

1. Consider the following code in a lexically scoped language:

```
let x = 1;  
fun f (x, y) {  
  let y = x+y;  
  fun h (z) { x+y+z };  
  h (y)  
}
```

- (a) Rewrite this code adding subscripts (*e.g.*, x_1) to distinguish the different variables.
 - (b) What are the free variables of `f` and `h`?
2. Recall the discussion of polymorphic typechecking in Handout 4. Assume that we have both `int` and `real` as base types. How might we extend the typechecker to support overloaded functions on integers and reals (*e.g.*, “+”)? You may assume that such functions are predefined, but you should explain how to modify the representation of types and the typechecking algorithm to handle overloaded functions.