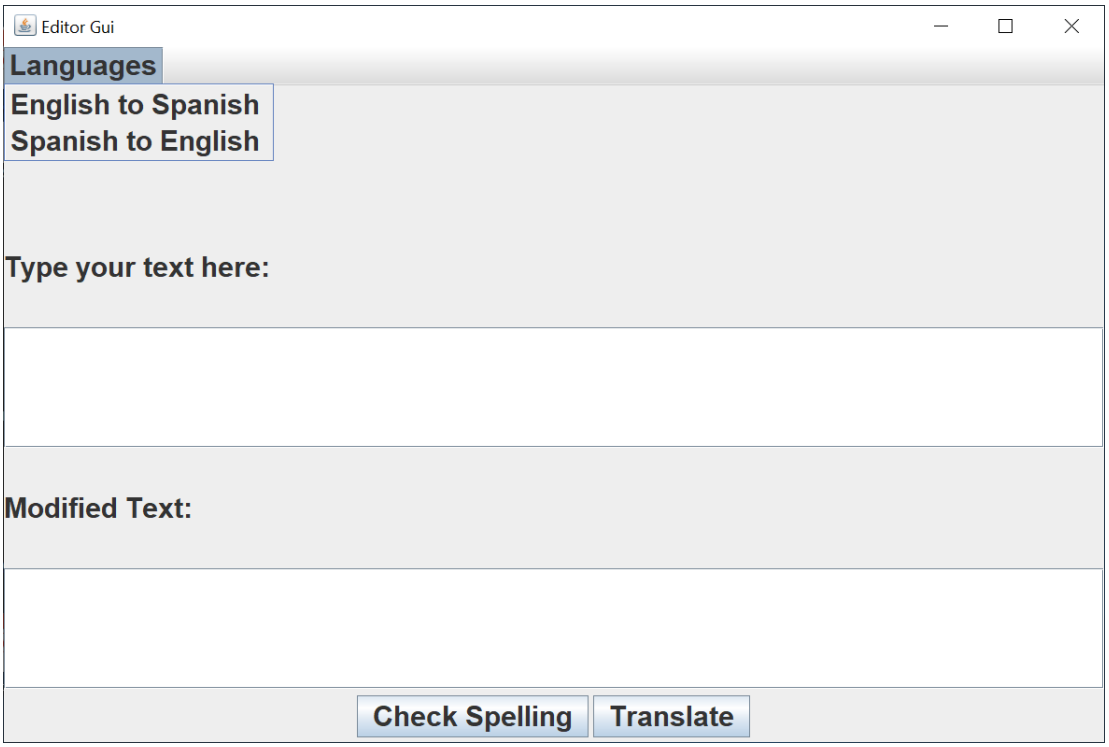
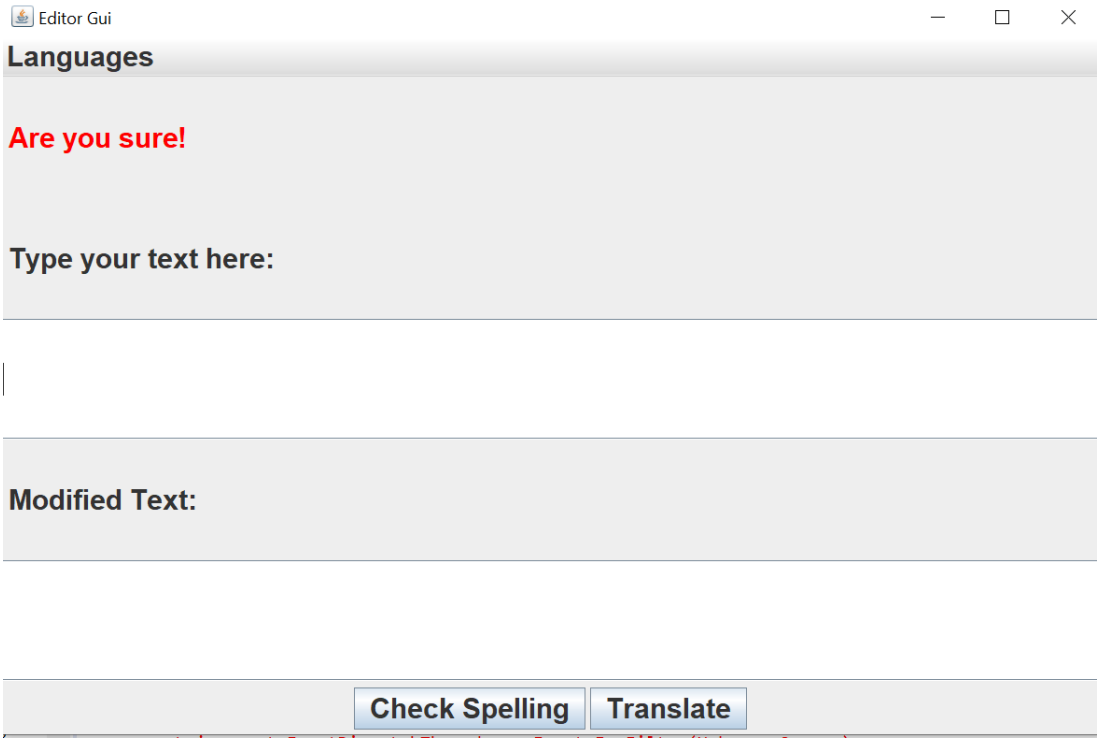
1. Write a Java program that creates and displays the following gui. (In this problem there is no need to program any listener response to user actions.)



