LULEÅ TEKNISKA UNIVERSITET

Namn: Samuel Gabrielsson

Personnummer: 7912090250

Sidnr: 1

Uppgift nr: 1 ab

Poäng: 10

Lärarens anteckning:

Sort the following numbers step by step

a) Sort 24, 13, 17, 34, 15, 27, 15, 29, 22, 26 using mergesort

24 13 17 34 15 27 15 29 22 26

Div. 24 13 17 34 15 | 27 15 29 22 26

Div 24 13 | 17 34 15 | 27 15 | 29 22 26

Div 24 | 13 | 17 | 34 15 | 27 | 15 | 29 | 22 26

Div 24 | 13 | 17 | 34 | 15 | 27 | 15 | 29 | 22 | 26

Merge 24 | 13 | 17 | 15 34 | 27 | 15 | 29 | 22 26

Merge 13 24 | 15 17 34 | 15 27 | 22 26 29

Merge 13 15 17 24 34 | 15 22 26 27 29

Merge 13 15 15 17 22 24 26 27 29 34

b) Sort 24, 13, 17, 34, 15 using insertionsort

sorted

24 | 13 17 34 15

13 24 | 17 34 15

13 17 24 | 34 15

13 17 24 34 | 15

13 15 17 24 34 |

LULEÅ TEKNISKA UNIVERSITET

Namn: Samuel Gabrielsson

Personnummer: 791209 0250

Sidnr: 2

Uppgift nr: 2ab

Poäng: 10

Lärarens anteckning:

By repeatedly inserting the input elements 22, 13, 17, 32, 19, 9 in turn:

a) Construct a hash table of size 10 using linear probing

$h(x,i) = (x \bmod 10 + i) \bmod 10 \quad i = 0, 1, \ldots, 9$

0	1	2	3	4	5	6	7	8	9
		22							
		22	13						
		22	13				17		
		22	13	32			17		
		22	13	32			17		19
9		22	13	32			17		19

$h(22,0) = (22 \bmod 10) \bmod 10 = 2 \bmod 10 = 2$

$h(13,0) = (13 \bmod 10) \bmod 10 = 3 \bmod 10 = 3$

$h(17,0) = (17 \bmod 10) \bmod 10 = 7 \bmod 10 = 7$

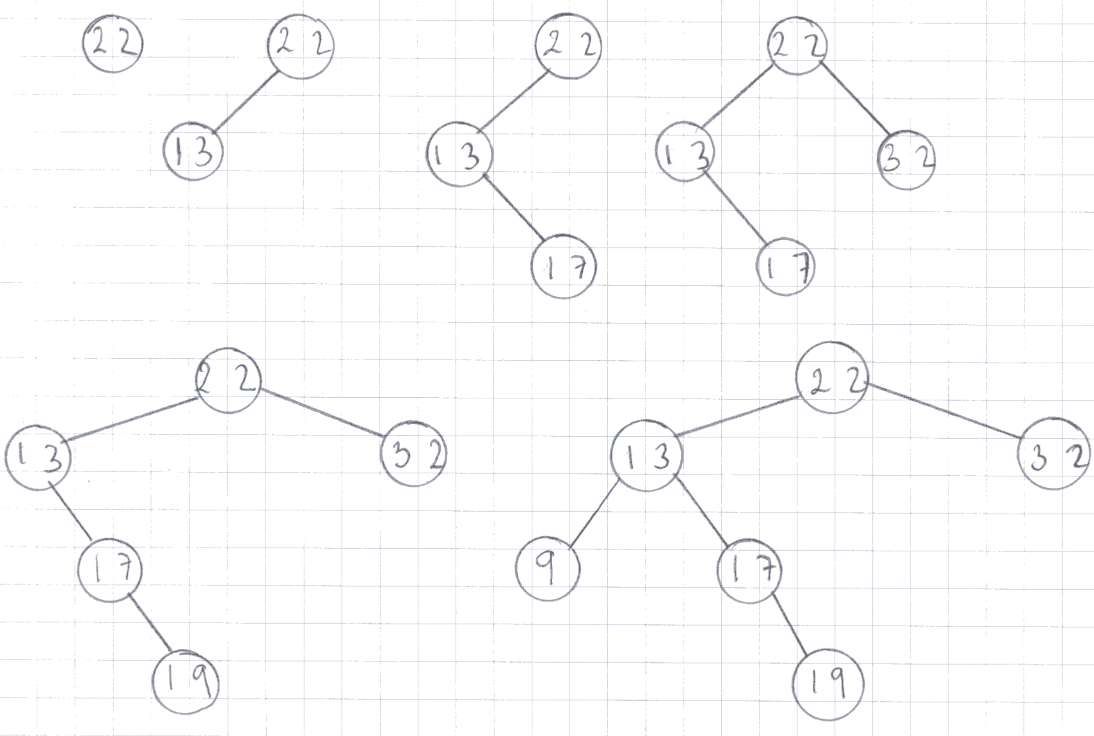
draw the hash table.

