Soluzione   19/11/19

string	h_1	h_2	h
AA	4	2	0
BB	1	4	1
BD	3	6	2
eD	6	0	3

rank	letter
1	A
2	B
3	C
4	D
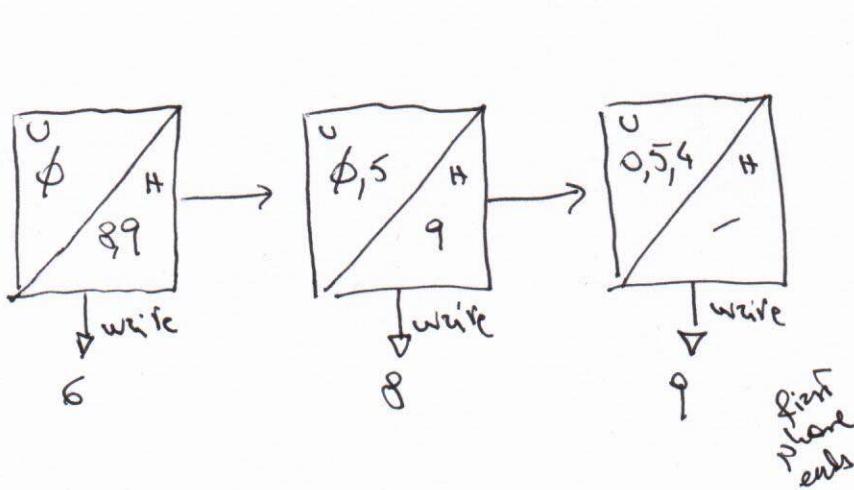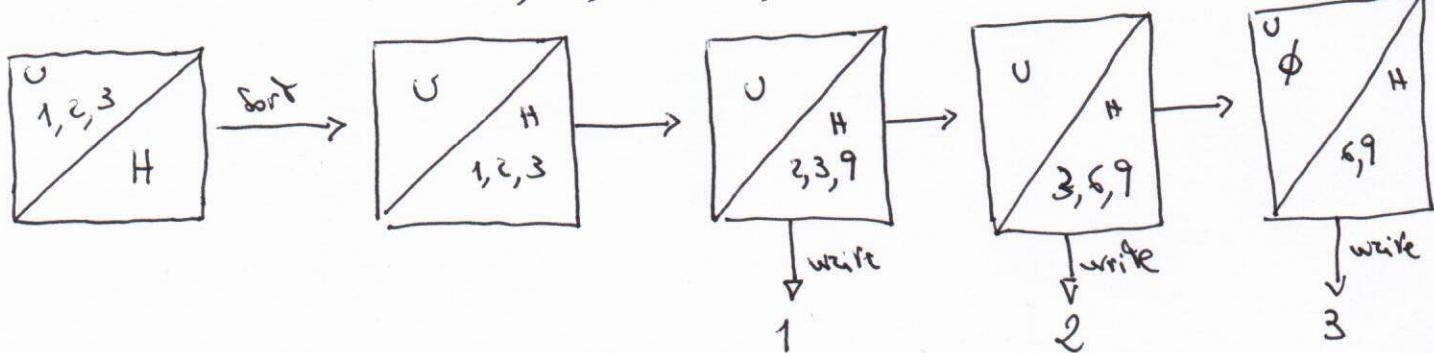
| t	$g(t)$
0	0
1	1
2	0
3	3
4	0
5	0
6	3

$$h(t) = g(h_1(t)) + g(h_2(t)) \bmod 4$$

=

Exercise 2

$$S = 1, 2, 3, 9, 6, 0, 8, 5, 4$$

U / H 1,2,3 → sort → U / H 1,2,3 → U / H 3,3,9 →write→ 1 → U / H 3,6,9 →write→ 2 → U Ø / H 6,9 →write→ 3

U Ø / H 89 →write→ 6 → U Ø,5 / H 9 →write→ 8 → U 0,5,4 / H - →write→ 9   first phase ends

‖
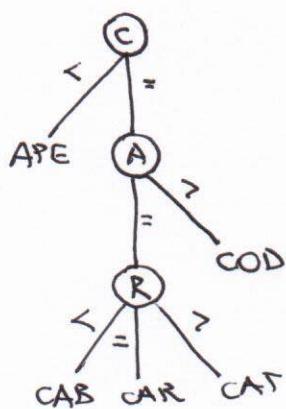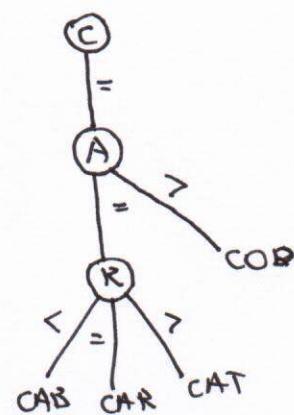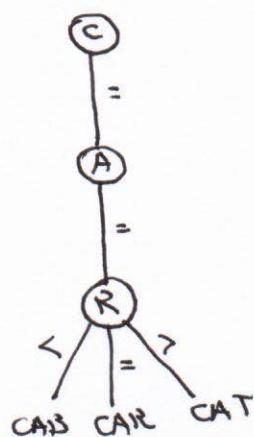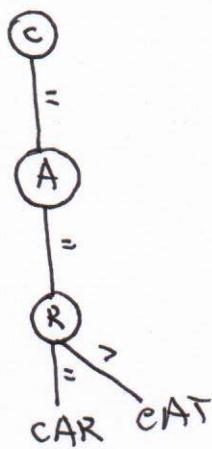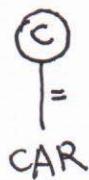
U / H 9,4,5 ~→ next steps will write these elements in order because S does not contain other items.

Exercise 3

C = CAR

C = A = R > CAR CAT

C = A = R < = ? CAB CAR CAT

C = A > COD = R < = ? CAB CAR CAT

C < APE = A > COD = R < = ? CAB CAR CAT

Exercise 4

Let us call L the list of edges stored contiguously on disk.

- We scan L, and for every edge $(i,j) \in L$, we create (j,i) That we store on disk. Call this list of "transposed edges" L^T.

- We sort L and L^T by their first component so that on disk we have adjacent the ~~edges~~ edges outgoing from each single node. We use $adj[u]$ to denote the adjacency list of u in L, and $adj^T[v]$ to denote the adjacency list of v in L^T. Notice that adj expresses the outgoing edges in G whereas adj^T expresses the ingoing edges in G.

- The key observation here is that the check for the existence of P in G can be executed independently per triples $(u_{2i+1}, *, u_{2i+3})$ over all the nodes u_{2i+1} which are specified in P.

- And the check for one triple consists just of computing the intersection between $adj[u_{2i+1}]$ and $adj^T[u_{2i+3}]$, which needs just a scanning of the two (sorted) lists.

Every intersection thus takes:

$$\frac{adj[u_{2i+1}]}{B} + 1 + \frac{adj[u_{2i+3}]}{B} + 1 \quad I/Os$$

Since the nodes are all distinct than the summation over all K nodes present in P gives $O\left(\frac{m}{B} + K\right)$, $m = |E|$

We finally need to add to this cost the I/Os needed for sorting and scanning, hence $O\left(\frac{m}{B} \log_{\frac{M}{B}} \frac{m}{B}\right)$ I/Os.

Here m is the number of edges.

Since K may be assumed small with respect to $\frac{m}{B}$ then the overall complexity is just the one of sorting.