FACULTY OF COMPUTER SCIENCE IAŞI

@ FII

UNIVERSITATEA „ALEXANDRU IOAN CUZA“ din IAŞI

Lab 1

- Back to main
- The following exercises are meant to be solved during the lab. Ask your instructor for help when needed.

Exercises

1. [1p] Find, if any, problems in the following Java implementation of the quicksort algorithm:

```
public class QuickSort {

    public static void quicksort(int[] arr, int low, int high) {
        if (low < high) {
            int pivotIndex = partition(arr, low, high);
            quicksort(arr, low, pivotIndex - 1);
            quicksort(arr, pivotIndex, high);
        }
    }

    private static int partition(int[] arr, int low, int high) {
        int pivot = arr[(low + high) / 2];
        while (low <= high) {
            while (arr[low] < pivot) {
                low++;
            }
            while (arr[high] > pivot) {
                high--;
            }
            if (low <= high) {
                int temp = arr[low];
                arr[low] = arr[high];
                arr[high] = temp;
                low++;
                high--;
            }
        }
        return low;
    }

    public static void main(String[] args) {
        int[] numbers = {3, 7, 8, 5, 2, 1, 9, 5, 4};
        quicksort(numbers, 0, numbers.length - 1);
    }
}
```

2. [1p] Find, if any, problems in the following Java implementation of the binary search algorithm:

```
public class BinarySearch {
    public static int binarySearch(int[] arr, int target) {
        int low = 0, high = arr.length - 1;
        while (low < high) {
            int mid = (low + high) >>> 1;
            if (arr[mid] == target) {
                return mid;
            } else if (arr[mid] < target) {
                low = mid;
            } else {
                high = mid - 1;
            }
        }
        return (arr[low] == target) ? low : -1;
    }

    public static void main(String[] args) {
        int[] nums = {1, 3, 5, 7, 9, 11, 13};
        // call the function
    }
}
```

3. [1p] Find, if any, problems in the following Python implementation of matrix multiplication:

```
def matrix_multiply(A, B):

    m = len(A)
    n = len(A[0])
    p = len(B[0])

    if n != len(B):
        raise ValueError("Matrices dimensions are incompatible")

    C = [[0] * p] * m

    for i in range(m):
        for j in range(p):
            sum_of_products = 0
            for k in range(n):
                sum_of_products += A[k][j] * B[i][k]

            C[i][j] = sum_of_products

    return C
```

4. [1p] Find, if any, problems in the following Python implementation of finding the longest increasing sequence:

```
def longest_increasing_subsequence(arr):
    if not arr:
        return 0

    n = len(arr)
    dp = [1] * n

    for i in range(1, n):
        for j in range(i):
            if arr[j] <= arr[i]:
                dp[i] = max(dp[i], dp[j] + 1)

    return max(dp)

if __name__ == "__main__":
    arr = [3, 10, 2, 1, 20, 4, 10]
    print("Length of LIS:", longest_increasing_subsequence(arr))
```