Geoinformatics Worksheet 1 120min

1. Fundamants:

a) Geospacial data is digital geodata which has references to a geometric sytem (geo refrenced data)

b) - Road connection network.
- Country boarders.
- Elevation Data.
- Position of restaurantes

c) - Input and update
- Managment and modelling
- Analysis and simulation
- Output and presentation

d) Vector data:
→ Disribe an object based on Coordinates Points. Points cam be combined to polylines and polygons
- Positon of sky lifts
- Forest area as a polygon

Raster data:
→ Decompose space into equal sized cells.
- Elevation data
- humidity data

e) - Land information systems
→ For land use administraiton
- Spatial information system
→ Planning and presentation
- Network information system
→ connection managemot between objects

W1/1

2. Modelling

a) -Data can be modeled as discried Objects. These have fix boundarys and can be seen as instances of an object.
- Data can be modelt as a continiros fild. Here every measurmant point distribes the data only in this exact point. Continius data can very in a space without boundarys.

b) Land cover

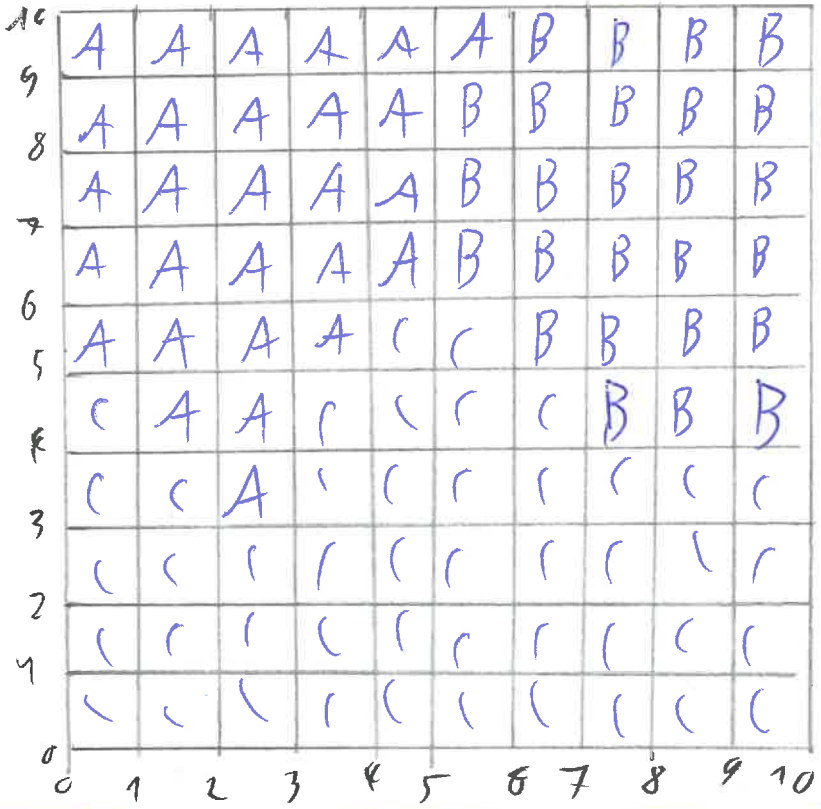
$B = [(6,10),(10,10),(10,4),(7,4),(5,6),(6,10)]$

Vector:

$A = [(0,10),(6,10),(5,6),(3,3),(0,5),(0,10)]$

$C = [(0,5),(3,3),(5,6),(7,4),(10,4),(0,10),(0,0),(0,5)]$

Raster:



Cities:

Vector:

