FILE: PancakeSort.cpp
TITLE: Pancake Sort Documentation
SUBMITTED BY: ISDL04
FOR COURSE: Integrated Software Development Lab
DUE DATE: 30th November, 2020

TABLE OF CONTENTS:

PURPOSE:

This program will sort the elements of an array in ascending order iteratively and recursively using the Pancake Sorting Algorithm. It will read from the user, the number of test cases and for each test case, the dimension of the array and its elements. The sorted array will be then printed.
The code handles the array of elements as a stack of pancakes which are to be flipped in order to stack them in increasing order of size, with the smallest pancake on the top.

OVERALL TASK:

The list of general tasks is:

1. Input number of test cases from the user.
2. Input array dimension and its elements for each test case.
3. Array is passed to the sorting method.
4. The sorted array is then displayed on the output console by the output function.

INCLUDED FILES:

1. iostream.h
2. limits.h
3. time.h
4. cstdlib.h

INCLUDED MACROS:

1. ll long long int
2. fastIO ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL)

DATA FILES:

None

CLASS: Helper

INHERITED BY: Pancake

INCLUDED METHODS:

1.name OF METHOD: **max**

SCOPE: Public

PURPOSE: The method returns the index of the maximum element from the elements of the array till index 'n'.

PARAMETERS:

name	**type**	**value/reference**	**description**
n	long long int	value	Index till which array has to be traversed
a[]	long long int	reference	The array of the elements

RETURN VALUE:

name	**type**	**value/reference**	**description**
index	long long int	value	the index of the maximum element upto index n.

CALLS TO: none

CALLED FROM: pancakeIter, pancakeRec

PSEUDOCODE:

```
1.   Initialize the temporary variable m to INT_MIN and index
     to -1.
2.   Loop and execute n times.
   2.1.   Use a loop counter as subscript of array.
   2.2.   If m is less than the current array element.
      2.2.1.   Store current array element in m and the loop
               counter in index.
3.   Return index.
```

2.name OF METHOD: **swap**

SCOPE: Public

PURPOSE: The method interchanges the values stored at the input references.

PARAMETERS:

name	**type**	**value/reference**	**description**
a	long long int	reference	Input 1
b	long long int	reference	Input 2

RETURN VALUE: void

CALLS TO: none

CALLED FROM: flip

PSEUDOCODE:

```
1.  Initialize the temporary variable c with 0.
2.  Store the value at reference a into variable c.
3.  2Store the value at reference b into reference at a.
4.  Store the value of variable c into reference at b.
```

3.name OF METHOD: **flip**

SCOPE: Public

PURPOSE: This function reverses the elements of input array between the index range of 0 to i

PARAMETERS:

name	**type**	**value/reference**	**description**
a[]	long long int	reference	The array of elements
i	long long int	value	Index till which array has to be reversed

RETURN VALUE: void

CALLS TO: swap

CALLED FROM: pancakeIter, pancakeRec

PSEUDOCODE:

```
1.  Initialize the temporary variable k with 0 and temporary
    variable j with i.
2.  Loop and execute till k is less than j.
   2.1.  Swap the array elements at index j and k.
   2.2.  Increment k by 1.
   2.3.  Decrement j by 1.
```

<u>CLASS: Pancake</u>

<u>INHERITED BY:</u> none

<u>INCLUDED METHODS:</u>

1.name OF METHOD: **pancakeRec**

SCOPE: Public

PURPOSE: The method sorts the array elements recursively using Pancake Sorting Algorithm.

PARAMETERS:

name	**type**	**value/reference**	**description**
n	long long int	value	Dimension of array to be sorted
a[]	long long int	reference	The array of the elements

RETURN VALUE: void

CALLS TO: max, flip, pancakeRec

CALLED FROM: driver,test

PSEUDOCODE:

```
1.  Set the base condition of n=0,to return to the calling
    function.
2.  Store the output of method max(a,n-1) to i.
3.  If i is less than (n-1):
```

    3.1. Call **flip(a,i)**,To bring the maximum element to the top of the input array.
    3.2. Call **flip(a,n-1)**,To bring the maximum element to the bottom of the input array.

