Q3. a)

```
 1. A <- [x : (P[i], W[i]), 0 <= i < size(P)]   // Group into a (p, w) tuple
for each ball
 2. B <- [x : groupBy(A, 0)]                    // Group weights by price to
get [p0: [w0..wn], .. pn: [w0..wn]]
 3. sort(B)                                     // Sort B by price (key)
 4. result <- []                                // Initialize result list
 5. low                                         // Declare lowest weight int
 6. for i <- 0 to size(B)-1 do {
 7.     subset = []                             // Set of balls to be added
to result
 8.     if i == 0 {
 9.         subset = B[pi]                      // Add all balls with lowest
price
10.     } else {
11.         // Add all balls with weight lower than previous low
12.         subset = [x : (pi, B[pi][y]), 0 <= y < size(B[pi]), B[pi][y] <=
low]
13.     }
14.     low <- min(subset)                      // Get new lowest weight
15.     result += subset                        // Add to result
16. }
```

Sorting complexity: `θ(n logn)`, loop complexity: `θ(n)`

Therefore, solution complexity: `θ(n logn)`

Q3. b)

```
 1. B <- [1: [], .. 200: []]            // Intialize list of lists with
given price range
 2. for i <- 0 to size(W)-1 do
 3.     B[P[i]] += W[i]                 // Add all weights to B to get [p0:
[w0..wn], .. pn: [w0..wn]]
 4. removeEmptyLists(B)                 // Remove keys with no weights
 5. result <- [], low                   // Declare result list and lowest
weight int
 6. for i <- 0 to size(B)-1 do {
 7.     subset = []                     // Set of balls to be added to
result
 8.     if i == 0 {
 9.         subset = B[pi]              // Add all balls with lowest price
10.     } else {
11.         // Add all balls with weight lower than previous low
12.         subset = [x : (pi, B[pi][y]), 0 <= y < size(B[pi]), B[pi][y] <=
low]
13.     }
14.     low <- min(subset)              // Get new lowest weight
15.     result += subset                // Add to result
16. }
```

Loop complexity for all loops: `θ(n)`

Therefore, solution complexity: `θ(n)`