Data Structures and Algorithms

—

Lab 1. Practice algorithm analysis

Warm-Up Exercise: Sorting

1. Write down an unsorted sequence of 5 integers
2. Rotate the papers
3. Apply any sorting algorithm you know, write down intermediate state at every step
4. Rotate the papers
5. Try to guess the algorithm and write down its name
6. Rotate papers back and mark the guess
7. Rotate again
8. Did you guess correctly?

Analysis of Bubble Sort

Exercise 1.1. Compute worst-case time complexity of bubble sort.

```
bubbleSort(A, n):
1 |    swapped := true
2 |    while swapped
3 |         swapped := false
4 |         for i := 1 to n-1
5 |             /* if this pair is out of order */
6 |             if A[i-1] > A[i]
7 |                 /* swap and remember something changed */
8 |                 exchange A[i-1] with A[i]
9 |                 swapped := true
```

Exercise 1.2. Compute best-case time complexity of bubble sort.

Theoretical Problem Set

ASSIGNMENT

Week 1. Problem set

Opens: Monday, 23 January 2023, 12:30 PM
Due: Wednesday, 25 January 2023, 11:59 PM

Submission format requirements:

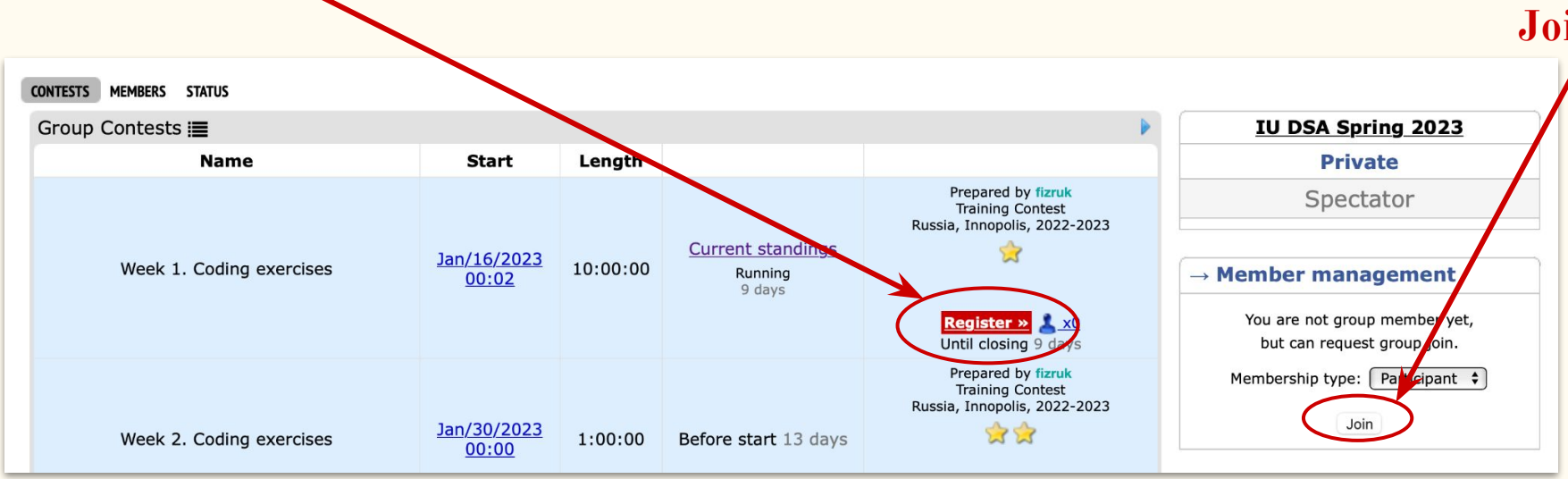
1. Typeset your solutions in Word or LaTeX.
2. Include the original problem statements.
3. Clearly indicate final answer in each problem.
4. Follow this file naming scheme: `<Firstname><Lastname>_problem_set_<N>.<ext>`
5. Submit both the PDF and the corresponding source file (e.g. `NikolaiKudasov_problem_set_1.pdf` and `NikolaiKudasov_problem_set_1.tex`)
6. Do not submit archives!

Available from **23 January 2023, 12:30 PM**

Coding Exercises

CODEFORCES
Sponsored by TON

1. https://codeforces.com/group/v3tYbkCHj3/contests
2. **Join** the group «**IU DSA Spring 2023**»
3. **Register** for the contest «**IU DSA Spring 2023 — Week 1. Coding exercises**»



Selection Sort: Exercises

Consider sorting N elements stored in an array A by finding the smallest element of A and exchanging it with the element in A[1]. Then finding the second smallest element and exchanging it with the element in A[2]. Continue similarly for the first N – 1 elements of A.
This algorithm is known as **selection sort**.

Exercise 1.3. Write pseudocode for selection sort.

Exercise 1.4. Specify the loop invariant for the main loop in the algorithm.

Exercise 1.5. Prove that it is enough to run the main loop for only N – 1 iterations.

Exercise 1.6. Find best- and worst-case time complexity of selection sort in Θ-notation.

Exercise 1.7. Find best- and worst-case time complexity of selection sort given that writing/modifying array takes significantly longer than reading or comparing elements.

Nearly Sorted Sequence

Exercise 1.8. Implement insertion or bubble sort, and selection sort.
Make a prediction of how well these algorithms might perform.
Submit both solutions to CodeForces.
Compare your prediction with results on CodeForces.
Can you explain what happens?

Examples

input	Copy
10 1 3 2 5 4 6 7 9 8 10	
output	**Copy**
1 2 3 4 5 6 7 8 9 10	

Recap

- What is the best/worst-case time complexity of insertion/selection/bubble sort?
- In what situations is insertion/selection/bubble sort good?
- Can you implement insertion/selection/bubble sort?

See you next week!