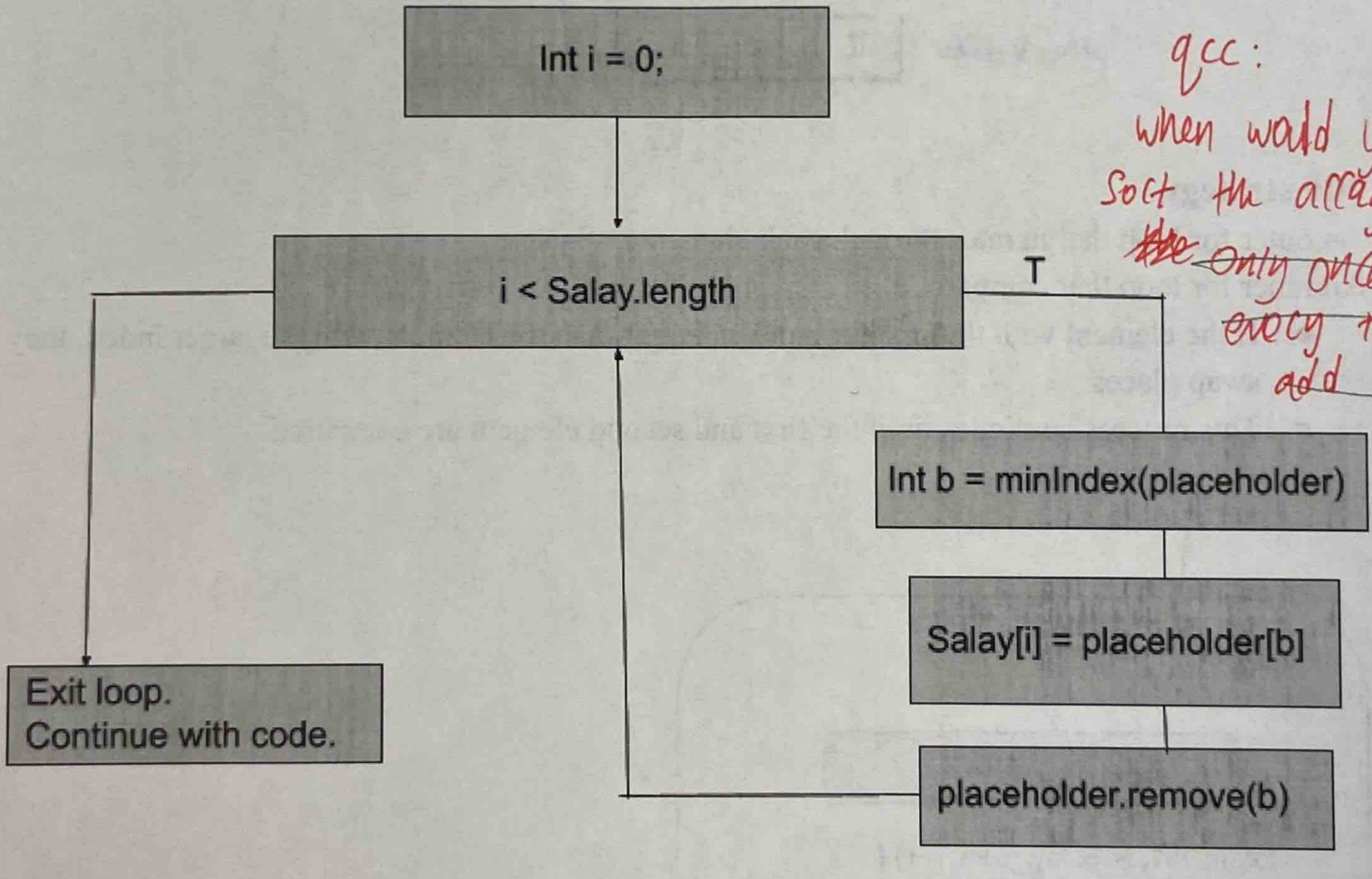
Team Frogs (Ruby Friedman, Ivina Wang, Samantha Hua)
APCS
HW46 -- Wrap the Wrapper
2021-12-09
Time Spent: 1.0 hrs