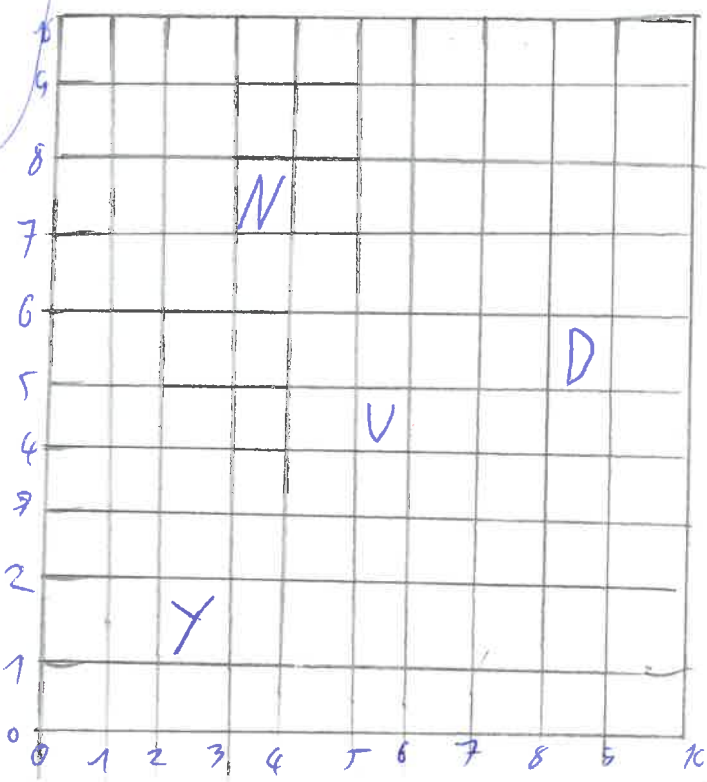
Nice: $[(3.5, 7.5)]$

Delft: $[(8.5, 5.5)]$

Ulm: $[(5, 4)]$

York: $[(2.5, 1.5)]$

Raster:



W1/2

c) Vector:

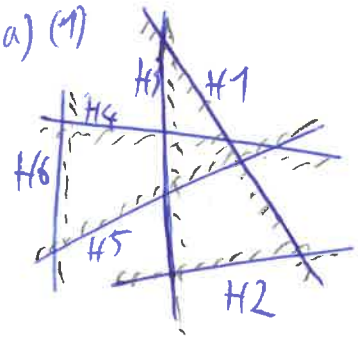
→ city points in an array
→ cycle through the array
→ check for every point if its inside a polygon (in land cover)
→ save all the land cover of the polygons that contained that point as attributes to the point.

Raster:

→ Have an array of all the citys and their raster name/ID
→ cycle through the city raster until there is a cell that contains a wanted city.
→ Check in the land cover Raster Table the cell with the same coordinates.
→ Save all the land cover data form this cell to the city array or Raster cell of cities

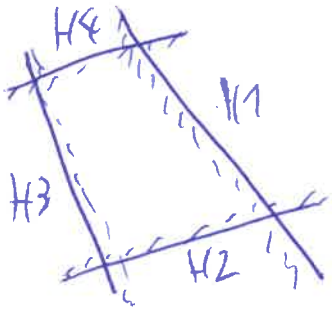
3). Half space modelling

a) (1)



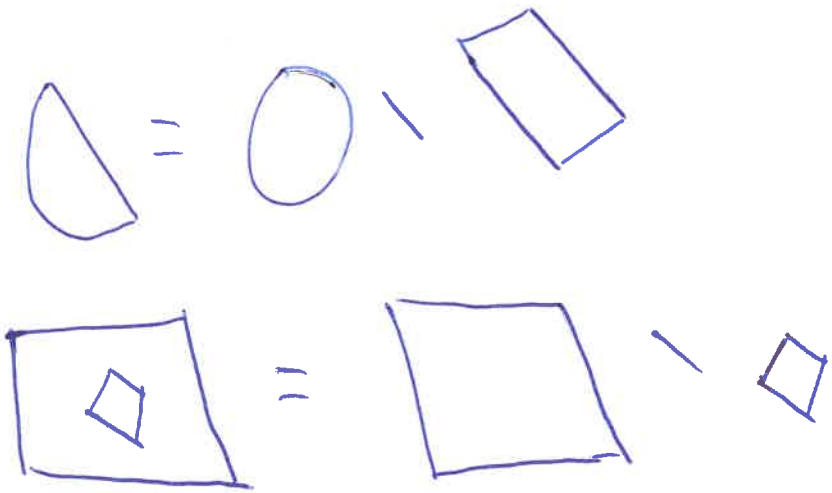
$(1) = H1 \cap H2 \cap H3 \cup H4 \cap H5 \cap H6$

