```
    if ((name.getFullName().equals(otherName.getLastName()))
        && (major.equals(((Student)object).getMajor()))) {
      return true;
    }
    return false;
  }

  /** Compare student's major, then last name */
  public int compareTo(Object object) {
  }

public class Person {
  // Variables
  private Name name;

  // Constructors
  public Person(Name name) {
    this.name = name;
  }

  // Default Constructor
  public Person() {
  }

  // Instance Methods
  public Name getName() {
    return name;
  }

  public void setName(Name name) {
    this.name = name;
  }

  public String toString() {
    return name.getFullName();
  }

  public boolean equals(Object object) {
    Name otherName = ((Person)object).getName();
    if (name.getFullName().equals(otherName.getFullName())) {
      return true;
    }
    return false;
  }
```

```
  /** Compare person's last name */
  public int compareTo(Object object) {

}

class Name {
  // Variables
  private String firstName;
  private char mi;
  private String lastName;

  // Constructors
  public Name(String firstName, char mi, String lastName) {
    // Set new variables to instance.
    this.firstName = firstName;
    this.mi = mi;
    this.lastName = lastName;
  }

  // Default Constructor
  public Name() {

  }

  // Instance Methods
  public String getFirstName() {
    return firstName;
  }

  public void setFirstName(String firstName) {
    this.firstName = firstName;
  }

  public char getMi() {
    return mi;
  }

  public void setMi(char mi) {
    this.mi = mi;
  }

  public String getLastName() {
    return lastName;
```

```
    }

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

    public String getFullName() {
      return firstName + " " + mi + " " + lastName;
    }
  }
}
```

Figure 8-2-1

4. Express the result to the console.
5. Check the Java file for errors and right-click the class in the project to display the context menu. Choose Run, Java Application in the context menu to run the class.
6. Save a screenshot of the output and submit it to your instructor.

Did it work?

Were you able to:

- Display an array of student names?
- Display an array of sorted student names?
- Display the max value of an array of student names?
- Display an array of strings?
- Display an array of sorted strings?
- Display the max value of an array of strings?
- Display an array of integers?
- Display an array of sorted integers?

Summary

This unit discussed how to define arrays and interfaces. The next unit will examine GUI programming in Java applications.

Reminders

- Students should review Chapter 14 of Introduction to Java Programming before starting Lab 8-1 and Lab 8-2.

(End of Unit 8)

Lab 9-1: Enabling GeometricObject comparable

Purpose

This lab will modify the GeometricObject class to implement the Comparable interface, and define a static max method in the GeometricObject class for finding the larger of two GeometricObject objects.

Finally, you will write a test program that uses the max method to find the larger of two circles and the larger of two rectangles.

How it Fits In

Students will continue working with the basic building blocks of Java programs. They will work with a Java statement that will deal with abstract classes and the creation of a GUI.

Objectives

- Create a Java program
- Create an abstract class
- Create a circle

Estimated Completion Time: 40 minutes

Task 1: Study Material

Read Chapter 15 from *Introduction to Java Programming*, Brief Version, (9th ed) by Liang, Y. D. (2013).

- Page references
  - Page 560, Section 15.2 (Abstract Classes)
  - Page 562, Section 15.2.1(Why Abstract Methods)
  - Page 562, Listing 15.4 (TestGeometricObject.java)

Eclipse and Java websites for additional reference

- http://www.cs.armstrong.edu/liang/intro9e/supplement.html
- http://help.eclipse.org/juno/index.jsp?topic=/org.eclipse.platform.doc.user/reference/ref-cheatsheets.htm

Keywords

- Abstract class

- GeometricObject
- Radius

Task 2: Create a Java Program to Implement the New GeometricObject Class

Procedure

1. Open the Eclipse software that is installed on the Lab computer.
2. Create a Java class named GeometricObject.
3. Use the sample code in Figure 9-1 as a starting point for this lab.

```
public class Test {
 // Main method
 public static void main(String[] args) {
  // Create two comparable circles
  Circle1 circle1 = new Circle1(5);
  Circle1 circle2 = new Circle1(4);

  // Display the max circle
  Circle1 circle = (Circle1)GeometricObject1.max(circle1, circle2);
  System.out.println("The max circle's radius is " +
   circle.getRadius());
  System.out.println(circle);
 }
}

abstract class GeometricObject1 implements Comparable {
 // Implement it
}

// Circle.java: The circle class that extends GeometricObject
class Circle1 extends GeometricObject1 {
 // Implement it
}
```

Figure 9-1

4. Express the result to the console.
5. Check the Java file for errors and right-click the class in the project to display the context menu. Choose Run, Java Application in the context menu to run the class.
6. Save a screenshot of the output and submit it to your instructor.

Did it work?

Were you able to:

- Display two circles?
- Create a max radius of circle 1?
- Create a max radius of circle 2?
- Display the compared results of the two circles?
- Display the area of both circles?
- Display the perimeter of both circles?