4. Recursively call the method with input **a** and **n-1**.

2.name OF METHOD: **pancakeIter**

SCOPE: Public

PURPOSE: The method sorts the array elements iteratively using Pancake Sorting Algorithm.

PARAMETERS:

name	**type**	**value/reference**	**description**
n	long long int	value	Dimension of array to be sorted
a[]	long long int	reference	The array of the elements

RETURN VALUE: void

CALLS TO: flip, max

CALLED FROM: driver, test

PSEUDOCODE:

```
1.   Initialize i with n-1 .
2.   Loop and execute till i is not less than 0.
   2.1.   Store the output of method max(a,i) to j.
   2.2.   If j is less than i:
      2.2.1.   Call flip(a,j),to bring the maximum element to
               the top of the array.
      2.2.2.   Call flip(a,i),to bring the maximum element to
               the bottom of the subarray from index 0 to i.
```

INCLUDED FUNCTIONS:

1. name OF FUNCTION: **sortedPrint**

PURPOSE: This function outputs the sorted array elements.

PARAMETERS:

name	**type**	**value/reference**	**description**
n	long long int	value	Dimension of array to be sorted
a[]	long long int	reference	The array of the elements

RETURN VALUE: void

CALLS TO: none

CALLED FROM: test,driver

PSEUDOCODE:

```
1.   Initialize the temporary variable i to 0 and use it as a
     loop counter.
2.   Loop and execute n times.
   2.1.   Use a loop counter as subscript of array.
   2.2.   Output the array element.
```

2. name OF FUNCTION: **driver**

PURPOSE: This function inputs from the user the array dimension and its elements.

PARAMETERS: none

RETURN VALUE: void

CALLS TO: pancakeIter,pancakeRec

CALLED FROM: main

PSEUDOCODE:

```
1.  Initialize the array dimension, n to zero.
2.  Prompt user for data expected(n).
3.  Loop and execute n times.
   3.1.  Use loop counter as subscript of array.
   3.2.  Prompt user for specific array element data.
4.  Create an instance of class Pancake, p.
5.  Use p to call the methods pancakeRec or pancakeIter of
    Pancake class to sort the array.
6.  Call sortedPrint to print the sorted array.
```

3. name OF FUNCTION: **test**

PURPOSE: This function inputs from the user the array dimension and its elements.

PARAMETERS: none

RETURN VALUE: void

CALLS TO: pancakeIter,pancakeRec

CALLED FROM: main

PSEUDOCODE:

```
1.   Set srand(time(NULL)) to make the seed correspond to the
     clock of the system, so it changes the sequence of
     random numbers generated for each run of the program.
2.   Initialize the array dimension, n to zero.
3.   Prompt user for data expected(n).
4.   Loop and execute n times.
  4.1.   Use predefined function rand() to generate the
         array element.
  4.2.   Output the generated array element.
5.   Create an instance of class Pancake, p.
6.   Use p to call the methods pancakeRec or pancakeIter of
     Pancake class to sort the array.
7.   Call sortedPrint to print the sorted array.
```

4. name OF FUNCTION: **main**

PURPOSE: This function is responsible for the execution of the code to start.

PARAMETERS: none

RETURN VALUE: 0

CALLS TO: driver, test

CALLED FROM: Operating System

PSEUDOCODE:

```
1. Initialize a variable t, which denotes number of test
   cases.
2. Prompt user for data expected(t).
3. Loop and execute till t is not equal to zero
   3.1 Call the driver function, if input is to be taken
   from the user.
   3.2 Call the test function, if a randomised array to be
   generated.
```

APPLICATIONS:

1. Pancake sorting is different from traditional comparison-based sorting algorithms. The goal here is to sort the input array in a minimum number of reversals/flips.
2. It is used when the only allowed operation to sort an input array is reversing/flipping.
3. It also appears in applications in parallel processor networks, in which it can provide an effective routing algorithm between processors.

RISKS:

1. If the number of input elements is less than the array dimension, then the suggested compiler will copy the element which is the last input to the function till the array dimension is achieved.
2. If the user inputs a non-positive array dimension more than twice, the program will terminate.