$h(32,0) = (32 \bmod 10) \bmod 10 = 2 \bmod 10 = 2$ collision
$h(32,1) = (32 \bmod 10 + 1) \bmod 10 = 3 \bmod 10 = 3$ collision
$h(32,2) = (32 \bmod 10 + 2) \bmod 10 = 4 \bmod 10 = 4$

$h(19,0) = (19 \bmod 10) \bmod 10 = 9 \bmod 10 = 9$

$h(9,0) = (9 \bmod 10) \bmod 10 = 9 \bmod 10 = 9$ collision
$h(9,1) = (9 \bmod 10 + 1) \bmod 10 = 10 \bmod 10 = 0$

b) Construct a binary search tree.



Namn:	Personnummer:	Sidnr:
Samuel Gabrielsson	7912090250	3

Uppgift nr: 3ab

Poäng: 5

Lärarens anteckning: Max 10p

Prove (by using the definitions) or disprove (by giving counterex.) the following assertions

a) Let $f(n)$ and $g(n)$ be positive integer functions. If $f(n) = O(g(n))$, then $g(n) = \Omega(f(n))$

$f(n) \in O(g(n)) \Rightarrow \exists c, n_0 > 0$ such that $f(n) \leq c \cdot g(n) \quad \forall n \geq n_0$

We want to show that $g(n) \geq c \cdot f(n) \quad \forall n \geq n_0$

$f(n) \leq c \cdot g(n) \Rightarrow \underbrace{\frac{1}{c}}_{=c} f(n) \leq g(n) \quad \forall n \geq n_0$

$\because \; g(n) \geq c \cdot f(n) \Rightarrow g(n) \in \Omega(f(n)) \quad \square$. True!

b) Let the worst-case time complexity of two algorithms A and B be $\Theta(t(n))$ and $O(t(n))$, respectively. Then algorithm A is asymptotically faster than algorithm B

$\because \; A \in \Theta(t(n))$ and $B \in O(t(n))$, then A "$>$" B ?

$A \in \Theta(t(n)) \Rightarrow \exists c_1, c_2, n_0 > 0$ such that $c_1 t(n) \leq A(n) \leq c_2 t(n) \quad \forall n \geq n_0$
$B \in O(t(n)) \Rightarrow \exists c, n_0 > 0$ such that $B(n) \leq c \cdot t(n) \quad \forall n \geq n_0$

Let $A(n) = n$ and $B(n) = n^2$, then $(t(n) = n^2)$ −5p

n !"$>$" n^2, hence assertion is False!

wrong example.

