2025-10-29/pancake-sorting.cpp

```cpp
#include <iostream>
using namespace std;

// Function to flip the array from index k to the end
void flip(int arr[], int k, int n) {
    int end = n - 1;
    while (k < end) {
        swap(arr[k], arr[end]);
        k++;
        end--;
    }
}

// Function to print the array
void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;
}

// Function to find the index of the maximum element in the array up to a given size
int findMaxIndex(int arr[], int currIndex, int n) {
    int maxIndex = currIndex;
    for (int i = currIndex; i < n; i++) {
        if (arr[i] > arr[maxIndex]) {
            maxIndex = i;
        }
    }
    return maxIndex;
}

// Pancake sort function
void pancakeSort(int arr[], int n) {
    for (int curr_size = n; curr_size > 0; curr_size--) {
        int correctIndex = n - curr_size;
        // Find the index of the maximum element in the current unsorted portion
        int maxIndex = findMaxIndex(arr, correctIndex, n);
        // If the maximum element is not already at the start of the current unsorted portion
        if (maxIndex != correctIndex) {
            // Bring the maximum element to the end of the current unsorted portion
            flip(arr, maxIndex, n);
            flip(arr, correctIndex, n);
        }
    }
}

```

```cpp
int main() {
    int arr[] = {5, 3, 4, 1, 6, 2};
    int n = sizeof(arr) / sizeof(arr[0]);

    cout << "Original array: ";
    printArray(arr, n);

    pancakeSort(arr, n);

    cout << "Sorted array: ";
    printArray(arr, n);

    return 0;
}
```