

IN-LAB DEMO
FRIDAY FEBRUARY 7

Principle of Information Hiding

Supplier:

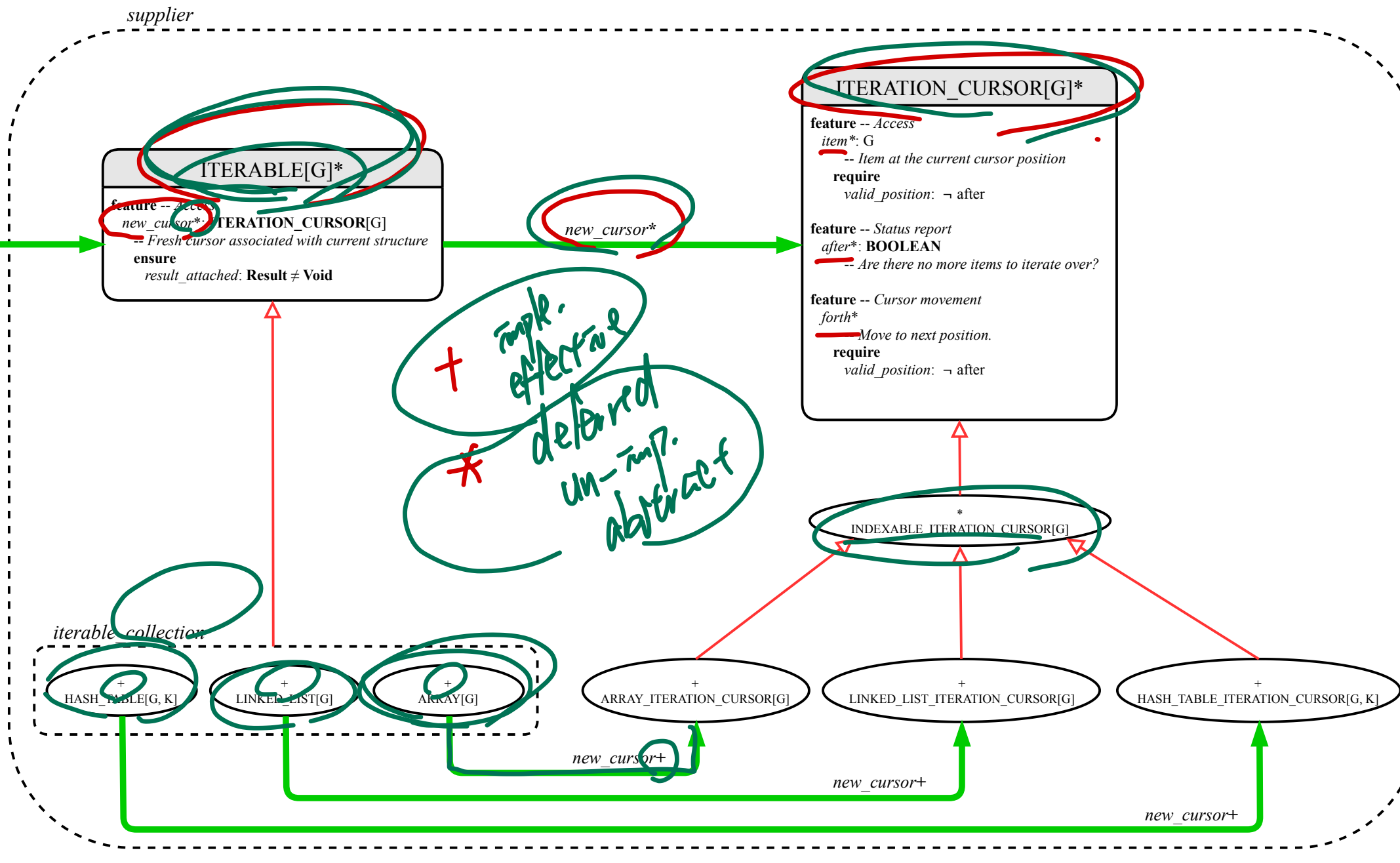
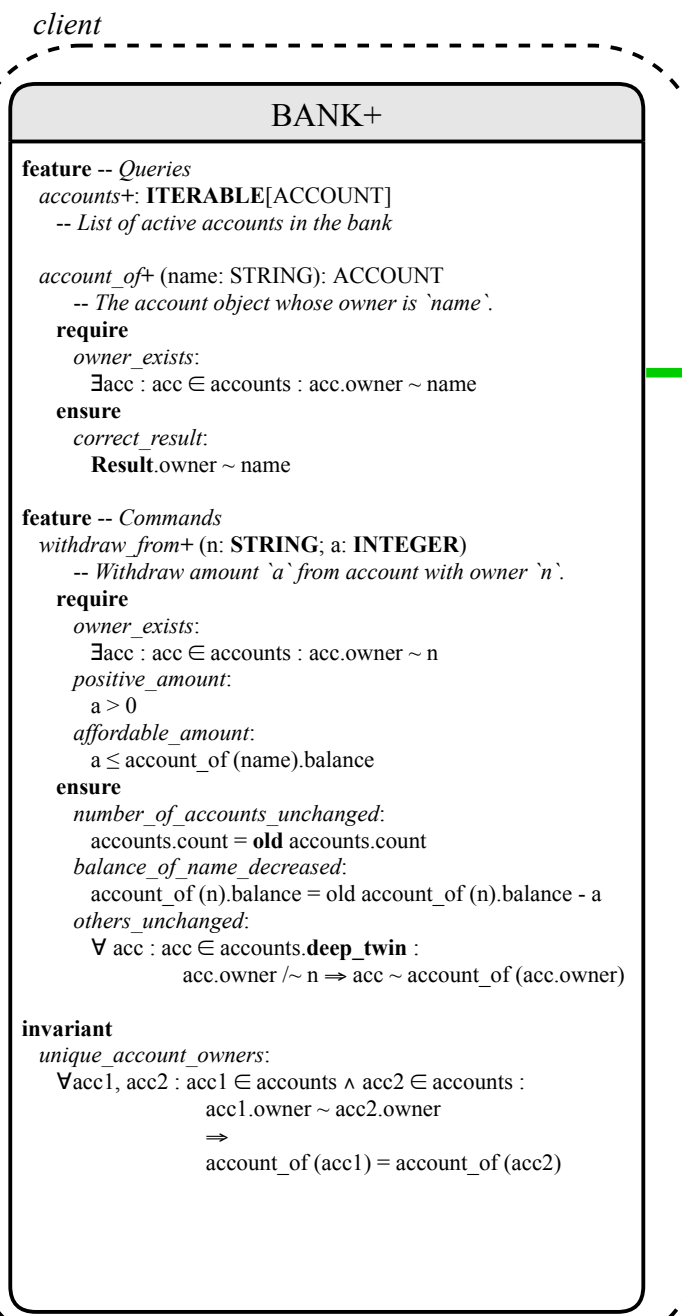
```
class
  CART
  feature
    orders: ARRAY[ORDER]
  end
```

```
class
  ORDER
  feature
    price: INTEGER
    quantity: INTEGER
  end
```

