Question 1 Consider the following abstract class which models lists of integers (the call \texttt{getElem(i)} returns the i-th element, if any, otherwise throws the exception \texttt{Undefined}): 

class Undefined extends Exception {}  
abstract class List {  
    abstract int getElem(int i) throws Undefined;
}

1. For each of the following declarations of heir classes of \texttt{List} tell whether it is well-typed and, if not, which is the error.

(a) class MyList1 extends List {  
    bool getElem(int i) { return true;}
}

(b) class MyList2 extends List {  
    int getElem(bool i) throws Undefined { return getElem(1); }
}

(c) class MyList3 extends List {  
    int getElem(bool i) throws Undefined { return 1; }
    int getElem(int i) { return (getElem(true)); }
}

2. Define the following heir classes of \texttt{List}

(a) \texttt{Empty} (empty lists),

(b) \texttt{Cons} (lists obtained adding an element on top of another list),

(c) \texttt{Interval} (lists consisting of the integers from \texttt{a} to \texttt{b}, for some \texttt{a}, \texttt{b} integers, empty if \texttt{a} > \texttt{b}).

3. Add in \texttt{List} a (non abstract) method which returns the length of a list.

Question 2 Let \( P \) be the following MINIJAVA (with inheritance) program 

class A {  
    A m (A a) {a}
}  

class B extends A {  
    A f;
}

1. Show the class type environment \( \Gamma_P \) which can be extracted from \( P \).

2. Prove, by applying the typing rules, that the following expression \( e \)

\[
B \ b = \text{new} \  B ; \ A \ a = \ b ; \ b . m ( b . f = b )
\]

is well-formed in the class type environment \( \Gamma_P \) and the empty local environment.
**Question 3** Consider the following class declarations in *SOOL* with `MyType`:

```plaintext
class Point (x0 : Integer) {
  x : Integer = x0;
  function getx () : Integer is {return x}
  function min (other : MyType) : MyType is {
    if (self <= getx()) < (other <= getx()) then {return self} else {return other}
  }
};

class BiPoint (x0 : Integer, y0 : Integer) inherits Point modifies min {
  y : Integer = y0;
  function gety () : Integer is {return y }
  function min (other : MyType) : MyType is {
    if (self <= gety()) < (other <= gety()) then {return self} else {return super.min(other)}
  }
};
```

1. Is the program (in particular, the declaration of `min` in `BiPoint`) well-typed? Briefly motivate the answer.

2. Write declarations for the types `PointType` and `BiPointType` of the objects generated by `Point` and `BiPoint`, respectively.

3. Which will be the type of the four possible calls `x.min(y)` with `x` and `y` of type either `PointType` or `BiPointType`? (be careful!)

4. Write analogous class declarations in Java in such a way that in a call `x.min(y)` the method in `BiPoint` is only invoked when both `x` and `y` have static type `BiPoint`.