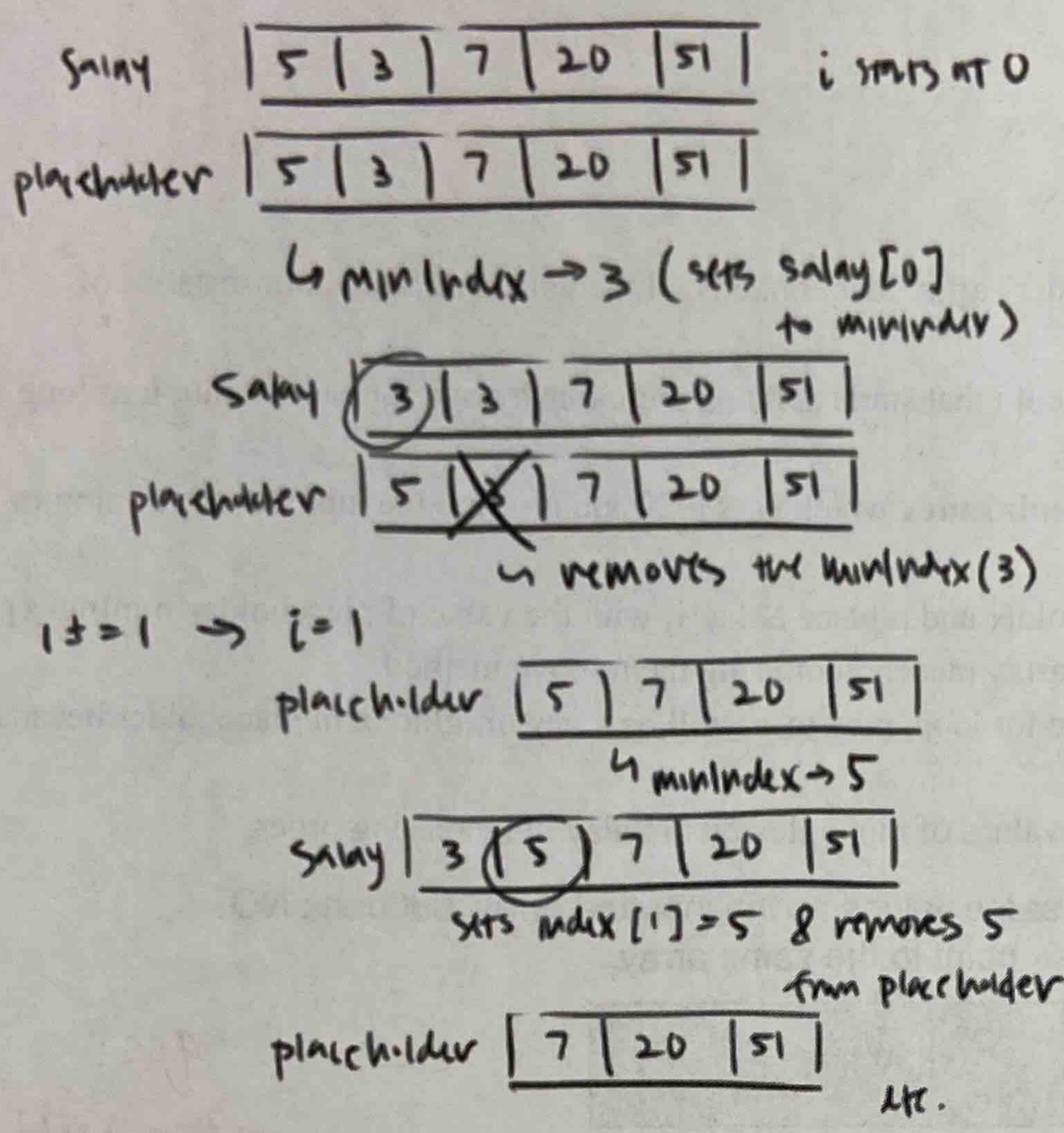
Reviewed by: SWAG SWASH

1st strategy:
Start by creating a "placeholder" array that contains all the values of Salay, our instance of SuperArray.
Use a for loop with a increment i that starts at 0 and increases by one for each iteration as long as i is less than Salay.length.
Use a helper function called **minIndex** which uses a for loop to find the index of the minimum value in a given array.
Find the minIndex of placeholder and replace Salay[i] with the value of placeholder[minIndex].
Remove the minIndex from array placeholder using the remove method.
Continue looping through the for loop, now there will be a new minIndex in placeholder because the old value was removed.
Continue to append the new values of min index on to Salay in increasing order.

Placeholder has the same values as the inputted array, but does NOT point to the same array.



qcc:
when wald you Sort the array? ~~the only once~~ or ~~every time you add something~~
never

