Gedächtnisprotokoll_algodat_26_7_2024

***aufgaben 1 bis 3 sind programmierungsaufgbe. of course the following content may contain errors, especialluy in the code sections. Sorry for the mistakes in advance

---

Aufgabe 1

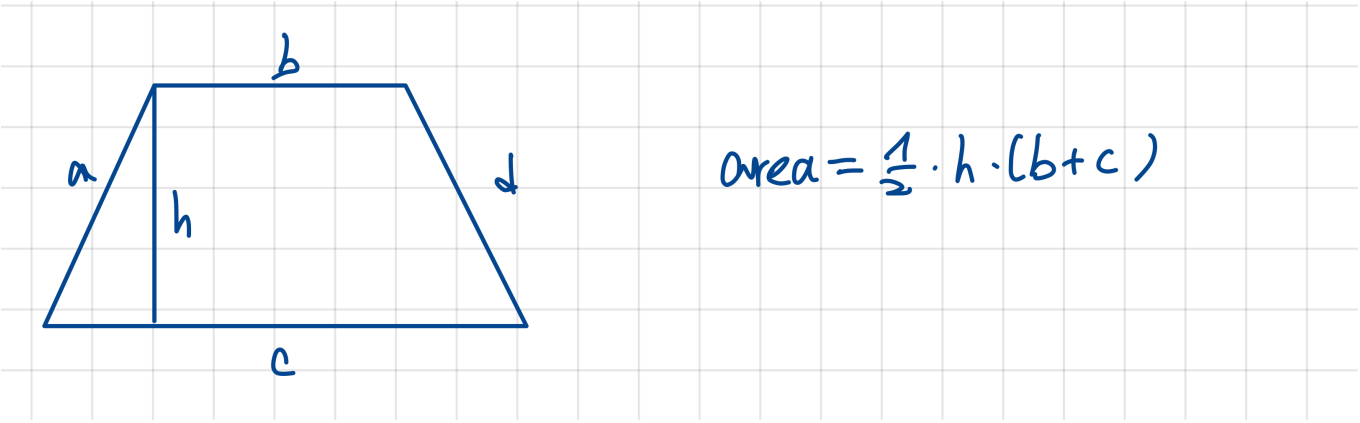
1. korriegiert Constructor:

```
class Test{
        int a;
        int b;
        class Test(int a. int b){
                a = this.a;
                b = this.b;
        }
}
```

2. korriegiert swap-method

```
public void swap(){
        this.a = this.b;
        this.b = thia.a;
}
```

Aufgabe 2



- schreib die laufzeit eines function

- correct following class:

```
import java.util.ArrayList;
import java.util.List;

public class Quadrilateral {

    List<Edge> edges;


    public Quadrilateral(double a, double b, double c, double d) {
        this.edges = new ArrayList<>();
        this.edges.add(a);
        this.edges.add(b);
        this.edges.add(c);
        this.edges.add(d);
    }


    public double getPerimeter() {
        double perimeter = 0;
        for (double edge : edges) {
                for(int i; i <edge.length;i++){

                    perimeter += edge.length;
                }

        }
        return perimeter;
    }
//correct version:
/*
        public double getPerimeter() {
        double perimeter = 0;
        for (double edge : edges) {
            perimeter += edge.length;
        }
        return perimeter;
    }
*/


    public double getArea(double height, double base1, double base2) {
        return 0.5 * height * (base1 + base2);
    }
```

```
}
```