Unit 9 Lab 2: The Complex Class

Purpose

A complex number is a number of the form $a+bi$, where a and b are real numbers and i is $\sqrt{-1}$. The numbers a and b are known as the real part and imaginary part of the complex number, respectively. You can perform addition, subtraction, multiplication, and division for complex numbers using the following formula:

$$a+bi+c+di=(a+c)+(b+d)i$$
$$a+bi-(c+di)=(a-c)+(b-d)i$$
$$(a+bi)*(c+di)=(ac-bd)+(bc+ad)i$$
$$(a+bi)/(c+di)=(ac+bd)/(c^2+d^2)+(bc-ad)i/(c^2+d^2)$$

You can also obtain the absolute value for a complex number using the following formula:

$$|a+bi|=\sqrt{a^2+b^2}$$

(A complex number can be interpreted as a point on a plane by identifying the (a,b) values as the coordinates of the point. The absolute value of the complex number corresponds to the distance of the point to the origin, as shown in Figure 15.12b.)

Design a class named Complex for representing complex numbers and the methods add, subtract, multiply, divide, abs for performing complex-number operations, and override toString method for returning a string representation for a complex number. The toString method returns a + bi as a string. If b is 0, it simply returns a.

Provide three constructors Complex(a, b), Complex(a), and Complex(). Complex() creates a Complex object for number 0 and Complex(a) creates a Complex object with 0 for b. Also provide the getRealPart() and getImaginaryPart() methods for returning the real and imaginary part of the complex number, respectively.

How it Fits In

Students will continue working with the basic building blocks of Java programs. They will work with a Java statement that will deal with complex numerical numbers and the results of addition, subtraction, multiplication, and division.

Objectives

- Create a Java program
- Create a Complex class
- Create numerical problems

Estimated Completion Time: 55 minutes

Task 1: Study Material

Read Chapter 15 from *Introduction to Java Programming*, Brief Version, (9th ed) by Liang, Y. D. (2013).

- Page references
  - Page 565, Section 15.3 (Case Study: the Abstract Number Class)
  - Page 566, Listing 15.5(LargestNumbers.java)
  - Page 584, Section 15.9 (Case Study: The Rational Class)

Eclipse and Java websites for additional reference

- http://www.cs.armstrong.edu/liang/intro9e/supplement.html
- http://help.eclipse.org/juno/index.jsp?topic=/org.eclipse.platform.doc.user/reference/ref-cheatsheets.htm

Keywords

- Abstract method
- Math
- Rational class

Task 2: Create a Java Program to Display a Complex Numerical Problem

Procedure

1. Open the Eclipse software that is installed on the Lab computer.
2. Create a Java class named Complex.
3. Write a test program that prompts the user to enter two complex numbers.
4. Display the results of their addition, subtraction, multiplication, and division.
5. A sample of the results is listed below in Figure 9-2.

<Output>

Enter the first complex number: 3.5 5.5

```
Enter the second complex number: -3.5 1
(3.5 + 5.5i) + (-3.5 + 1.0i) = 0.0 + 6.5i
(3.5 + 5.5i) - (-3.5 + 1.0i) = 7.0 + 4.5i
(3.5 + 5.5i) * (-3.5 + 1.0i) = -17.75 + -15.75i
(3.5 + 5.5i) / (-3.5 + 1.0i) = -0.5094 + -1.7i
|3.5 + 5.5i| = 6.519202405202649
                                    <End Output>
```

Figure 9-2

6. Express the result to the console.
7. Check the Java file for errors and right-click the class in the project to display the context menu. Choose Run, Java Application in the context menu to run the class.
8. Save a screenshot of the output and submit it to your instructor.

Summary

This unit examined how to get started with abstract classes and the creation of in depth numerical problems. The next unit will look at event-driven programming and more GUI programming in Java applications.

Reminders

- Students should review Chapter 15 of Introduction to Java Programming before starting Lab 9-1 and 9-2.

(End of Unit 9)

Lab 10-1: Alternating Two Messages

Purpose

Write a program to rotate two messages –"Java is fun" and "Java is powerful"—displayed on a panel with a mouse click.

How it Fits In

Students will continue working with the basic building blocks of Java programs. They will work with a Java statement that will deal with dialog box and the different methods used with the MouseEvent command.

Objectives

- Create a Java program
- Create a MouseEvent statement
- Create a GUI

Estimated Completion Time: 50 minutes

Task 1: Study Material

Read Chapter 16 and 17 from *Introduction to Java Programming*, Brief Version, (9th ed) by Liang, Y. D. (2013).

- Page references
  - Page 617, Section 16.8 (Mouse Events)
  - Page 620, Section 16.9 (Listener Interface Adapters)
  - Page 621, Section 16.10 (Key Events)

Eclipse and Java websites for additional reference

- http://www.cs.armstrong.edu/liang/intro9e/supplement.html
- http://help.eclipse.org/juno/index.jsp?topic=/org.eclipse.platform.doc.user/reference/ref-cheatsheets.htm

Keywords

- Event-driven programming
- Event object