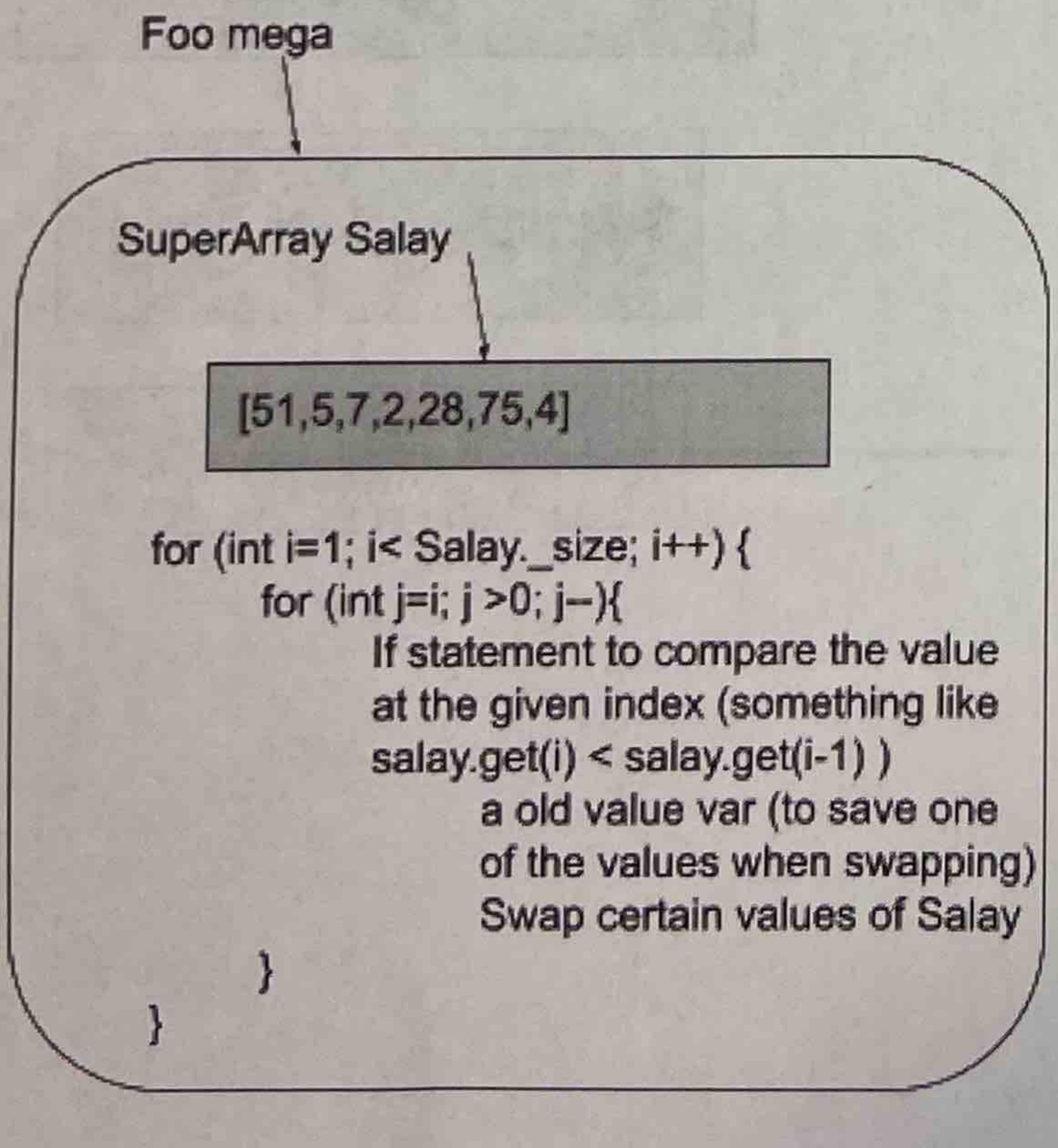

Salay | 5 | 3 | 7 | 20 | 51 |
i starts at 0
placeholder | 5 | 3 | 7 | 20 | 51 |
↳ minIndex → 3 (sets salay[0] to minindex)
Salay | 3 | 3 | 7 | 20 | 51 |
placeholder | 5 | 3 | 7 | 20 | 51 |
↳ removes the minIndex (3)
i+=1 → i=1
placeholder | 5 | 7 | 20 | 51 |
↳ minIndex → 5
Salay | 3 | 5 | 7 | 20 | 51 |
sets index [1] = 5 & removes 5 from placeholder
placeholder | 7 | 20 | 51 |
etc.

2nd strategy

An outer for loop that iterates through each element in Salay

An inner for loop that compares each element to the one that comes before it

- If the element with the smaller index is larger than the element with the larger index, they swap places
- This process continues until the first and second element are compared



```
for (int i=1; i< Salay._size; i++) {
        for (int j=i; j >0; j--){
                If statement to compare the value
                at the given index (something like
                salay.get(i) < salay.get(i-1) )
                        a old value var (to save one
                        of the values when swapping)
                        Swap certain values of Salay
        }
}
```

salay

51	5	7	2	28	75	4

int i=1 compared these, (1st for loop)

and 51 > 5 → swap values

salay

5	51	7	2	28	75	4

← nothing is swapped from the 2nd loop ← yet

i=2

compare 51 < 7 → swap values

salay

5	7	51	2	28	75	4

i=3
j=3 compare 51 < 2 → swap values

salay

5	7	2	51	28	75	4

i=3
j=2

* second for loops goes backwards to check if the previous elements are in order

7 > 2 → swap values

salay

5	2	7	51	28	75	4

compares i=3
j=1

5 > 2 → swap

salay

2	5	7	51	28	75	4

i=3
j=0 → stops 2nd for loop

continues back to first for loop and 51 < 28, etc.

{ }

add(3)

{3}

add(5)

{3,5}

add(4)

{3,4,5}

the sorting should return the index at which the new thing should be added