Aufgabe 3

assume following 2 class is correct

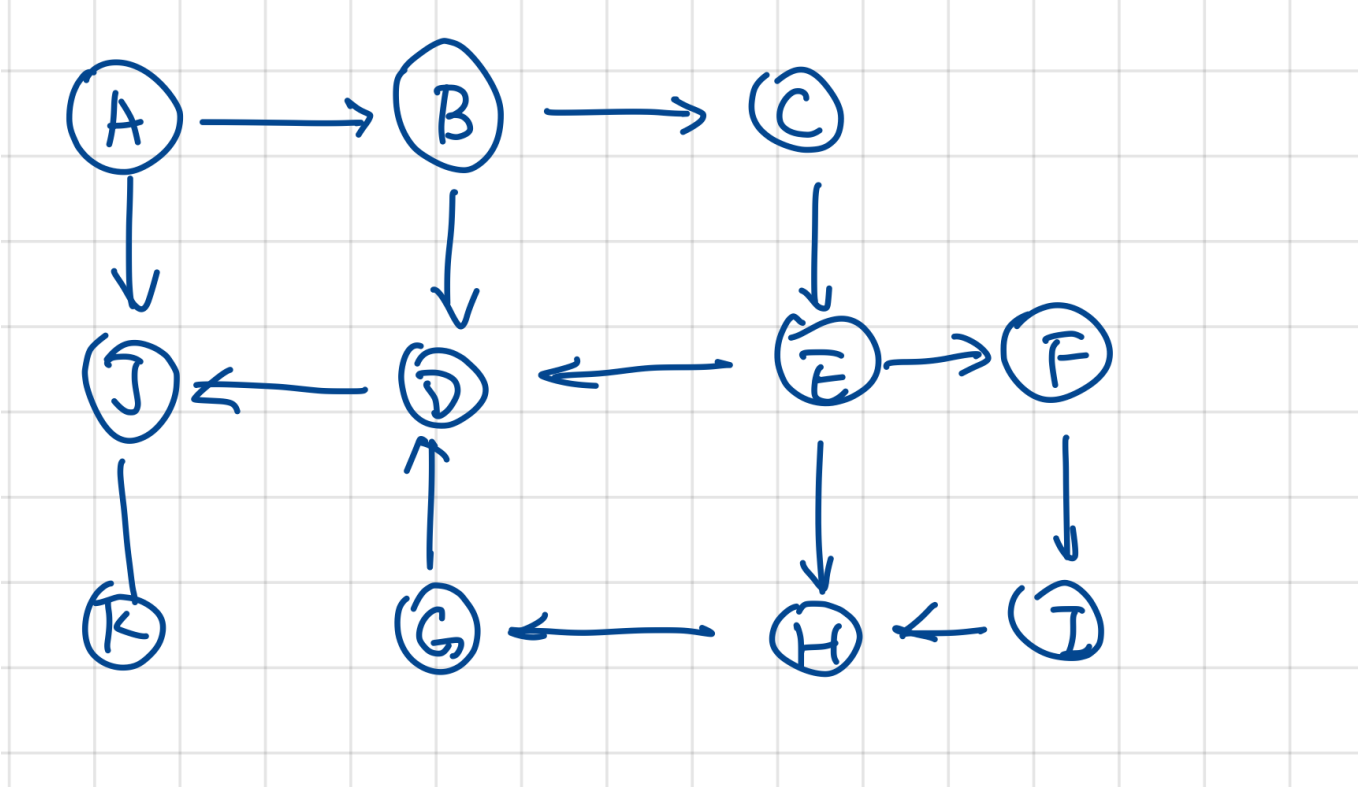
```
public class Interval {
    private int start;
    private int end;

    public Interval(int start, int end) {

        this.start = start;
        this.end = end;
    }

    public int getStart() {
        return start;
    }

    public int getEnd() {
        return end;
    }

}
```

```
import java.util.Comparator;

public class IntervalComparator implements Comparator<Interval> {

    @Override
    public int compare(Interval o1, Interval o2) {
        if (o1.getStart() != o2.getStart()) {
            return Integer.compare(o1.getStart(), o2.getStart());
        } else {
            return Integer.compare(o1.getEnd(), o2.getEnd());
        }
    }
}
```

- please correct the following class
- what is the output of main

```
import java.util.*;

public class IntervalGreedy {

    public static Queue<Interval>
findEarliestEndIntervals(Collection<Interval> intervals) {
        // Create a priority queue that orders intervals by their end time
        PriorityQueue<Interval> pq = new PriorityQueue<>(new
Comparator<Interval>());


        pq.addAll(intervals);

        Queue<Interval> result = new LinkedList<>();

        while (!pq.isEmpty()) {
            Interval current = pq.peek(); // soll poll() sein
            if (current.getStart() >= result.peek().getEnd()) {
                result.add(current);
            }
        }

        return result;
    }

    public static void main(String[] args) {
        List<Interval> intervals = new ArrayList<>();
        intervals.add(new Interval(1, 3));
        intervals.add(new Interval(2, 4));
        intervals.add(new Interval(3, 5));
        intervals.add(new Interval(1, 2));
        intervals.add(new Interval(5, 6));

        Queue<Interval> result = findEarliestEndIntervals(intervals);

        System.out.println("Selected intervals:");
        for (Interval interval : result) {
            System.out.println(interval);
        }
    }
}
```

aufgabe 4 DFS



- preorder
- postorder
- remove which edge so that the first element in postorder is not K.
- can it find the shortest path? Provide a reason.

minimaler spannbaum

wie altklausur 2023 1.termin aufgabe 5

bellmann ford algo

wie altklausur 2020 1.termin aufgabe 8
mit queue

Hashing

$$hash(h) = k \; mod \; 10$$

- key 17, 8 habe been added to the hashtable

- linear sondierung mit inresment 1
- given key: 12, 22, 13, 14

index	0	1	2	3	4	5	6	7	8	9	
key								17	8		

- Question: Please insert such keys where the order of given keys results in the maximum number of collisions.