(2)



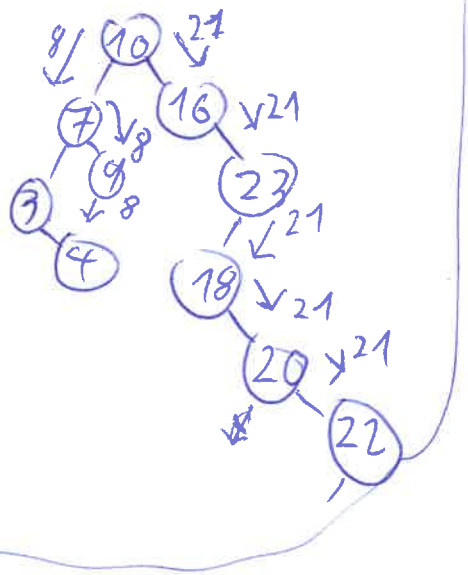
$(2) = H1 \cap H2 \cap H3 \cap H4$

b)



4.) Search Trees & Heap

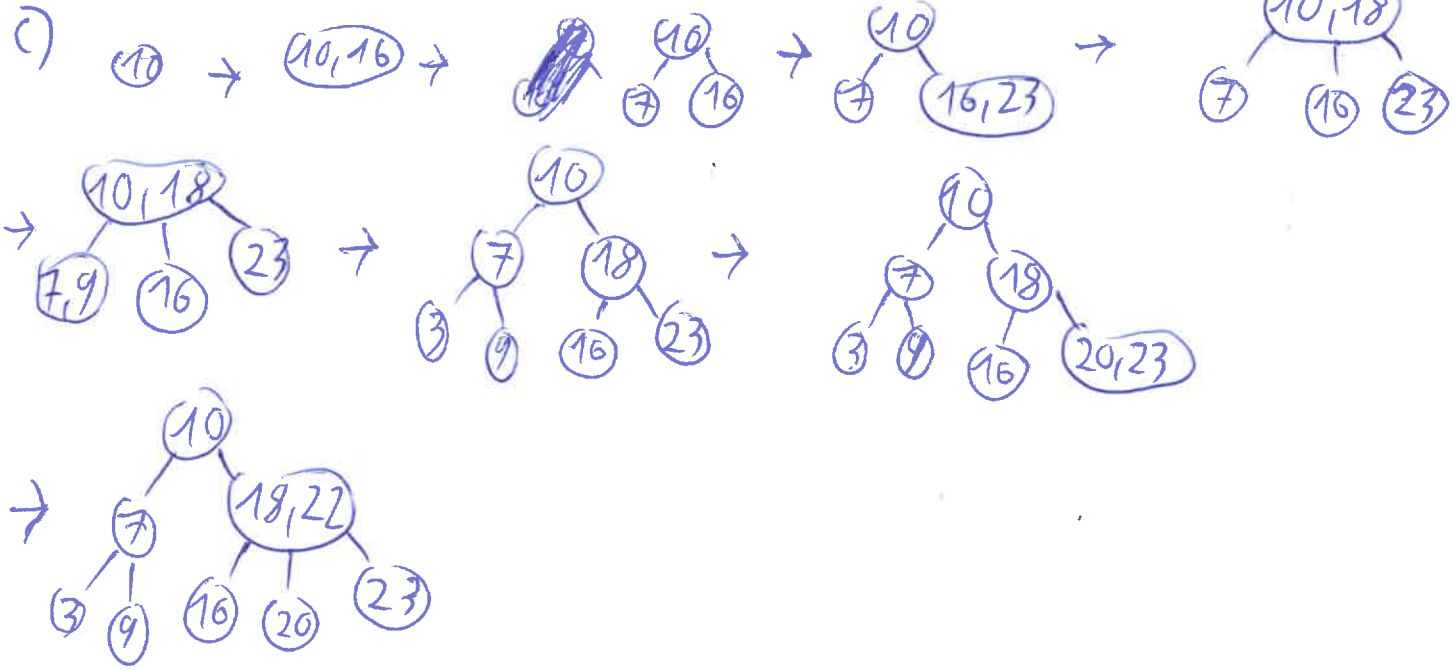
a)



b) Starting by the root check if the value is smaller or greater than the on in the Tree.
Smaller → Test the same for the left children
right → Test the same for the right children.
When you reach a leaf check if its the wanted value or not.

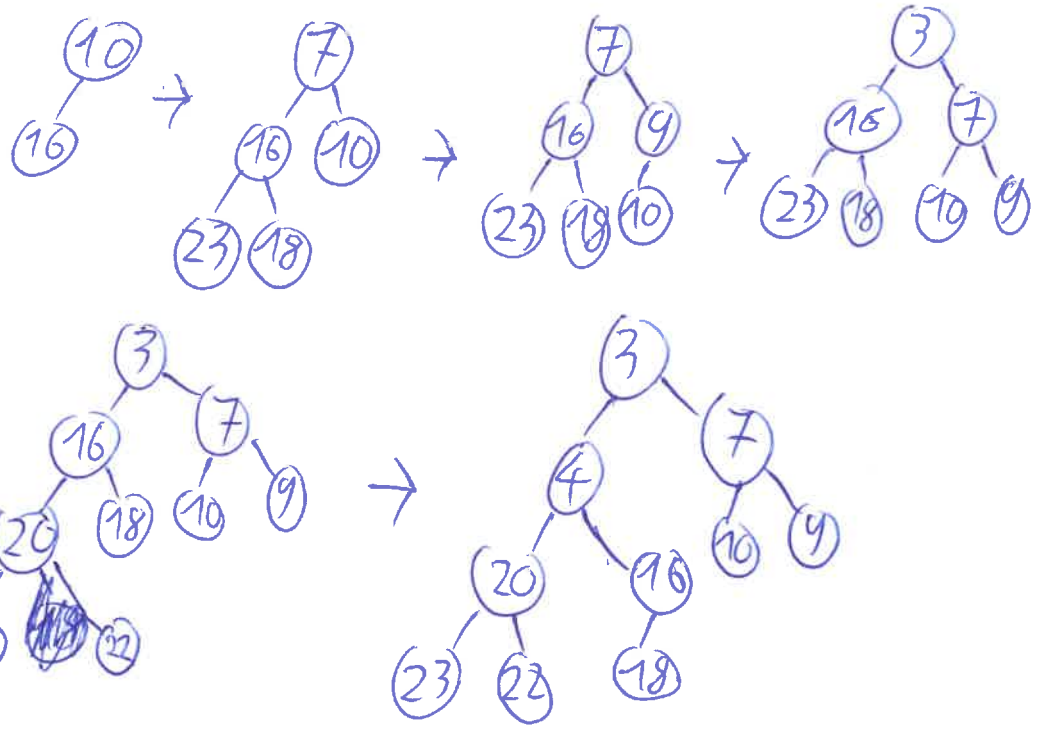
~~For 8:~~
~~will find 9 as a leaf which is not correct~~
~~For 21:~~
- For 8:
- Will find no value left from 9
  For 21:
- Will find no value left from 22

c)

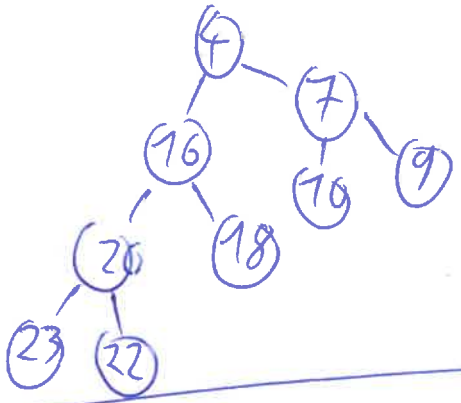


W1
/4

d)



e) rome root



5. Spatial Overlays

a) → Define for every vertex of each polygon if its inside, outside or on the boundary of the other Polygon

=> A has one inside point

=> B has two inside points

→ Add intersection Points to both polygons. They get added as boundarys to the vertex arrys

=> both get two new intersetion points

→ difine all intersecting edges as inside, outside or boundary edges

- A get 5 edges
- B gets also 5 edges.

→ combine both dataset step by step until they build a cycle. Then start with a new cycle.
=> Two edges in the same direction => remove one
=> Two edges in opposite direction => remove both

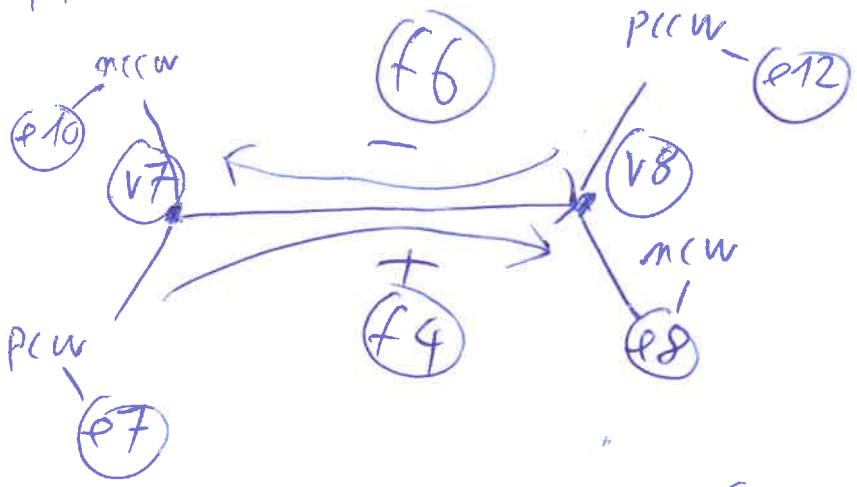
b) 1. Create minimal bounding boxes for all rivers and the bridge.
2. Check which river bounding box intersec the on of the bridge. Continue only with these rivers.
3. Check for every edge of the river (cronological two point pairs), if they cross the poly line of the river.

=> This is testest by calculating the vector product of these 4 points form both edges.
if one vector product negative and the othe positive the bridge crosses the river at this edge.

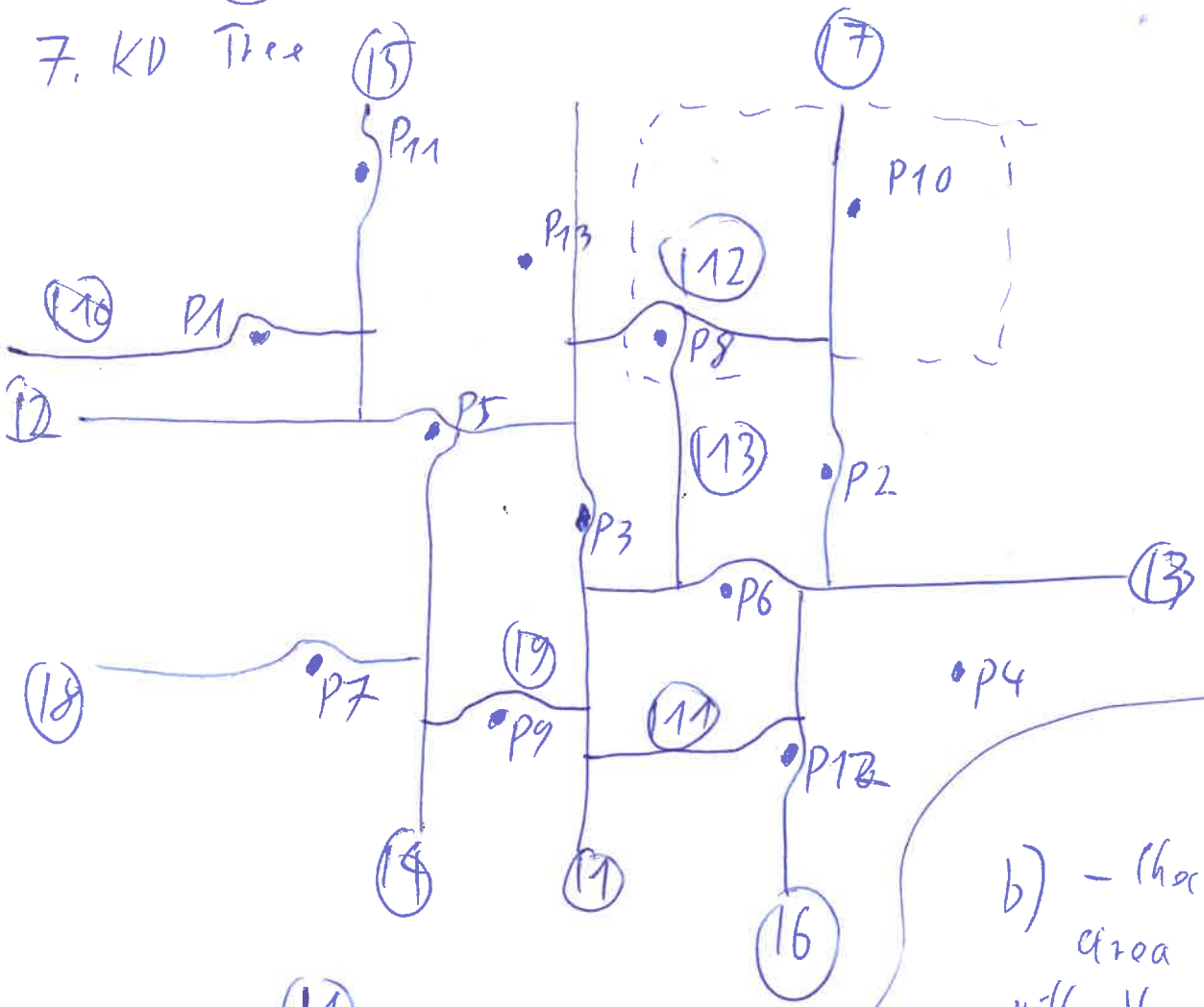
6. Topological Data structure for boundary Models.

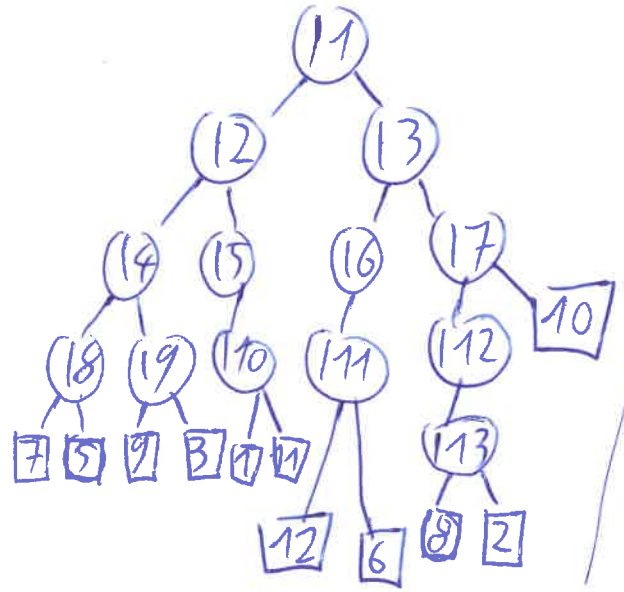
a) for f3
→ look in the face table for the starting edge.
→ look in the edge table if the face is listed as fcw or fccw.
→ When fcw the next edge is found in ncw.
else → When fccw the next edge is found in nccw.
- Store all start and end vertices along the way and repeat this for all the next edges until a cycle is formed.

b) e11

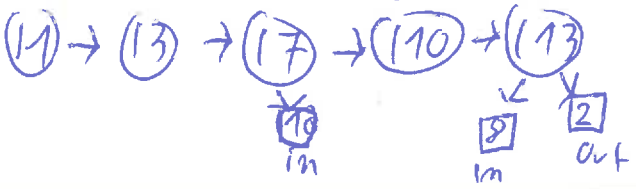


7. KD Tree





b) - Check for every area if it intersects with the query window.
If yes traverse the graph further and test if for the two children.
When only points are left, Test the points individual.
How it could look



W1
7

c) - k nearest neighbour query
- Ball query

8. Shortest path

a)

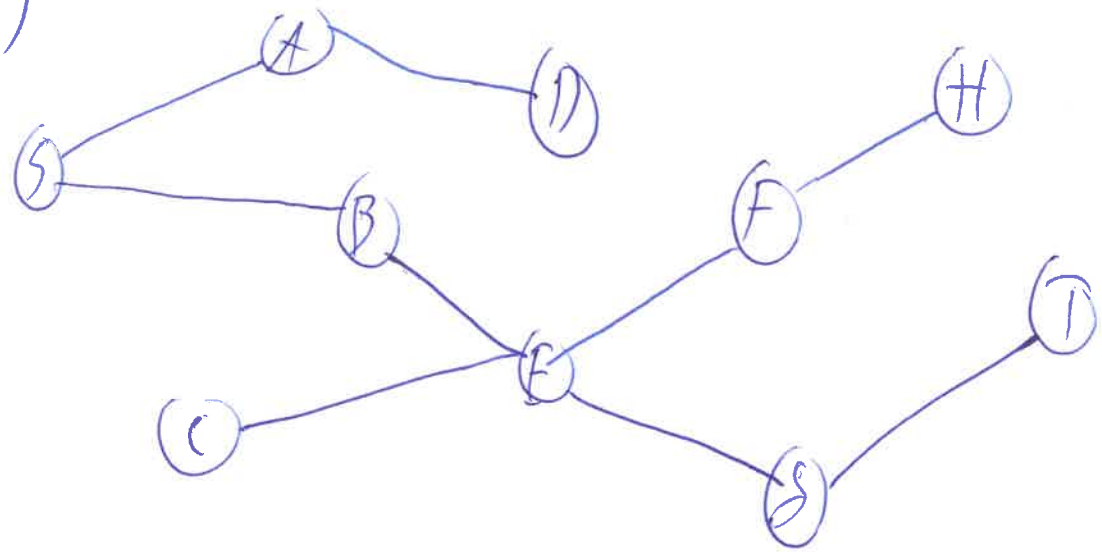
Values	S	A	B	C	D	E	F	G	H	T	Markt
Init	$^{S}0$	∞	∞	∞	∞	∞	∞	∞	∞	∞	S
S=0	~~$^{S}0$~~	$^{S}2$	$^{S}1$	$^{S}5$	∞	∞	∞	∞	∞	∞	A, B, C
B=1	/	$^{S}2$	/	$^{S}5$	∞	$^{B}3$	∞	∞	∞	∞	A, C, E
A=2	/	/	/	$^{S}5$	$^{A}7$	$^{B}3$	∞	∞	∞	∞	C, E, D
E=3	/	/	/	$^{E}4$	$^{A}7$	/	$^{E}4$	$^{E}7$	∞	∞	C, D, F, G
C=4	/	/	/	/	$^{A}7$	/	$^{E}4$	$^{E}7$	∞	∞	D, F, G,
F=4	/	/	/	/	$^{A}7$	/	/	$^{E}7$	∞	∞	D, G, H
~~D=7~~	~~/~~	~~/~~	~~/~~	~~/~~	~~/~~	~~/~~	~~/~~		~~∞~~	∞	~~G, H~~
F=4	/	/	/	/	$^{\nearrow}7$	/	/	$^{E}7$	$^{F}6$	∞	D, G, H
H=6	/	/	/	/	$^{A}7$	/	/	$^{E}7$	/	$^{H}11$	D, G, T
D=7	/	/	/	/	/	/	/	$^{E}7$	/	$^{H}11$	G, T
G=7	/	/	/	/	/	/	/	/	/	$^{G}8$	T
T=8 ✓											

b) The A* algorithm also use predefined weights like distances. The are constident for choosing the next edge.

c) The predefined weights have to be =< the real weights of the edges.

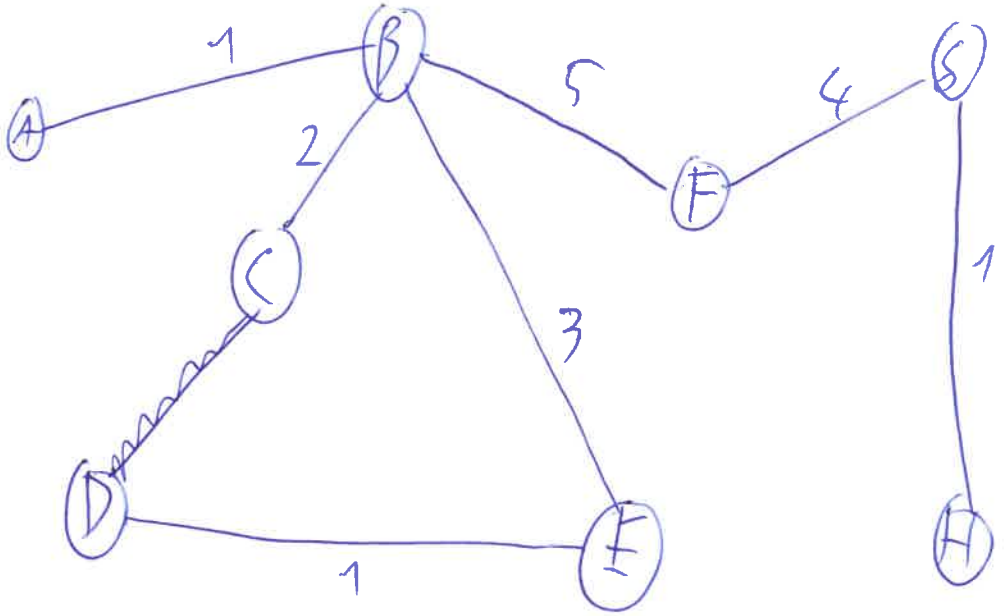
8

Aufg 8)



a) Minimum spanning tree

~~q: [(A,B,1), (A,C,3)...~~



A,B 1
D,H 1
D,E,1
C,B 2
A,C 3 *
F,G 4
C,D 4*

B,E 3
B,F,5
A,D 5*
E,H 7*
E,F 8*

W = 17

- Edges get selected by priority queue

W1
a

- First try
- The time constrain is ~~[illegible]~~ crazy
122 minutes needed / 2 minutes to long

10