1) (15%) Create a UML class diagram for the following: A visitor class hierarchy consisting of abstract base class V plus subclasses V1 and V2 which visit subclasses C1 and C2 of base class C. Diagram V, V1, V2, C, C1, and C2, indicating which the methods visit and accept in the appropriate classes, using <abstract> to indicate which classes and methods are abstract.

2) (20%) Visitor subclass V1 performs the following simple task: if a V1 instance visits a C1, it prints "I am visiting a C1" else if it visits a C2 instance it prints "I am visiting a C2". Write the entire base class V and the subclass V1 so that it visits C1 and C2 accordingly. Do not use any "instanceof" or "if" statements. Use the space below or on the back of this sheet.
3) (10%) Consider:

```java
c C c = new C1();
V v = new V1();
```

Since those instances of C1 and V1 were both cast to their respective superclasses as variables c and v, is it still possible to write a line of code that creates a visitation between c and v, without explicit casting or use of instanceof? If yes, write it and explain why it works, and if no, explain why not.

4) (15%) Given

```java
C1 c1 = new C1();
V1 v1 = new V1();
```

Use a UML sequence diagram to detail a visitation between c1 and v1. If you are not fully happy with your diagram, add English as necessary.
5a) (5%) Could both V and C be interfaces? Explain.

5b) (5%) Provide a design reason why it might be practical necessity for V or C to be an interface.

6a) (5%) Describe the rationale (idea behind) the Visitor Pattern, i.e., its purpose, and why it is useful.
6b) (10%) Referring to the above hierarchy, what would be two representative uses for visitors (such as V1 or V2)?

6c) (5%) Outline the steps required to incorporate a new “visitee” subclass C3 in the C hierarchy, so that it follows this pattern.

6d) (5%) Outline the steps required to incorporate a new visitor subclass V3 into this pattern.

7) (5%) What are two reasons for making a class abstract?