Problems?

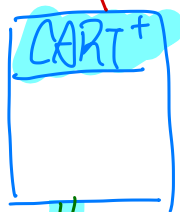
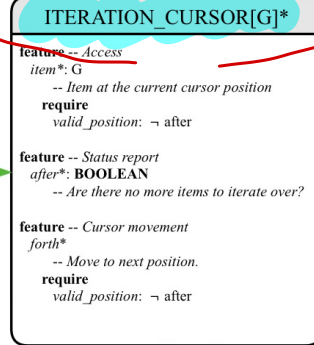
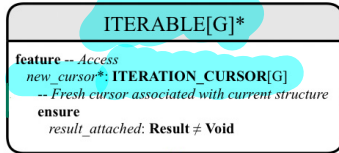
Client:

```
class
  SHOP
  feature
    cart: CART
    checkout: INTEGER
    do
      from
        i := cart.orders.lower
      until
        i > cart.orders.upper
      do
        Result := Result +
          cart.orders[i].price
          *
          cart.orders[i].quantity
        i := i + 1
      end
    end
  end
end
```

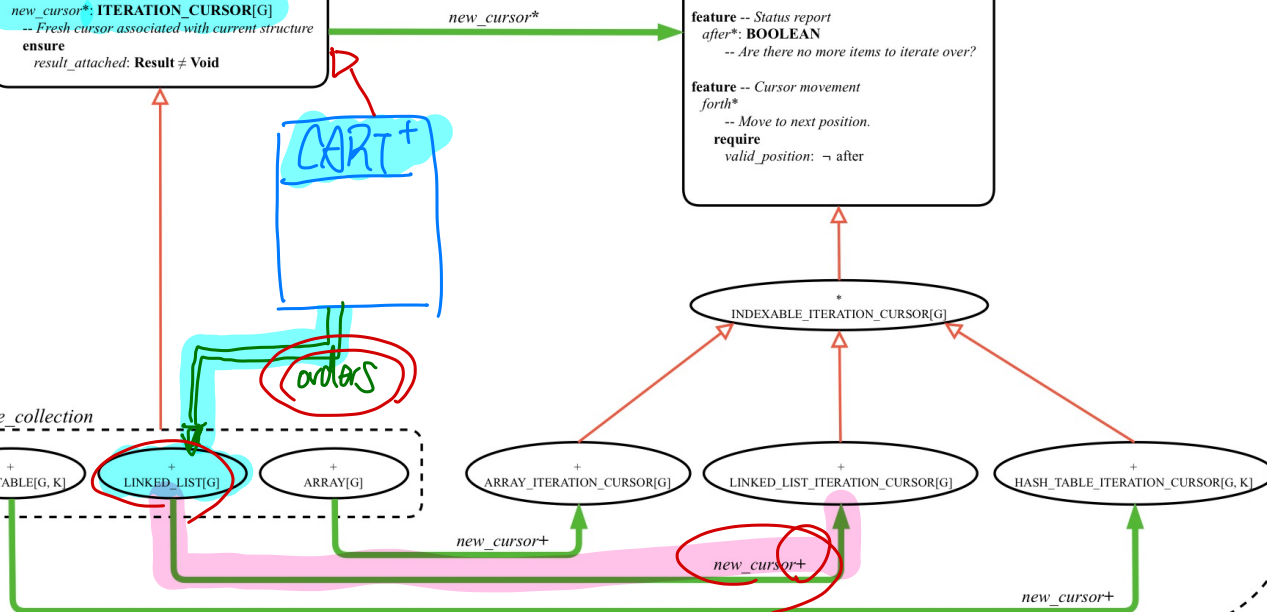

 + imp. effective
 * deferred un-imp. abstract

Implementing the Iterator Pattern: Easy Case

supplier



iterable_collection

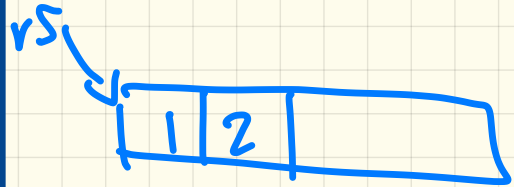


Implementing the Iterator Pattern: Hard Case

```
class
  BOOK[G]

  feature { NONE }
  names: ARRAY[STRING]
  records: ARRAY[G]

end
```



class CART
inherit

ITERABLE

~~ORDER~~

CART :

CART

ITERABLE

across

CART

is

order

of type

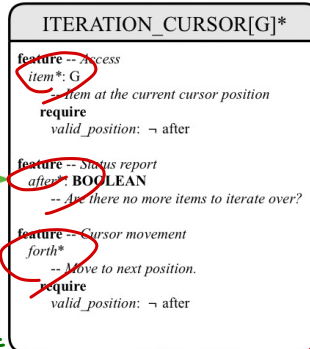
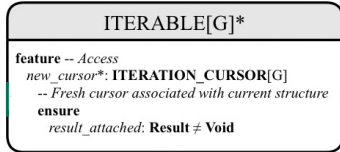
ORDER

loop

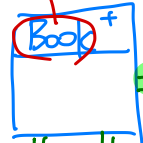
end

Implementing the Iterator Pattern: Hard Case

supplier



TUPLE [_, _]
 1
 2
 []