Universitetstryckeriet, Luleå

LULEÅ TEKNISKA UNIVERSITET

Namn: Samuel Gabrielsson

Personnummer: 791209 0250

Sidnr: 4

Uppgift nr: 5abc

Poäng:

Lärarens anteckning:

p. a = 2p
b = 4p
c = 5p
d = 4p

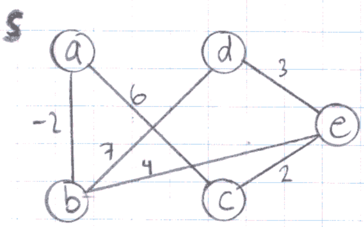
Given a graph with the adjacency matrix:

$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & -2 & 6 & 0 & 0 \\ -2 & 0 & 0 & 7 & 4 \\ 6 & 0 & 0 & 0 & 2 \\ 0 & 7 & 0 & 0 & 3 \\ 0 & 4 & 2 & 3 & 0 \end{bmatrix} \end{array}$$

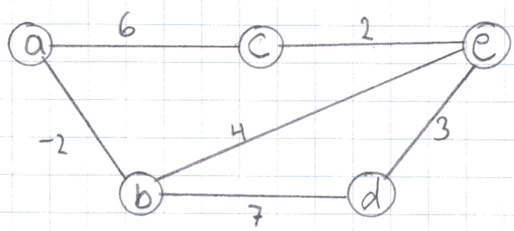
a) Draw this graph.

2p

5



or (for better perspective)



b) Consider the problem of computing a MST of this graph. Edges (a,b), (c,e), and (d,e) have already been put into the MST. Determine whether these edges were selected by Prim's algorithm or by Kruskal's algorithm. Justify your answers.

4p

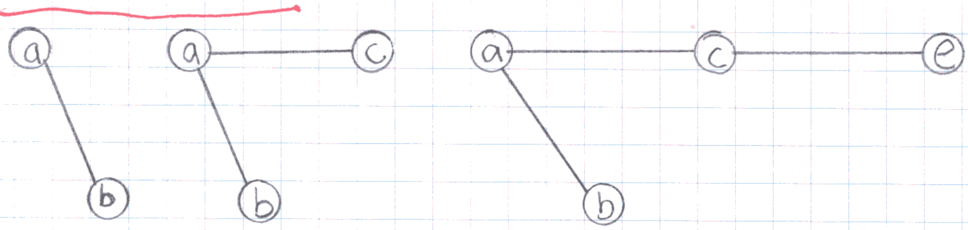
Prim's alg. (1) (a,b), (2) (b,e), (3) (c,e) and (4) (e,d)

Kruskal's alg. (1) (a,b), (2) (c,e), (3) (d,e) and (4) (b,e)

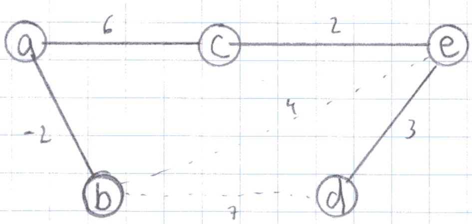
So obviously the edges were selected by Kruskal's algorithm

c) show how to perform a depth-first search and a breadth-first search on the above graph starting at node b, respectively

depth-first search:



OK



Universitetstryckeriet, Luleå

LULEÅ TEKNISKA UNIVERSITET

Namn: Samuel Gabrielsson | Personnummer: 7912090250 | Sidnr: 5

Uppgift nr: 5 cd

Poäng: 0

Lärarens anteckning:

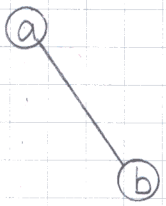
Max a-d 15p

c) Continue.... breadth-first search:

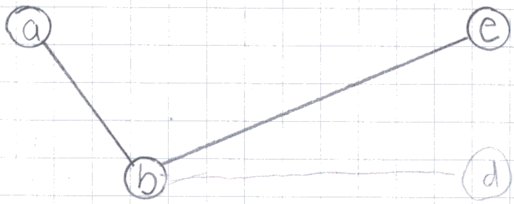
b

b | Queue: a e d

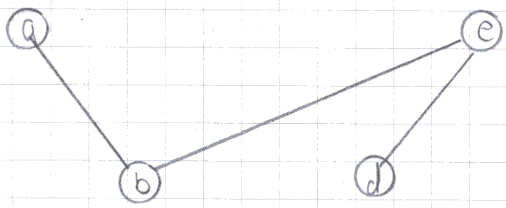
5p



b a | Queue: e d c



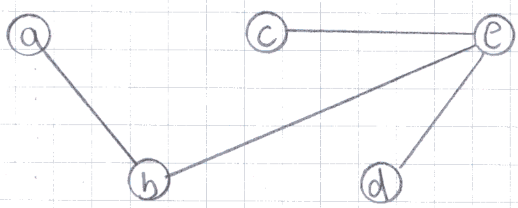
b a e | Queue: d c



b a e d | Queue: c

1

-4p



b a e d c | Queue: /

d) Explain why Bellman-Ford's single-source shortest path algorithm does not work in this graph.

