Question 1. [Total Marks 5]

eXtreme Programming (XP) is an agile development method invented by Kent Beck.

Beck argues that while good design is important, *documenting* your design, or modelling your design, is not important. He argues that you should focus on "designing always" while coding, rather than producing any design artefacts before implementation.

In about half a page and in your own words, describe some key advantages and disadvantages to this approach, compared to explicitly documenting your design using UML notations such as class and sequence diagrams.

Question 2. [Total Marks 5]

The Java language uses **static typing**. That is, the type of every variable and method parameter, and the return types of methods, must be explicitly specified by the programmer.

Other languages, such as Python and JavaScript, use **dynamic typing**. Variables can hold a value of any type, and methods can return values of any type. At runtime, just before an operation such as an attribute access is performed, the interpreter for the language checks that this will be possible.

Explain in your own words what you think the advantages and disadvantages of static typing are compared to dynamic typing, in the context of developing reliable, maintainable systems. Write about half a page.

Question 3. [Total Marks 10]

In Java, **polymorphism** describes the process whereby a subclass overrides a method in a base class (or implements a method defined in an interface). The method code invoked depends on the class of the actual object, not the type specified by the reference.

In the context of the Harry Potter game, explain how the `act()` method in `HPActor` and its subclasses uses polymorphism. You may use pseudocode or snippets of Java code, or any kind of diagram if you wish to assist in your explanation. You must clearly explain how polymorphism works in this example.

If you can't remember what the `act()` method did, instead invent your own example where you would use polymorphism.

Explain why the use of polymorphism in the example is advantageous for writing understandable, maintainable and extensible code.

Take about a page for your answer.

Question 4. **[Total Marks 1 + 2 + 2 + 1 + 2 + 2 = 10]**

Consider the following code for Java classes `Watch`, `Counter`, and `Driver`, similar to those that we saw in lectures at the start of the semester (continued on next page):

```
public class Watch {

   private Counter milliseconds = new Counter();
   private Counter seconds = new Counter();
   private Counter minutes = new Counter();
   private Counter hours = new Counter();

   public void tick() {
      milliseconds.increment();
      if (milliseconds.getValue() == 1000) {
         milliseconds.reset();
         seconds.increment();
         if (seconds.getValue() == 60) {
            seconds.reset();
            minutes.increment();
            if (minutes.getValue() == 60) {
               minutes.reset();
               hours.increment();
               if (hours.getValue() == 24) {
                  hours.reset();
               }
            }
         }
      }
   }

   public void testWatch(int numTicks) {
      for (int i = 0; i < numTicks; i++) {
         System.out.println(
               String.format("%02d", hours.getValue())
               + ":"
               + String.format("%02d", minutes.getValue())
               + ":"
               + String.format("%02d", seconds.getValue())
               + ":"
               + String.format("%03d", milliseconds.getValue())
         );
         tick();
      }
   }

}
```

```
public class Counter {

   private int value = 0;

   public void reset() {
      value = 0;
   }

   public void decrement() {
      value--;
   }

   public void increment() {
      value++;
   }

   public int getValue() {
      return value;
   }
}

public class Driver {

   public static void main(String[] args) {

      Watch myWatch = new Watch();
      myWatch.testWatch(20000000);
   }

}
```

(a) Give an example of a place in class `Watch` where there is duplicate code. **(1 mark)**

(b) Explain how duplicate code can cause problems during software development and maintenance. Give an example of a problem duplicate code could cause, using the code example above. **(2 marks)**

(c) Suggest a way that the system above could be redesigned so that the duplicated code mentioned in your answer to part (a) would be removed. Explain how this redesign would solve the problem(s) you identified in part (b). **(2 marks)**

(d) There is a dependency between methods `tick()` and `testWatch(…)` in class `Watch` due to hard-coded constants. Explain what this dependency is. **(1 mark)**

(e) Explain why excessive dependencies between modules can cause problems during software development and maintenance. Give an example of a problem that such dependencies could cause, using the code example above. **(2 marks)**

(f) Suggest a way that the system could be redesigned so that the dependency described in your answer to part (d) would be removed. Explain how this redesign would solve the problem(s) you identified in part (e). **(2 marks)**

Question 5. [Total Marks 5]

It important to manage dependencies in software design. Encapsulation can be used to help limit dependencies.

(a) Explain what is meant by encapsulation in object-oriented design and how it helps to limit dependencies. **(2 marks)**

Elements of code can be encapsulated at various levels. One example is encapsulation within a method. Variables declared inside a method are local to that method – they cannot be accessed from outside the method.

(b) Give examples of two other levels of encapsulation that are possible in Java. Explain the features of the Java language that allow these levels of encapsulation to be enforced. **(3 marks)**