Name:_____

Exam #1 [Practice]
15-111/Kesden/Fall 2003

1. Please consider the following `Person` class. Notice the use of the Java keyword "this". What is "this"? And why is it used in the constructor below?

```
class Person {
  private String fullName;
  private String fullAddress;

  public Person (String fullName, fullAddress) {
    this.fullName = fullName;
    this.fullAddress = fullAddress;
  }

  public String getFullname() {
    return fullName;
  }

  public String getFullAddress() {
    return fullAddress;
  }
}
```

2. Below is the skeleton of a `PersonFinder` class based on the `Person` class defined above. Please complete this skeleton using a `Vector` as the primary data structure.

```
class PersonFinder {

  // Your code here

  // Constructor
  public PersonFinder () {
    // Your code here
  }

  // Return a Person's fullAddress
  public String findAddress(String fullName) {
    // Your code here
  }


  // Add a Person to this PersonFinder
  public void addPerson (Person newEntry)
  {
    // Your code here
  }
}
```

3. Below is a skeleton for a `miniVector` class, please complete it. This class should use an array as its primary data structure. Consistent with the semantics of Java's `Vector,` it should grow as needed.

```
public class miniVector {

  // Your code here.

  public void setElementAt (Object o, int index)  throws ArrayIndexOutOfBoundsException
  {
    // Your code here
  }

  public Object elementAt (int index)  throws ArrayIndexOutOfBoundsException
  {
    // Your code here
  }

  public void addElement (Object o)
  {
    // Your code here
  }

  public void insertElementAt (Object o, int index) throws ArrayIndexOutOfBoundsException
  {
    // Your code here
  }

  public void removeElementAt (int index) throws ArrayIndexOutOfBoundsException
  {
    // Your code here
  }
}
```

4. Repeat the exercise described for question #3, but this time, please use Java's `LinkedList` as the underlying data structure. Additionally, please create your own `Exception`, `MiniVectorBoundsException` and throw it in the event of an underflow or an overflow (negative index, or index greater than actually exists).

```
public class miniVector {

 // Your code here
 // Don't forget to define the MiniVectorBoundsException class


 public void setElementAt (Object o, int index)  throws MiniVectorBoundsException
 {
  // Your code here
 }

 public Object elementAt (int index)  throws MiniVectorBoundsException
 {
  // Your code here
 }

 public void addElement (Object o)
 {
  // Your code here
 }

 public void insertElementAt (Object o, int index) throws MiniVectorBoundsException
 {
  // Your code here
 }

 public void removeElementAt (int index) throws MiniVectorBoundsException
 {
  // Your code here
 }
}
```

5. Inheritance is a mechanism that is used to define classes with a certain relationship to each other. Please characterize this relationship. If it is easier for you, please answer the question, "When is it appropriate to define a new class using inheritance?"

6. What is the difference between an object and a class?

7. Consider the following definition, what does "p" represent?

```
Person p;
```

LinkedList reference

(Abbreviated Method Summary from Sun's API Documention):