4p

Although Bellman-Fords algorithm works on negative edge weights (in contrary to Dijkstras algorithm) it does not work when negative-weights are reachable from the source node s. In our case the -2 weight in (a,b)

allow

-4p

If $G=(V,E)$ contains negative-weight cycles that are reachable from start node, then the algorithm returns FALSE

Tot -5p

cycles!

Universitetstryckeriet, Luleå

Behandla endast en uppgift per sida

Skriv ej på baksidan

LULEÅ TEKNISKA UNIVERSITET

Namn: Samuel Gabrielsson

Personnummer: 791209 0250

Sidnr: 6

Uppgift nr: 6

Poäng: 10

Lärarens anteckning:

Let S be an array of n elements. Each element is colored with either red, blue or black. The task is to rearrange the array so that all the red elements precede all the blue ones and all the blue ones precede all the black ones. Your algorithm should be in-place and run in O(n) time in the worst case.

The partition method from quicksort run in time O(n) in the worst case.

We can sort the array by black and blue and ignoring the red ones because they will end up sorted utomatically by the process and gather at the end of the array S.

Consider S = { blue, red, black, blue, black, red, red, black }

and we want the elements in S to be sorted as

black < blue < red then

blue ⁝ red black blue black red red black

blue ⁝ red | black blue black red red black

black blue ⁝ red | blue black red red black

black blue blue ⁝ red | black red red black

black black blue blue ⁝ red | red red black

black black blue blue ⁝ red red | red black

black black blue blue ⁝ red red red | black

black black black blue blue ⁝ red red red

black < blue < red

LULEÅ TEKNISKA UNIVERSITET

Namn: Samuel Gabrielsson

Personnummer: 7912090250

Sidnr: 7

Uppgift nr: 7

Poäng: 0

Lärarens anteckning:

Let $A = \{a_1, a_2, \ldots, a_n\}$ be distinct numbers. Design an algorithm to compute the sum

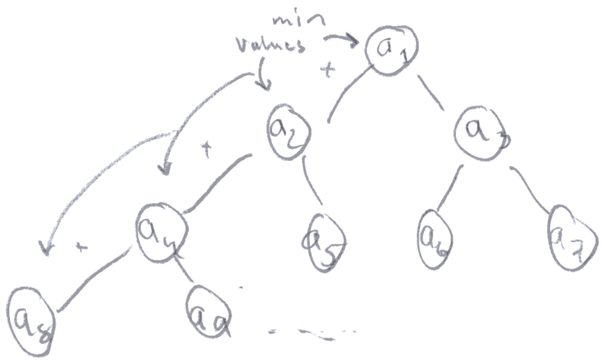
$$\sum_{i=1}^{\lfloor \log_2 n \rfloor} b_{2^i}$$

where b_j is the jth smallest element in A. You may assume that n is a power of 2 and your algorithm should run on time $O(n)$ in the worst case.

$$\sum_{i=1}^{\lfloor \log_2 n \rfloor} b_{2^i} = b_2 + b_4 + b_8 + b_{16} + \ldots + \overbrace{b_{2^{\log_2 n}}}^{= b_n}$$

We basically construct a binary search tree on $a_1, a_2, \ldots, a_n$ and perform an inorder walk in the tree. cost $O(n)$

The minimum value is the leftmost element (and maximum, the rightmost).



