Homework 3

Negin Mohammadi

Your task is to read a positive integer N and the elements of an array of size N from the standard input and sort the array in ascending order using merge sort. The sorting is done using a C function named **`mergeSort`**.

```
void mergeSort(int arr[], int l, int r)
```

The function above takes an array `arr` and sorts the array from the index `l` to the index `r` (inclusive). Thus, calling `mergeSort(arr, 0, size-1)` sorts the whole array (of length `size`). This function must be written in a separate file called **`sort.c`**.

To merge two sorted arrays the function `mergeSort` must call an assembly function named `merge`.

```
void merge(int arr[], int l, int m, int r);
```

It merges the (sorted) subarray from `l` to `m` with the subarray from `m+1` to `r` (inclusive). Write the function `merge` in a file called **`merge.asm`**. You also need to create a Makefile that assembles, compiles and links the source files and creates an executable named **run.out**. Your program therefore must contain:

- **main.c** (given to you)
- **sort.c** (containing the C function `mergeSort`)
- **merge.asm** (containing the assembly function `merge`)
- **Makefile**

You will get 70% of the score by just writing the 32-bit version of the assembly function. To receive the full score you need to also implement the 64-bit version of the function `merge`. Put the two versions in two separate folders named “32” and “64”. Submit a .**zip** file containing both folders exactly structured as the template provided provided to you.

Your code **must** comply with the following rules:

- You must observe all the **C calling conventions**.
- Your code must work with the provided **main.c** file. **Do not change it**.
- You can only use the commands you have learned so far in the class.
- You MUST NOT PRINT ANY REDUNDANT OUTPUT. Results might be checked by script.

Remember that your code will be checked for similarity. In the case of cheating the student will receive a **negative** point. It is your responsibility to protect your code.

Please create the **.zip** file as explained and upload it on vc.kntu.ac.ir.

Example:

Input 1:
3
1 4 3

Output 1:
1 3 4

Input 2:
4
1 1 1 1

Output 2:
1 1 1 1

Input 3:
5
5 4 2 3 1

Output 3:
1 2 3 4 5

Input 4:
10
5 6 4 2 7 9 8 9 3 1

Output 4:
1 2 3 4 5 6 7 8 9 9