index	0	1	2	3	4	5	6	7	8	9
key								17	8	

- Question: Please insert such keys where the order of given keys results in the minimum number of collisions.

index	0	1	2	3	4	5	6	7	8	9
key								17	8	

Graph

1. **Distanz (dist)**:
   - Die Distanz zwischen zwei Knoten (u) und (v) in einem Graphen (G) ist die Länge des kürzesten Pfades, der diese beiden Knoten verbindet. Dies wird oft als $\mathrm{dist}(u, v)$ bezeichnet.
2. **Exzentrizität (Eccentricity)**:
   - Die Exzentrizität eines Knotens (u) in einem Graphen (G) ist die maximale Distanz von (u) zu irgendeinem anderen Knoten im Graphen. Mathematisch wird dies als (e(u)) ausgedrückt:

     $$[e(u) = \max_{v \in V} \mathrm{dist}(u, v)]$$

   - Die Exzentrizität gibt an, wie weit ein Knoten von den am weitesten entfernten Knoten im Graphen entfernt ist.
3. **Radius (Radius)**:
   - Der Radius eines Graphen (G) ist das Minimum der Exzentrizitäten aller Knoten im Graphen. Mathematisch wird dies als (r(G)) ausgedrückt:

     $$[r(G) = \min_{u \in V} e(u)]$$

   - Der Radius gibt die kleinste maximale Distanz eines Knotens zu allen anderen Knoten im Graphen an und repräsentiert somit den „zentralsten“ Punkt des Graphen

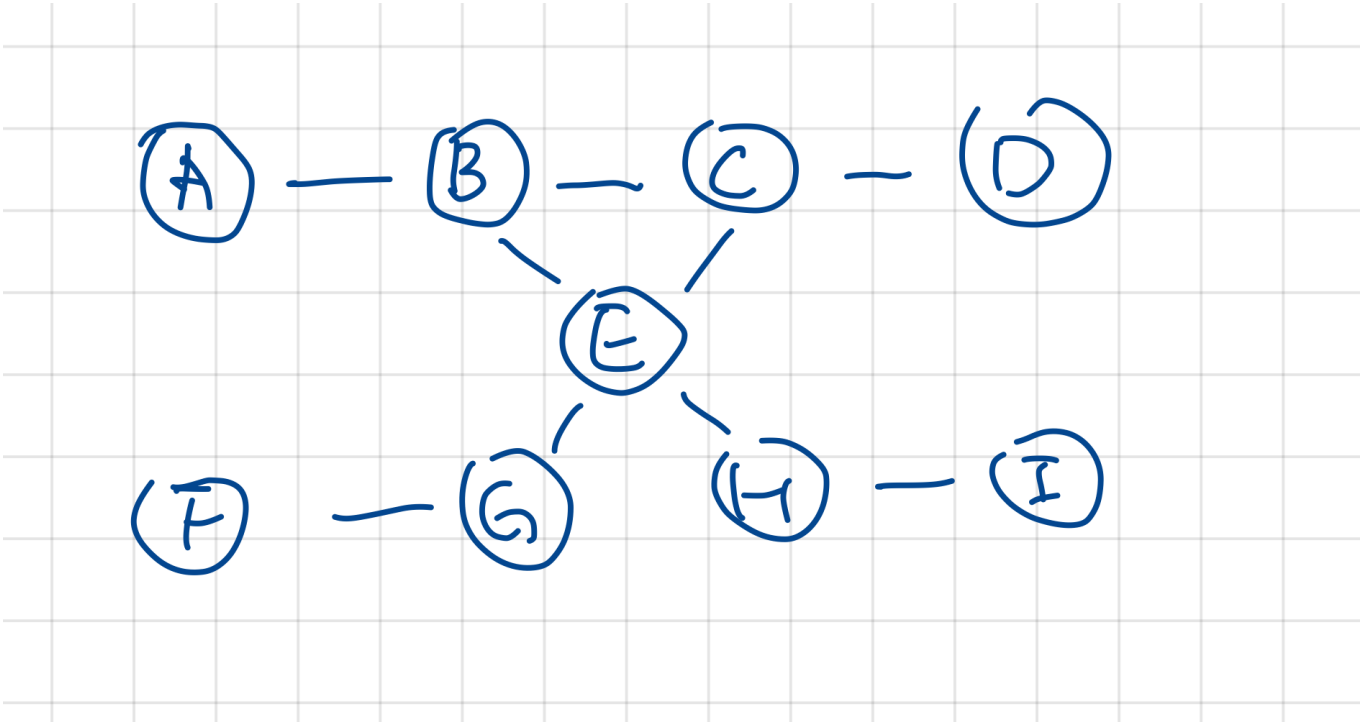
in Bezug auf die maximalen Distanzen.

