University of Zurich

Department of Informatics
Database Technology Group
Prof. Dr. Michael Böhlen

Informatics II
Exercise 1

Week 2
Introduction, Basic Sorting

Exam-style tasks are marked with $*$.

Task 1*. Bubble sort, Insertion sort and Selection sort.

Task 1.1. The following statements are about three sorting algorithms: bubble sort, insertion sort and selection sort.Complete the boxes by checking the correct checkbox or by filling your answer into the empty box.

(a) The bubble sort algorithm can be implemented using two nested `while` loops.

Answer: ☐ True ☐ False

(b) The insertion sort algorithm can be implemented using two nested `for` loops.

Answer: ☐ True ☐ False

(c) Given the same input, all three sorting algorithms always need the same number of comparisons.

Answer: ☐ True ☐ False

(d) All three sorting algorithms only compare two adjacent elements in an array.

Answer: ☐ True ☐ False

Task 2. Play with vowels.
The letters 'A', 'a', 'E', 'e', 'I', 'i', 'O', 'o', 'U' and 'u' are vowels.

Task 2.1 number of vowals. Implement the C function `int count_vowels(char A[])` that returns the number of vowels for a given string `A`.

Task 2.2 *B-Sprache String.* Given a string A[1..n], the B-Sprache string BS of A is generated as followed. We traverse the string A from the first element of A to the last one, one after another. If A[i], $1 \leq i \leq n$, is not a vowel, A[i] is copied to the BS, otherwise, three consecutive letters: A[i], 'b' and A[i] are copied to BS. Implement the C function `void BS(char A[])` that prints the B-Sprache String of A.

In Task 2.1 and Task 2.2, the string A has less than 1000 characters.

Example:
Input string: **Informatik**
B-Sprache string: IbInfobormabatibik

Task 3. Given an array `A[1..n]` with `n` integers, the C function `void even_odd_selection_sort(int A[], int n)` prints the following elements: a sorted array `E` with the even numbers in `A` and a sorted array `O` with the odd numbers in `A`. Implement the C function `void even_odd_selection_sort(int A[], int n)`, using the selection sort as the sorting algorithm. An input/output example is illustrated below (input is typeset in bold):

Values of A separated by spaces (non-number to stop): **2 10 3 22 15 12 end**
Sorted even numbers: 2, 10, 12, 22
Sorted odd numbers: 3, 15

Task 4. Find the Gap
You are employed as a system administrator for the London Underground. You are responsible to manage the file servers of the company. Several processes write files the disks managed by the file server. Every time a file is accessed or modified, the file system will store a timestamp when this operation was performed. The timestamp is an integer number counting seconds from a particular epoch.
You want to find out what the best time is to perform a major overhaul of the system's disks. For this purpose you want to find out at which times the files are used the least. A colleague of yours has already written a script which retrieves the timestamps for all files on the disks attached to the server and a small program to read them as an array of integers in C. Your task is it, to write a program in C which analyses this array and finds in it the biggest time gap, i.e. the point in time when there hasn't been any access or modify operations on the file system for the longest period of time. Note that the script retrieves the timestamps in no particular order and that there might be duplicate timestamps. For example, in the array $A = [9, 1, 4, 9, 5, 3, 9]$ has two gaps and the longest gap is between the timestamps 5 and 9 and has length 4. Write a program in C that finds in an arbitrary array of integers representing timestamps (i) the oldest timestamp, (ii) the most recent timestamp and (iii) the length of the longest gap within the timestamps. You may use a hard-coded array of timestamp values to test your program.