CÂU 1:

Dùng Linear Search để duyệt qua từng phần tử của mảng, nếu có phần tử bằng K thì trả về index của phần tử đó. Nếu không tìm thấy, trả về -1.

Các bước thực hiện:

1. Duyệt mảng từ phần tử đầu tiên đến phần tử cuối cùng.
2. Kiểm tra từng phần tử xem có bằng K không.
3. Nếu tìm thấy, trả về chỉ số của phần tử đó.
4. Nếu duyệt hết mà không tìm thấy, trả về -1.

CÂU 2:

Dùng Linear Search với Sentinel bằng cách thêm K vào cuối mảng làm phần tử đánh dấu, giúp giảm số lần kiểm tra điều kiện. Khi tìm thấy K, kiểm tra xem đó có phải là Sentinel không. Nếu đúng, trả về -1, nếu không, trả về index của K.

Các bước thực hiện:

1. Thay thế phần tử cuối cùng của mảng bằng K
2. Duyệt mảng bằng cách cho vòng lặp while chạy đến khi nào tìm được K
3. Khi tìm thấy K, kiểm tra xem đó có phải là Sentinel không.
4. Có 2 trường hợp: Nếu K được tìm thấy là Sentinal và phần tử cuối cùng của mảng lúc đầu khác K thì trả về -1. Còn nếu K được tìm thấy không phải là Sentinal hoặc là Sential nhưng phần tử cuối cùng của mảng lúc đầu bằng K thì trả về index của K

CÂU 3:

Dùng tìm kiếm nhị phân để tìm phần tử nhỏ nhất trong mảng xoay.

Các bước thực hiện:

1. Xác định chỉ số trái (left) và phải (right) của mảng.
2. Lặp lại quá trình:
    - Tính chỉ số mid = (left + right) / 2.
    - Nếu mid > phần tử cuối, phần nhỏ nhất nằm bên phải → cập nhật left = mid + 1.
    - Nếu mid < phần tử cuối, phần nhỏ nhất nằm bên trái → cập nhật right = mid.
3. Lặp lại cho đến khi left == right, khi đó phần tử nhỏ nhất là nums[left].

CÂU 4:

Dùng Binary Search để tìm giới hạn tải trọng nhỏ nhất có thể vận chuyển trong số ngày cho phép.

Các bước thực hiện:

1. Xác định phạm vi tải trọng hợp lý:
    - Giá trị nhỏ nhất là low (là giá trị của gói hàng lớn nhất vì ít nhất phải chở được gói hàng lớn nhất).
    - Giá trị lớn nhất là high (là tổng tất cả các gói hàng trong trường hợp phải chở tất cả trong một ngày).
2. Thực hiện tìm kiếm nhị phân trên khoảng này:
    - Giả định tải trọng trung bình mid
    - Kiểm tra xem có thể chuyển hàng trong số ngày yêu cầu với tải trọng mid không. (tạo một hàm nhỏ để kiểm tra)
    - Nếu có thể, thử tải trọng nhỏ hơn.
    - Nếu không, tăng tải trọng.
3. Kết thúc khi low = high là giá trị nhỏ nhất của tải trọng cần để chở hàng trong số ngày qui định

CÂU 5:

Dùng kỹ thuật Two Pointers để tìm mảng con ngắn nhất có tổng lớn hơn hoặc bằng target.

Các bước thực hiện:

1. Khởi tạo hai con trỏ left và right tại vị trí đầu mảng.
2. Duyệt mảng, mở rộng cửa sổ (right tăng) đến khi tổng >= target.
3. Khi tổng thỏa mãn điều kiện, thu hẹp cửa sổ (left tăng) để tìm độ dài nhỏ nhất.
4. Cập nhật kết quả và tiếp tục lặp đến hết mảng.
5. Nếu không tìm thấy, trả về 0.

CÂU 6:

Dùng hai con trỏ để kiểm tra xem có hai phần tử nào trong mảng có tổng bằng target không.

Các bước thực hiện:

1. Nếu mảng chưa được sắp xếp, sắp xếp mảng trước.
2. Khởi tạo hai con trỏ left (đầu mảng) và right (cuối mảng).

3. Kiểm tra tổng nums[left] + nums[right]:
    - Nếu bằng target, trả về YES.
    - Nếu nhỏ hơn target, tăng left.
    - Nếu lớn hơn target, giảm right.
4. Lặp đến khi left >= right.
5. Nếu không tìm thấy, trả về NO.

CÂU 7:

Dùng kỹ thuật Two Pointers sau khi sắp xếp mảng để tìm các bộ ba phần tử có tổng bằng 0 mà không trùng lặp.

Các bước thực hiện:

1. Sắp xếp mảng theo thứ tự tăng dần.
2. Duyệt từng phần tử làm phần tử cố định nums[i].
3. Với mỗi nums[i], dùng hai con trỏ left và right để tìm hai số còn lại sao cho tổng = 0:
    - Nếu tổng = 0, xuất ra màn hình bộ 3 , tăng left và right (tránh trùng lặp bằng cách cho 2 vòng lặp while với điều kiện nếu left = left +1 thì left++ , right = right -1 thì right--).
    - Nếu tổng < 0, tăng left.
    - Nếu tổng > 0, giảm right.
4. Bỏ qua các phần tử trùng lặp để tránh kết quả trùng.(nếu nums[i] = nums[i+1] thì continue;)