frage 1

given a graph, say what is e(A) and e(B)
i can't remember the graph

frage 2

mark the knote with minimal Exzentrizität



frage 3

schreib eine Algo mit Laufzeit O(V (V+E)), sodass die radius von G gesucht wird

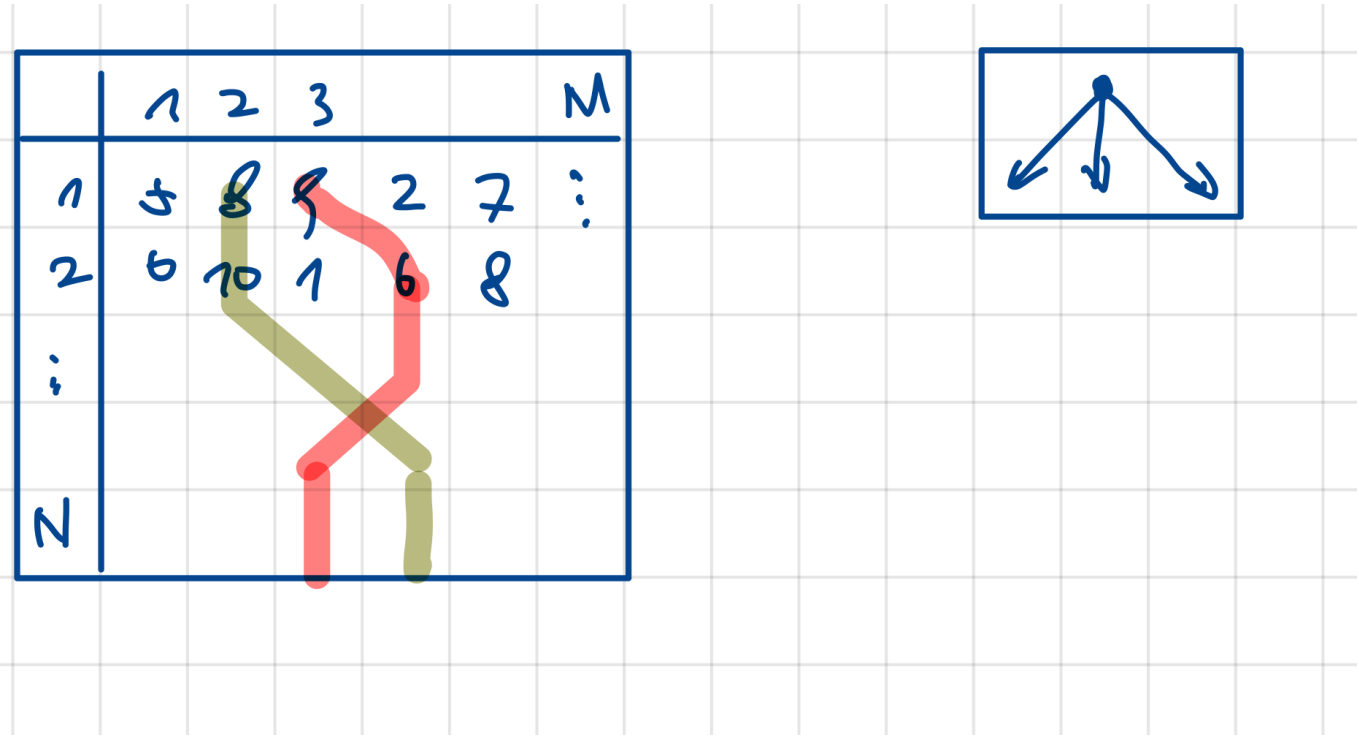
dynamische programmierung

The rules of the game are as follows:

1. **Game Board**: A (M \times N) grid where each cell contains a number.
2. **Starting Point**: Begin from any cell in the first row (i.e., (n = 1)).
3. **Objective**: Traverse from the first row to the (N)-th row, selecting cells such that the sum of the numbers along the path is maximized.
4. **Movement Rules**:
    - From a cell ((m, n)) in the current row, you can move to one of the three cells in the next row:

        - ((m-1, n+1)) (the cell to the bottom-left)
        - ((m, n+1)) (the cell directly below)
        - ((m+1, n+1)) (the cell to the bottom-right)
    - You can only move to adjacent cells in the row directly below.
5. **Path Selection**: Following the movement rules, choose a path from the first row to the (N)-th row such that the sum of the numbers along the path is maximized.

Goal: Find the path with the maximum possible sum of numbers and output this maximum sum.



- frage: schreibe OPT(n,m)
- frage: cant remember.

Das war was ich noch im Kopf hatte. vielleicht können die anderen ergänzen