Type unification

Find the most general type for all identifiers in the following programs:

1. def fold f l a =
   if l = []
     then a
     else fold f tl(l) (f hd(l) a)
   fi

   a : 'a
   l : 'b list
   f : 'b -> 'a -> 'a
   fold : ('b -> 'a -> 'a) -> 'b list -> 'a -> 'a

2. def y n z = if n < 1 then [] else [z]:(y (n - 1) z) fi
   def f x z = y (1 + (x z)) z

   z : 'a
   x : 'a -> int
   y : int -> 'a -> 'a list
   n : int
   f : ('a -> int) -> 'a -> 'a list

3. def g n h = h (1 + n) h
   def f = (g 0) g

   n: int
   g, h: int -> 'g -> 'r
   f : 'r

x86-64 Assembly

Read the following program and annotate each line with the number of its most plausible-sounding explanation. You can pick from these explanations:

1. Just a label. No instructions generated. 5. Clean current frame (pop dynamic link).
2. Save caller’s frame pointer (dynamic link). 6. Store 0 into the EAX register.
3. Move frame pointer to bottom of stack. 7. Dereference the address from RAX into EAX.
4. Push return address. Jump to some block. 8. Pop (and jump to) the return address.
What does this program do? How would you use a debugger to find problems with it? **Answer:** It dereferences a garbage/null pointer (which usually results in a segmentation fault). You could run it under a debugger, get a backtrace to find the frame in which the fault occurred, disassemble the buggy part of the program, and dump the contents of the registers to check for garbage pointers.