2. Modify the program that you wrote for the last gui by adding a windowListener to modify the behavior of the close window button. The first time it is pressed the Gui should display the message *Are you sure!* in red. The second time it is pressed, the Gui should display *Are you really sure!* and the third time it is pressed the program should exit. (One tricky issue to solve is how to prevent the gui window from disappearing when the close window button is clicked the first two times.)



3. What will be printed by the following program?

```
public class Question2 {
   public static void main (String[] args) {

      B b = new B();
      A a = b;

      System.out.println(a);          // line A
      b.increment();
      System.out.println(a);          // line B
      a = new A();
      System.out.println(a.getB());   // line C
      System.out.println(b.getA());   // line D
   }
}
```

The two classes are implemented as follows.

```
class A {
   int x = 0;

   public void increment() {
      x++;
   }
   public String toString() {
      return "class A: " + x;
   }
   public B getB() {
      return new B( );
   }
}
```

```
class B extends A {
   public String toString( ) {
      return "Class B: " + x;
   }

   public A  getA( ) {
      return getB( );
   }
}
```

(a) The output generated at line A is:
Answer:

(b) The output generated at line B is:
Answer:

(c) The output generated at line C is:
Answer:

(d) The output generated at line D is:
Answer:

4. For this question, write code fragments or methods as directed.

   (a) Create the Point class that depends on a generic data type parameter T. It has two instance variables called xCoordinate and yCoordinate that both have type T. Write a two parameter constructor to initialize the instance variables.
   Answer:

   (b) Write a main method that has instructions to perform the following tasks (in order): Declare an ArrayList of Strings called carModels. Add three models. Print the size of the list. Remove the model from index 1. Insert a model at index 0. Replace the model at index 2.
   Answer:

   (c) Write a main method with a try-catch block. Inside the block it creates an array of Strings — the array is called *list* and has size 10. Print the value stored in list[10]. The catch block should catch ArithmeticException and RuntimeException. Your code shouldn't be longer than 9 lines excluding curly braces.
   Answer:

   (d) Write the private method *inputCircles* that is called by the following main method. Assume that *class Circle* exists and has constructors and a toString method. Your method should open a Scanner and obtain a radius for each of the 20 circles that you make from the user. (You should assume that the class Scanner has already been imported.)

```
public static void main(String args[]) {
   Circle[] data = new Circle[20];
   inputCircles(data);
   for (Circle c:data) System.out.println(c);
}
```

   Answer:

5. Create a class called **RoomCounter** which counts the number of people in a room. We know that the number of people in a room can never be negative. The class should have the following methods:

   - addPerson - adds one person to the room
   - removePerson - removes one person from the room
   - getCount - returns the number of people in the room

   If removePerson() would make the number of people less than zero, throw a NegativeCounterException.

