SE 1105 Fall 2021

Project II

Yigithan Yigit **20070001033**

Project Description:

Write a function which is includes weirdSort, readArray, printArray and weirdPrint. Weird sort kind a sorting algorithm but it sorts elements with even indexes in increasing order and elements with odd indexes in decreasing order. readArray is gets array and fill array from user, printArray is simply prints array and final one weirdPrint printing weirdSort output in two seperate lines where in the first line it prints even indexes and in the second line prints the elements with odd indexes.

Project Solution:

weirdSort is the hard one in my opinion, first of all i read inputs and send it to weirdSort. In weirdSort i created 2 arrays (arrOdd, arrEven) and seperated odd indexes into arrOdd, even ones into arrEven after that i wrote a basic sorting algorithm and then merge in one array with reverse order the arrOdd. In final i printed all of them.

Implementation:

```c
#include <stdio.h>

void weirdSort(int arr[], int size)
{
    /*
    Takes an integer array and its size as parameters. Sorts the elements with
even
    indexes in increasing order and the elements with odd indexes in decreasing
```

```
order. Note that after sorting, the elements with even indexes should still
remain at even indexes and the elements with odd indexes should remain at
odd indexes (See the example below)
*/
int arrOdd[5];
int arrEven[5];
int indexOdd = 0, indexEven = 0;
for (int i = 0; i < size; i++)
{
    // Split even and odd indexes to diffrent arrays
    if (i % 2 == 0)
    {
        arrEven[indexEven] = arr[i];
        indexEven += 1;
    }
    else if (i % 2 == 1)
    {
        arrOdd[indexOdd] = arr[i];
        indexOdd += 1;
    }
}

int i, j, tempEven, tempOdd;
for (i = 1; i < size - (indexEven); i++)
{
    tempEven = arrEven[i];
    for (j = i - 1; j >= 0 && tempEven <= arrEven[j]; j--)
    {
        arrEven[j + 1] = arrEven[j];
    }
    arrEven[j + 1] = tempEven;

    tempOdd = arrOdd[i];
    for (j = i - 1; j >= 0 && tempOdd <= arrOdd[j]; j--)
    {
        arrOdd[j + 1] = arrOdd[j];
    }
    arrOdd[j + 1] = tempOdd;
}

int reverseIndex = indexOdd - 1;
indexOdd = 0;
indexEven = 0;
for (int i = 0; i < size; i++)
{
```

```
        // merge two arrays
        if (i % 2 == 0)
        {
            arr[i] = arrEven[indexEven];
            indexEven += 1;
        }
        else if (i % 2 == 1)
        {
            arr[i] = arrOdd[reverseIndex - indexOdd];
            indexOdd += 1;
        }
    }
}

void readArray(int arr[], int size)
{
    /*
    Takes an integer array and its size as parameters. The function must fill the
    array with integers read from the user.
    */
    for (int i = 0; i < size; i++)
    {
        int arrElement;
        printf("Please Type a integer that you want to put in array\n");
        scanf_s(" %d", &arrElement);
        arr[i] = arrElement;
    }
}

void printArray(int arr[], int size)
{
    /*
    Takes an integer array and its size as parameters. Prints the contents of the
    array in a single line separated by spaces
    */
    for (int i = 0; i < size; i++)
    {
        printf("%d ", arr[i]);
    }
    printf("\n");
}
```

```c
void weirdPrint(int arr[], int size)
{
    /*
    Takes an integer array and its size as parameters. Prints the contents of the
    array in two separate lines where in the first line it prints the elements
with
    even indexes and in the second line it prints the elements with odd indexes.
    */
    for (int i = 0; i < size; i++)
    {

        if (i % 2 == 0)
        {
            printf("%d ", arr[i]);
        }
    }
    printf("\n");
    for (int i = 0; i < size; i++)
    {

        if (i % 2 == 1)
        {
            printf("%d ", arr[i]);
        }
    }
}

void main()
{
    /*
    reads an array of size 10 by using readArray function. Sorts the array by
    using weirdSort function. Prints the array by using weirdPrint function
    */
    int arr[10];
    int size = 10;
    readArray(arr, size);
    weirdSort(arr, size);
    printArray(arr,size);
    weirdPrint(arr, size);
}
```

Output of The Program:

If the array is { 2,3,11,31,1,7,16,35,48,4 }

The result of printing should be as:

1 35 2 31 11 7 16 4 48 3

The result of weird printing should be as:

1 2 11 16 48

35 31 7 4 3

Conclusion:

It was helpful about understanding sorting concepts and understanding array structure, it was a summary of what we have done so far.