1 Concepts in Programming Languages (am21)

This question focuses on concepts – exact syntactic encoding is unimportant.

(a) Explain two possible implementation behaviours of the following program

```ml
let test(n:int) = (let f = (\x:int.n+x); n++; return f(10))
```

Now, give a corresponding Java-like program with `test` being a method containing a lambda and explain how Java resolves this situation.  

(b) Explain two possible implementation behaviours of the following program

```ml
int n = 0; int g() = n; int f(int n) = g(); print(f(1));
```

briefly indicating how `g` would be implemented in each case.  

(c) ML-family languages generally restrict polymorphic exceptions. Consider:

```ml
exception E of ('a->'a);;
try raise (E(fun x->x)) with E(f) -> (f 1, f true);;
```

Giving reasons: (i) would this code type-check in an ML-family language? (ii) would it execute successfully? Give a modified version of the code with similar structure (e.g. retaining two separate function applications to 1 and `true`) but which both type-checks and successfully evaluates to `(1, true)`.  

(d) A blog proclaims “prototypes and virtual method tables are alternative implementations of inheritance”. Clarify what was intended.  

(e) Explain, with reasons, how much a Java compiler can optimise `CreateVec`, given that its definition and its calls may appear in separate Java source files

```java
class IPair { final int x,y; IPair(int X,int Y) { x=X;y=Y; } /* other methods */ };
static IPair[] CreateVec(int n) { var V = new IPair[n]; for (int i=0; i<n; i++) V[i] = new IPair(0,0); return V; }
```

[4 marks]