   (a) Create the class **RoomCounter** with a default constructor and one instance variable **count** of type int. Write the three methods and throw an Exception where appropriate. This class should be short and to the point. The only getter method is getCount().
   Answer:

   (b) Define the class **NegativeCounterException**.
   Answer:

   (c) In the class RoomCounter, write a static main() method. Create a RoomCounter object, and add three people to a room. Then, in a try-catch block remove one person at a time and catch any exception. After each removal of a person, print the count.
   Answer:

6. On the next page you are given two Java files: Shape.java and Rectangle.java.

   (a) Create a class called **Square** that is a subclass of Rectangle and implements the interface Shape, in the package squares. Add a field called **count** to this class to count the number of squares created in the program.

   Answer:

   (b) Implement the constructor of the Square class. Remember to update the variable count. Write only one method called toString to return the String "square".
   Answer:

   (c) On the next page, take a look at the main() function. Write the output from each of the six println statements below.

   1. Output from (a):
   2. Output from (b):
   3. Output from (c):
   4. Output from (d):
   5. Output from (e);
   6. Output from (f):

The file Shape.java contains:

```
package shapes;

public interface Shape {
   public int area( );
   public int perimeter( );
}
```

The file Rectangle.java contains:

```
package rectangles;


public class Rectangle implements Shape {
   private int width;
   private int length;

   public Rectangle(int length, int width) {
      this.length = length;
      this.width = width; }

   public int area() { return length * width; }

   public int perimeter() { return 2 * ( length + width ); }
     // setter and getter methods are omitted.

   public String toString() { return "rectangle"; } }
```

The main() function in the class Square.java.

```
public static void main(String[] args) {
  Square s = new Square(3);
  System.out.println(s.perimeter()); //---------(a)
  Shape p = new Square(5);
  System.out.println(p.toString()); //----------(b)
  Rectangle r = new Square(6);
  Syste.out.println(r); //----------------------(c)
  Square s = new Square(3);
  System.out.println(s.area()); //--------------(d)
  Shape p = new Square(5);
  System.out.println(p.perimeter()); //---------(e)
  Rectangle r = new Square(6);
```

```
  System.out.println(r.area()); //---------------(f)
}
```

---

7. In the following code the marked lines contain errors. For each part, answer the question about the error.

```
Part 1: What is the error at line(a)?

interface A {
  void m1();
}

class B implements A {
  void m1() { //---------------------------(a)
    System.out.println("m1");
  }
}

Part 2: What is wrong with lines (b) and (c)?

class MyGen<T, V> {
  T t;
  V v;
  void set(T o) { t = o; } //-------------(b)
  void set(V o) { v = o; } //-------------(c)
}

Part 3: What is wrong with line d?

void methodX() {
  throw new ClassNotFoundException(); //-----(d)
}

Part 4: What is the error in line e?

class Y {
  int x = 8;
  static void changeX() {
    x = x/2;   //----------------------------(e)
  }
}

Part 5: What is the error in line f?
```

```
class P {
  final void meth() {
    System.out.println(("In P's meth()");
  }
}
class Q extends P {
  void  meth() { //-------------------------(f)
    System.out.println("IN Q's meth()");
  }
}
```

Part 6: For this part write the corrected lines below in the answer section.

```
class MyCla$$ {
  integer x = 3.0; //-------------------------(1)
  boolean b = = false; //---------------------(2)
  MyClass(boolean b) { b = b; } //-----------(3)
  int doIt() { } //---------------------------(4)
  int don'tDoIt() { return this; }  //-------(5)
}
```

Identify the error in each of the labeled statement above.

