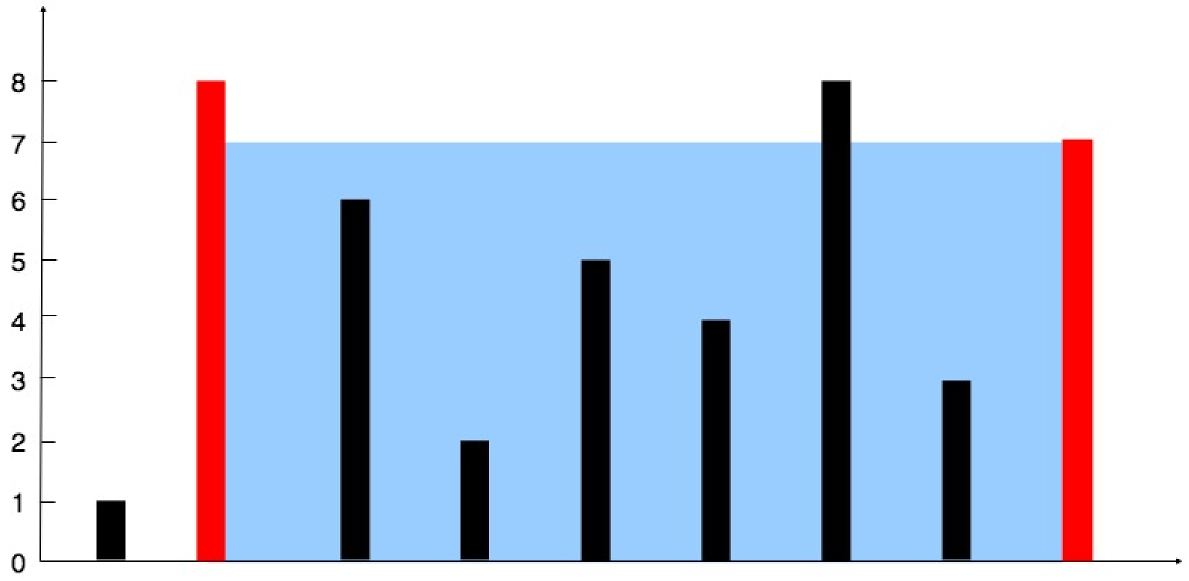
Container With Most Water

You are given an integer array height of length n. There are n vertical lines drawn such that the two endpoints of the i^{th} line are (i, 0) and (i, height[i]).

Find two lines that together with the x-axis form a container, such that the container contains the most water.

Return *the maximum amount of water a container can store*.

Notice that you may not slant the container.



Input: height = [1,8,6,2,5,4,8,3,7]

Output: 49

Explanation: The above vertical lines are represented by array [1,8,6,2,5,4,8,3,7]. In this case, the max area of water (blue section) the container can contain is 49.

```
#include <iostream>
#include <vector>

#include <sstream>
using namespace std;

int maxArea(vector<int>& height) {
  //Write your logic here
  return max_area;
}

int main() {
  vector<int> height;
  int h;
  // Read heights from one line separated by spaces
  string line;
  getline(cin, line);
```

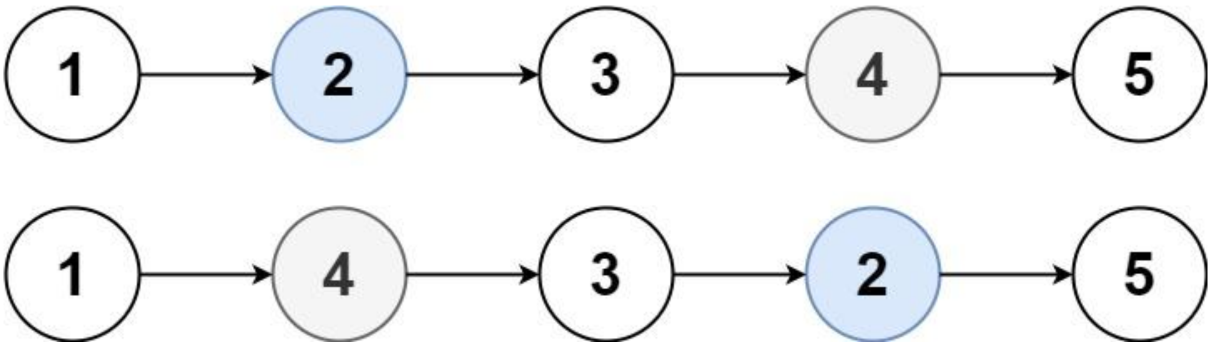
```
    istringstream ss(line);
    while (ss >> h) {
        height.push_back(h);
    }

    cout << maxArea(height) << endl;
    return 0;
}
```

Swapping Nodes in a Linked List

You are given the head of a linked list, and an integer k.

Return *the head of the linked list after **swapping** the values of the* kth *node from the beginning and the* kth *node from the end (the list is **1-indexed**).*

Example 1:



Input: head = [1,2,3,4,5], k = 2

Output: [1,4,3,2,5]

```
#include <iostream>

#include <vector>

#include <sstream>

using namespace std;


// Definition for singly-linked list node

struct ListNode {
    int val;
    ListNode* next;
    ListNode(int x): val(x), next(NULL) {}
};

// Convert a vector to a linked list
ListNode* vectorToList(const vector<int>& nums) {
    ListNode* dummy = new ListNode(0);
    ListNode* curr = dummy;
    for (int num : nums) {
        curr->next = new ListNode(num);
        curr = curr->next;
    }
    return dummy->next;
}

// Print the linked list as comma-separated values
void printList(ListNode* head) {
    while (head) {
        cout << head->val;
        if (head->next) cout << " ";
        head = head->next;
    }
    cout << endl;
}

// Swap kth node values
ListNode* swapNodes(ListNode* head, int k) {


    return head;
}
```

```
int main() {
    string line;
    getline(cin, line);       // Read the entire line for node values
    vector<int> nums;
    stringstream ss(line);
    int num;
    while (ss >> num) nums.push_back(num);
    int k;
    cin >> k;                 // Read k

    ListNode* head = vectorToList(nums);
    head = swapNodes(head, k);
    printList(head);
    return 0;
}
```