5 Further Java (ARB)

A social network can be represented as an undirected graph where vertexes represent people and edges represent bidirectional friendship relationships between two people. A developer constructs a simple representation of a social network using instances of the following Java class:

```
public class Person implements Serializable {
  private final long accountId;
  private String fullname;
  private Set<Person> friends;
  public Person(String fullname) { ... }
  public void addFriends(Set<Person> friends) { ... }
}
```

(*a*) The developer wishes to ensure that all public methods for `Person` are thread-safe. Describe what thread-safety means in this context. [1 mark]

(*b*) Write a thread-safe implementation of the constructor for `Person`. You must ensure no two instances of `Person` share the same `accountId`. You may create additional private fields, methods or inner classes in `Person`. [4 marks]

(*c*) Write a thread-safe implementation of `addFriends` which uses fine-grained locking to atomically establish new bidirectional friendship links between people in the social network. [5 marks]

(*d*) The developer wishes to serialize a copy of the graph of `Person` objects to disk. Without modifying the definition of `Person`, write an implementation of a non-thread-safe static method `void SocialNet.save(Set<Person> everyone, ObjectOutputStream oos)` which will write a *single* copy of all people in `everyone` to `oos`. Hint: `oos.writeObject(obj)` saves a copy of all objects reachable via references from `obj` and safely handles any cycles of references. [6 marks]

(*e*) Give two reasons why it is useful to store a single copy of each `Person` object in Part (*d*). [2 marks]

(*f*) Describe in words the modifications you would need to make if there are multiple threads attempting to add friendship links to the social network concurrently with the execution of `SocialNet.save`. [2 marks]