X

$-10p$

Namn:	Personnummer:	Sidnr:
Samuel Gabrielsson	7912090250	8

Uppgift nr: 8

Poäng: 0

Lärarens anteckning:

An anagram of a word W is another word made up of the same letters as W. For example, stop, tops and poots are anagrams of each other. Given a set of words, design an efficient algorithm to make a list for each word of all its anagrams that appear in the set. Let n denote the sum of lengths of the words in the set. If we count only the number of letter-letter comparisons used, your algorithm should run in O(n) time and space in worst case.

By using a hash table of size m we should be able to do the job.

With chaining:

Insert (T, x) will insert x at the head of list T[H(key[x])]. worst case O(1)

Search (T, k) will search for key k in the list T[H(key[x]]) worst case O(n)

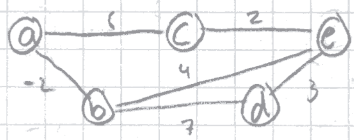
Delete (T, x) delete x from the list T[H(key[x]]) worst case O(1).

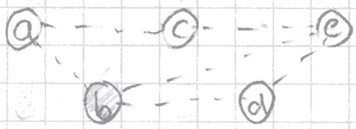
X

-10p

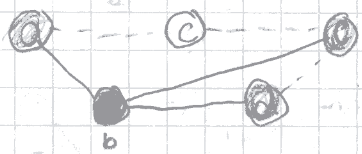
Komplettering

Breadth - First Search (BFS)



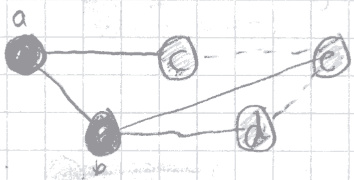


Q: b

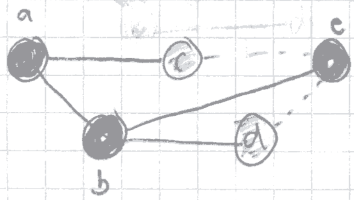


b Q: aed

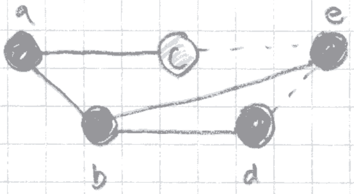
order: we pick e first.



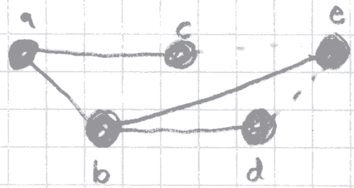
ba Q: edc



bae Q: dc



baed Q: c



baedc Q: NIL

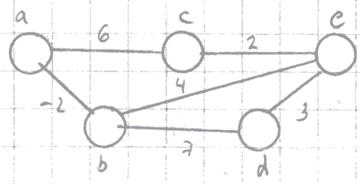
d = ?

The result of of BFS may depend upon the order in which the neighbors of a given vertex are visited. The breadth first search tree may vary but the distances d computed by the algorithm may not.

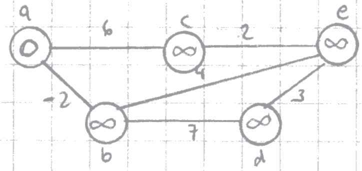
Komplettering

Bellman - Ford

- Four passes (5-1 = 4)