(a) Describe the errors in lines (a) - (f) below:

1. **Answer to (a):**
2. **Answer to (b), (c):**
3. **Answer to (d):**
4. **Answer to (e):**
5. **Answer to (f):**

(b) Write corrected answers from Part 6 below.

1. **Answer to (1):**
2. **Answer to (2):**
3. **Answer to (3):**
4. **Answer to (4):**
5. **Answer to (5):**

8. Read the following code.

```
class A {
   void method(char ch) {  System.out.println("A.method() " + ch); }
   String what() { return "Returning A";}
   void adjust(){ System.out.println("Adjusting A");}
}

class B extends A {
   void method(char ch) { System.out.println("B.method() " + ch);}
```

```
   String what() {return "Returning B";}
   void adjust() {System.out.println("Adjusting B");}
}

class C extends A {
   void method(char ch) {System.out.println("C.method() " + ch);}
   String what() {return "Returning C";}
   void adjust() {System.out.println("Adjusting C");}
}

class D extends A {
   void method(char ch) { System.out.println("D.method() " + ch);}
   String what() {return "returning D";}
   void change() {System.out.println("Changed String"); }
   void adjust() {System.out.println("Adjusting D"); }
}

class E extends B {
   void method(char ch) { System.out.println("E.method() " + ch); }
   String what() { return "Returning F"; }
}

class F extends B {
   void method(char ch) { System.out.println("F.method() " + ch); }
   void adjust() { System.out.println("Adjusting F"); }
}

public class Q1 {
    public static void main(String[] args) {
       A a = new C();
       System.out.println(a.what()); //-----(a)
       A a1 = new E();
       a1.method('X');//-----(b)
       B b = new B();
       b.adjust(); //--------(c)
       B b1 = new F();
       System.out.println(b1.what()); //----(d)
       A a2 = new D();
       ((D)a2).change(); //-------(e)
   }
}
```

(a) What is printed on line (a)?

Answer:

(b) What is printed at line (b)?

Answer:

(c) What is printed at line (c)?
Answer:

(d) What is printed at line (d)?
Answer:

(e) What is printed at line (e)?
Answer:

9. For this question you need to write some methods and class headers.

   (a) Assume that you have written a Rectangle class with instance variables length and width. You have already written all set and get methods and area and perimeter methods. **Write an equals() method** that takes Object o as a parameter. The method should return true when the Object o is a rectangle with the same length and width.
   Answer:

   (b) A class named Fruit implements an interface called Edible. The interface has a single method called howToEat(). A class called Orange extends Fruit and implements Edible. **Write the class header for the Orange class and override the howToEat() method** of the Fruit class. The method should print a brief message to the screen about how to eat an orange. Do not write any other methods or constructors.
   Answer:

10. Predict the output generated at the marked *println* lines in the following program. The program makes use of the *class Prob1*. Please enter each answer in the space provided.

```
import java.util.Stack;

public class Prob1 {

   public static int mystery(int m, int n) {
      if(n == 0) return 0;
      if(n % 2 == 0)
         return mystery(m+m, n/2);
      return mystery(m+m, n/2) + m;
   }


   public static void main(String[] args) {

      System.out.println(mystery(2, 29)); //------------------(a)

      String str = "Harry Potter";
      str.toLowerCase();
      str.substring(0, 7);
      System.out.println(str); //-----------------------------(b)
```

```
      Stack<Integer> stack = new Stack<>();
      int n = 50;
      while (n > 0) {
         stack.push(n % 2);
         n = n/2;
      }
      while (!stack.isEmpty())
         System.out.print(stack.pop()); //--------------------(c)
      System.out.println();

   }
}
```

(a) What is printed at line (a)?
Answer:

(b) What is the output at (b)?
Answer:

(c) What is printed at line (c)?
Answer: