of vowels in the first name is 2.

Creating the second record:

Enter the first name:
Cole

Enter the last name:
Andison

Enter the student number:
9493

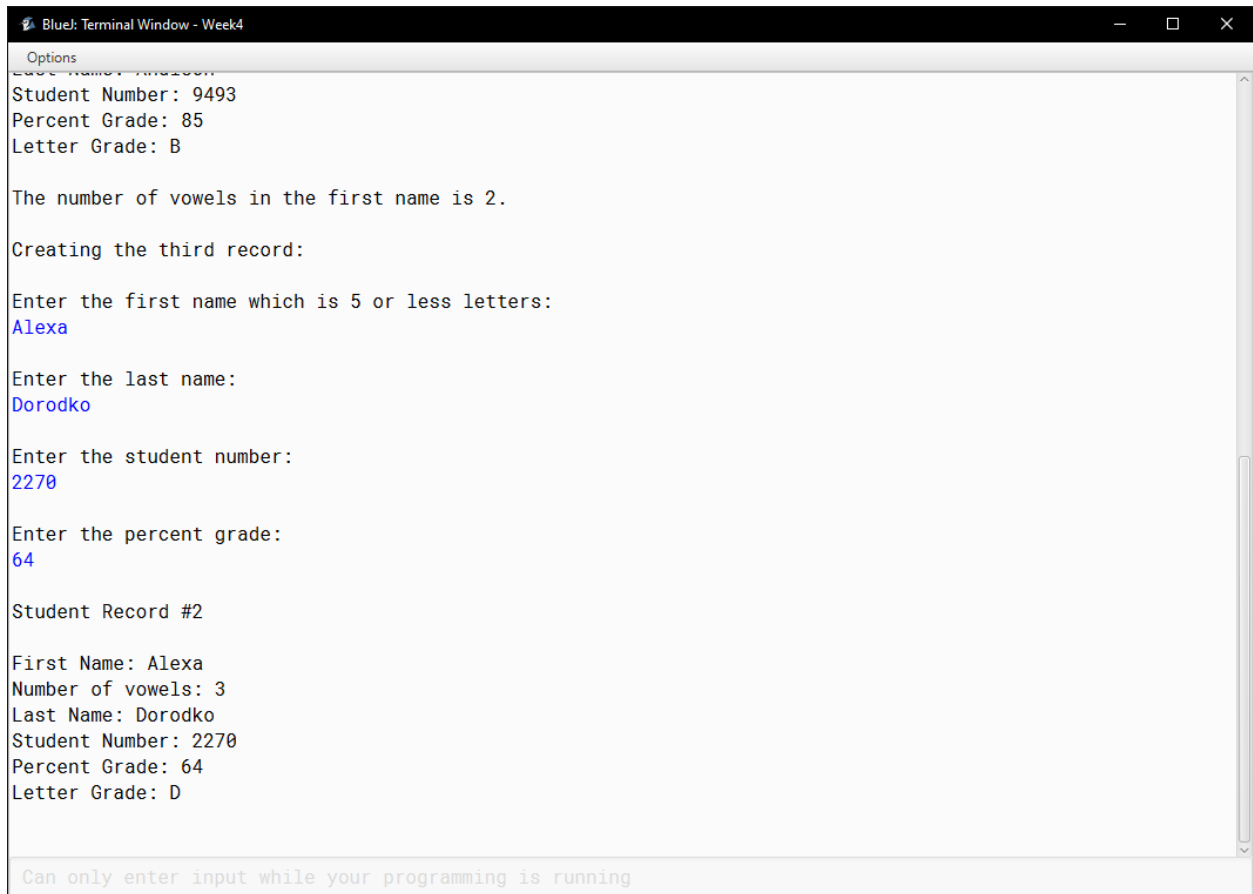
Enter the percent grade:
85

Student Record #2

First Name: Cole
Last Name: Andison
Student Number: 9493
Percent Grade: 85
Letter Grade: C

Can only enter input while your programming is running
```

## Part 2



The screenshot shows a BlueJ Terminal Window titled "BlueJ: Terminal Window - Week4". The window contains the following text:

```
Options
Enter Name: Anderson
Student Number: 9493
Percent Grade: 85
Letter Grade: B

The number of vowels in the first name is 2.

Creating the third record:

Enter the first name which is 5 or less letters:
Alexa

Enter the last name:
Dorodko

Enter the student number:
2270

Enter the percent grade:
64

Student Record #2

First Name: Alexa
Number of vowels: 3
Last Name: Dorodko
Student Number: 2270
Percent Grade: 64
Letter Grade: D

Can only enter input while your programming is running
```

The input for the first name, last name, student number, and percent grade is shown in blue text, indicating it was entered by the user. The window also features a standard window title bar with minimize, maximize, and close buttons, and a vertical scrollbar on the right side.