CS3101: Class Test 4

Duration: 2 hours **Total marks: 20**

Instructions

- All programs should be compatible with GNU C compiler.
- There will be a plagiarism check. The suspected copies will attract a deduction of marks for the problem irrespective of who is the original author of the code. Copying from internet will also be considered as plagiarism.
- Late submissions will attract marks deduction.
- You are advised to compile the code before submitting them to WeLearn. If a code does not compile at our end, marks will be deducted for the problem.
- Each program should follow a strict naming convention: **QNo.c** (e.g. Q1.c, Q2a.c etc.). Programs not adhering to the convention will not be corrected.
- All codes (.c files) should be submitted in a single zipped folder (folder name containing your WeLearn ID e.g. 17MS001) to WeLearn.
- You should put appropriate comments in the code.
- You may reuse the codes shared by the instructor.

1. **(Marks: 3)** Write a C program to store the abbreviation (considering the first letter of each word) of a string s, ignoring the words *of* and *and*, of a string s to a string s_out and print s_out on the terminal. Note that although the string s can be in lower or upper case (or in a mixture of both), s_out should always be in the upper case. E.g.if s is *The United States of America*, s_out should be *USA*; if s is *please find attached*, s_out should be *PFA* etc.

   [FYI: For a *char* type variable c, `tolower(c)` converts c to lower case, `toupper(c)` converts c to upper case; `tolower(c)` and `toupper(c)` are defined in the header file `ctype.h`]

2. **(Marks: 6)** Write a C program to implement the sorting algorithm described as below:

   For an array $a[0..n-1]$ (to be taken as input; n, a number of your choice, is the number of elements in the array)

   - In step 1, the smallest element in $a[0..n-1]$ is placed in $a[0]$; In step 2, the smallest element in $a[1..n-1]$ is placed in $a[1]$,..., in step n - 1, the smallest element in $a[n-2..n-1]$ is placed in $a[n-2]$
   - In general, in step i, the smallest element in $a[i-1..n-1]$ is placed in $a[i-1]$
   - In step i, the original element in $a[i-1]$ is swapped with the smallest element. That is, if $minpos$ is the position of the smallest element in $a[i-1..n-1]$, $a[i-1]$ and $a[minpos]$ are swapped.

   If the array is 5, 1, 3, 2, 4; the steps are as below:

   1. 1, 5, 3, 2, 4
   2. 1, 2, 3, 5, 4
   3. 1, 2, 3, 5, 4
   4. 1, 2, 3, 4, 5

Also, compute the time complexity of the algorithm in terms of $\mathcal{O}$ and n with proper explanation.

3. **(Marks: 4)** Write a C program to take an `int` type array *arr* of n (number of your choice) elements as input from the user, store the duplicate (occurring two or more times) elements in *arr* in another array *dup* (repetition removed) and print *dup* if there exists at least one duplicate element in *arr*; otherwise print an appropriate message on the terminal. E.g. if *arr* is 1, 1, 2, 2, 3, *dup* should contain 1, 2 (note that only unique elements are added in *dup*) while if *arr* is 1, 2, 3, 4, 5, *dup* should be empty.

4. **(Marks: 7)** Consider the Grand Finale of a music reality show where six contestants: *Niharika*, *Bidipta*, *Arkadeep*, *Raktim*, *Jyoti*, *Anushka* compete for the winner's trophy. After the performances, three reputed judges provided ranked lists of the contestants in the decreasing order of excellence. An example ranking setup is given in Table 1. Here Judge 1 wants *Niharika* to secure the 1st position, *Arkadeep* the 2nd position and so on. Now, the marks obtained by a contestant (say c), for a judge, is calculated as 1/(rank of c by that judge). For example, for Judge 1, *Niharika* will obtain marks = 1/(rank of *Niharika* by judge 1) = 1/1 = 1. Similarly the marks for *Arkadeep* according to Judge 1 is 1/(rank of *Arkadeep* by judge 1) = 1/2 = 0.5. The total marks obtained by a contestant is the sum of the marks obtained by that contestant over the three judges. For example, the total marks obtained by *Niharika* is 1/1 + 1/2 + 1/1 = 2.5. The contestant that obtains the maximum total marks wins the trophy. Similarly, the 1st runner up (2nd position) and 2nd runner up (3rd position) can be determined. For each position, if there is a tie, the award for that position will be shared.

Judge	**Rank 1**	**Rank 2**	**Rank 3**	**Rank 4**	**Rank 5**	**Rank 6**
Judge 1	*Niharika*	*Arkadeep*	*Bidipta*	*Anushka*	*Raktim*	*Jyoti*
Judge 2	*Arkadeep*	*Niharika*	*Anushka*	*Bidipta*	*Raktim*	*Jyoti*
Judge 3	*Niharika*	*Bidipta*	*Anushka*	*Arkadeep*	*Jyoti*	*Raktim*

Table 1: Example judge rankings for Q4.

Implement the following scheme in C. For n (a number of your choice) contestants take the following as input from the user:

1. The names of the contestants (you may store it in an array *names*)
2. The ranked list of these contestants for Judge 1 (you may store it in array *judge1*)
3. The ranked list of these contestants for Judge 2 (you may store it in array *judge2*)
4. The ranked list of these contestants for Judge 3 (you may store it in array *judge3*)

From the above information compute the total marks of each contestant is an array (say *marks*) of suitable data type and using these arrays print (in this order) the names of the 2nd runner up, 1st runner up and the winner.