names
 records

iterable_collection



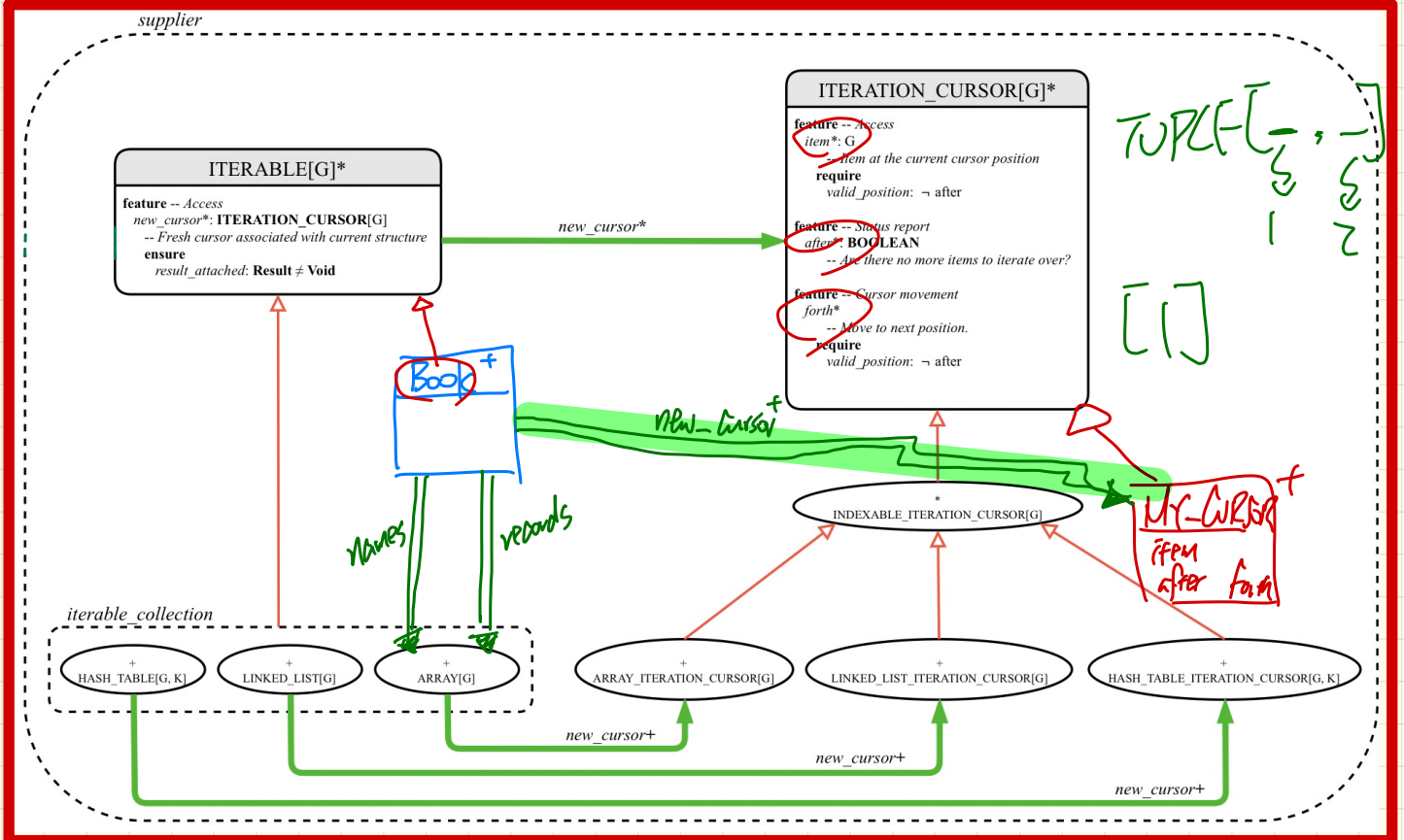
new_cursor+

new_cursor+

new_cursor+

new_cursor*

new_cursor+



class Book(G)

what

b: Book[TYPE]

ITERABLE[??]

TUPLE[STR, G]

??

type?

across b is e

loop

end