2000

plq 10
ACN

```
int n, p;
ModInt(int n, int p)
{   this.n = n; this.p = p;
}

ModInt add(ModInt b)
{
    if (p != b.p) <moan in some way>
    return new ModInt((a+b)%p, p);
}
// etc for subtract, multiply

Division code must implement extended Euclidean GCD as per
discrete maths notes!
}
```

3 Java full

Explain each of the following: in relevant cases include up to four lines of Java code (but not more) to illustrate your point:

1. How to print a table of integers in neat columns, eg

   ```
   Heading1  Heading2
   1234      989898976
   76        -735743
   8823674   66300
   ```

2. How to test if a floating point number is a NaN, an infinity or any other special cases that come to mind;

3. How to arrange that, in an applet, a screen image is refreshed properly whenever the user moves, obscures or uncovers windows;

4. Circumstances in which you might choose to declare on of your classes `final`, and cases when you might declare a method in a class `final` or `protected`.

5. The keyword `finally` and its use.

(4 marks per section)

3.1 Marking Notes

1. How to print a table of integers in neat columns, eg

   ```
   Heading1  Heading2
   1234      989898976
   76        -735743
   8823674   66300
   ```

Basically you are liable to do a lot by steam. I would set up a StringBuffer and add in first integer into it, then I can measure the length of the string buffer and deduce how much padding is needed, so I then put that in by adding in blanks. At the end I turn the string buffer into a string and print it. Yuk.

2. How to test if a floating point number is a NaN, an infinity or any other special cases that come to mind;

   There may be library routines to test. However also: The test (f == f) can yield false only if f is a NaN. Infinity is not a NaN (and it is not zero) but Infinity-Infinity is.

   To test for "-0.0" I might consider taking its reciprocal and seeing if that (an infinity) was negative.

3. How to arrange that, in an applet, a screen image is refreshed properly whenever the user moves, obscures or uncovers windows;

   Define a public void method called `paint` in your applet. It gets a `Graphics` as its arg and should be called for you whenever re-painting is needed.

4. Circumstances in which you might choose to declare on of your classes `final`, and cases when you might declare a method in a class `final` or `protected`.

   If a class is final you are not permitted to sub-class it. Eg String is such.

   protected methods go in a library in some package other than the one that the user writes code in, but the user is permitted to derive a new classe and override them. default package access would not let the user access them at all if they are in a different package.

   Making a method final prevents you adjusting that in any derived class.

   So these are ways to ensure that people can sub-class in controlled manners which helps the code that uses instances of the top level class know what they can be certain of wrt behaviour.

5. The keyword `finally` and its use.

   This is as in `try {} finally {}` and the finally clause will get obeyed on ANY exit from the protected block. used for clean-up, eg closing files etc.