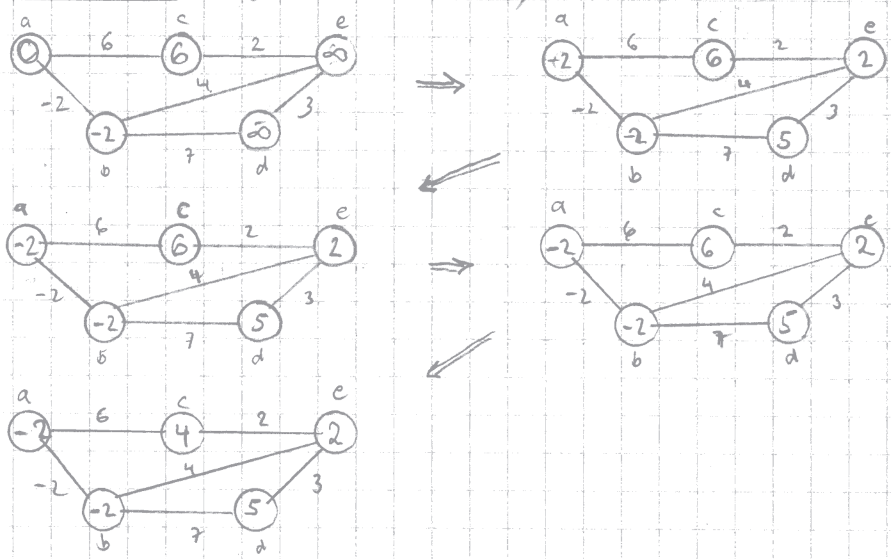


Start:

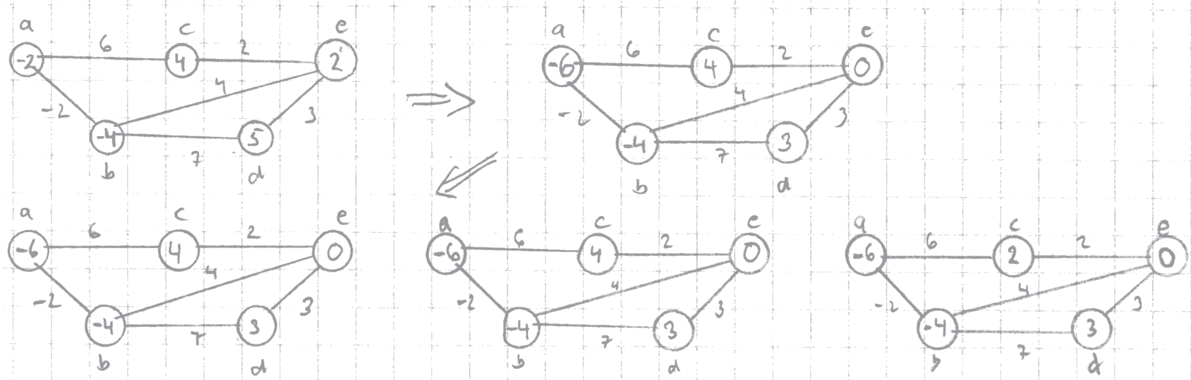


Relaxation (alphabetical order)

Pass 1:

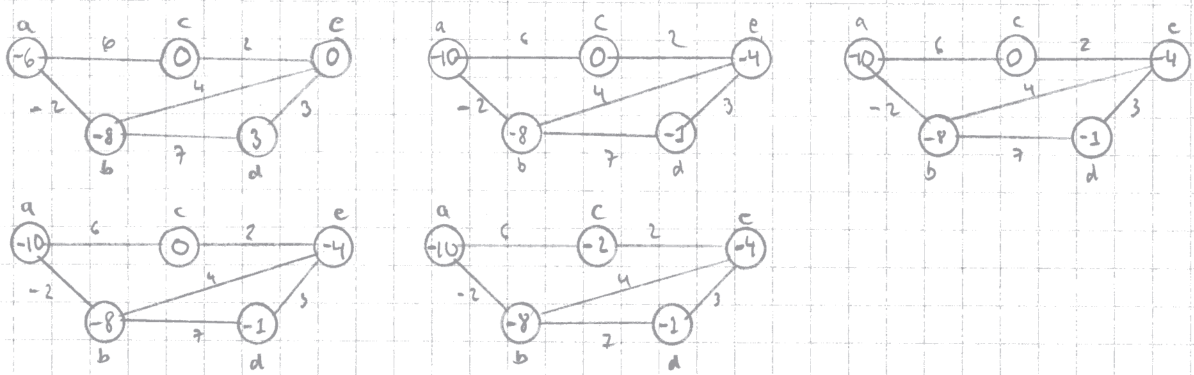


Pass 2:



Behövs ej

Pass 3:



Pass 4: (Not really necessary) lots of negative-weight cycles that are reachable from a detected:

Ex: