Here is some apims code:

```
(nu list:rec list.
            1→2:1
            { Delete⟦true⟧:
                end,
              Insert⟦true⟧:
                1→2:1⟨Int⟩;
                list,
              Lookup⟦true⟧:
                1→2:1⟨Int⟩;
                2→1:2
                { NONE⟦true⟧:
                    list,
                  SOME⟦true⟧:
                    2→1:2⟨Int⟩;
                    list                    }
             } )
( def List=
    link(2,list,s,2);
    ( List()
    | def NIL(this:  rec list.
                     1≫
                     { Delete:
                         end,
                       Insert:
                         1≫ ⟨Int⟩;
                         list,
                       Lookup:
                         1≫ ⟨Int⟩;
                         2≪
                         { NONE:
                             list,
                           SOME:
                             2≪ ⟨Int⟩;
                             list
                         }
                     }@(2 of 2))=
        this[1]≫
        { Delete:
          end,
          Insert:
          this[1]≫val;
          link(2,list,s,1);
          def CONS(this:  rec list.
```

```
                1≫
                { Delete:
                    end,
                  Insert:
                    1≫ ⟨Int⟩;
                    list,
                  Lookup:
                    1≫ ⟨Int⟩;
                    2≪
                    { NONE:
                        list,
                      SOME:
                        2≪ ⟨Int⟩;
                        list
                    }
                }@(2 of 2),
        next:  rec list.
                1≪
                { Delete:
                    end,
                  Insert:
                    1≪ ⟨Int⟩;
                    list,
                  Lookup:
                    1≪ ⟨Int⟩;
                    2≫
                    { NONE:
                        list,
                      SOME:
                        2≫ ⟨Int⟩;
                        list
                    }
                }@(1 of 2),
        v:  Int)=
this[1]≫
{ Delete:
  next[1]≪ Delete;
  end,
  Insert:
  next[1]≪ Insert;
  this[1]≫newVal;
  if (v <= newVal)
  then next[1]≪newVal;
       CONS(this, next, v)
  else next[1]≪v;
       CONS(this, next, newVal),
```

```
                Lookup:
                this[1]≫pos;
                if (pos <= 0)
                then this[2]≪ SOME;
                     this[2]≪v;
                     CONS(this, next, v)
                else next[1]≪ Lookup;
                     next[1]≪(pos - 1);
                     next[2]≫
                     { NONE:
                       this[2]≪ NONE;
                       CONS(this, next, v),
                       SOME:
                       this[2]≪ SOME;
                       next[2]≫val;
                       this[2]≪val;
                       CONS(this, next, v)                    }          }
           in CONS(this, s, val),
           Lookup:
           this[1]≫pos;
           this[2]≪ NONE;
           NIL(this)        }
      in NIL(s)
    )
  in List()
| link(2,list,myList,1);
  myList[1]≪ Insert;
  myList[1]≪5;
  myList[1]≪ Insert;
  myList[1]≪2;
  myList[1]≪ Insert;
  myList[1]≪7;
  myList[1]≪ Insert;
  myList[1]≪3;
  myList[1]≪ Lookup;
  myList[1]≪2;
  myList[2]≫
  { NONE:
    myList[1]≪ Delete;
    end,
    SOME:
    myList[2]≫val;
    myList[1]≪ Delete;
    end  }
)
```