# Problem 1 – Exception Handling

Write a program that asks the user to enter two numbers and divides the first number by the second. Use try/catch to handle DivideByZeroException and FormatException. Show a friendly message in each case and always print a message in finally that says 'Program finished.'

# Problem 2 – IComparable

Create a class Employee with the following properties: Name (string), Salary (double), and HiringDate (DateTime). Implement IComparable<Employee> to compare employees by HiringDate (earlier dates come first). Create a list of employees, sort them, and print the result.

# Problem 3 – IDisposable

Create a class FileLogger that implements IDisposable. In the constructor, open a text file for writing logs. In the Dispose method, close the file. Write a method LogMessage(string message) that writes a message to the file. Use the class with a using block to write 3 messages to the file.

# Problem 4 – SOLID Principles

You are given the following code that violates all SOLID principles:  
  
public class Employee {  
 public int Id { get; set; }  
 public string Name { get; set; }  
 public string Role { get; set; }  
 public double Salary { get; set; }  
  
 public void SaveEmployee(List<Employee> employees) {  
 employees.Add(this);  
 Console.WriteLine("Employee saved.");  
 }  
 public void PrintPayslip() {  
 Console.WriteLine($"Employee: {Name}, Role: {Role}, Salary: {Salary}");  
 }  
 public void CalculateBonus() {  
 if (Role == "Manager") Console.WriteLine($"Bonus: {Salary \* 0.2}");  
 else if (Role == "Developer") Console.WriteLine($"Bonus: {Salary \* 0.1}");  
 else Console.WriteLine("No Bonus");  
 }  
 public void SendEmail() {  
 Console.WriteLine($"Sending email to {Name}");  
 }  
}  
  
Explain what SOLID principles are violated in this code and rewrite it following the SOLID principles (use SRP for separate responsibilities, OCP for bonus calculation, DIP for repository, and ISP for notifications).