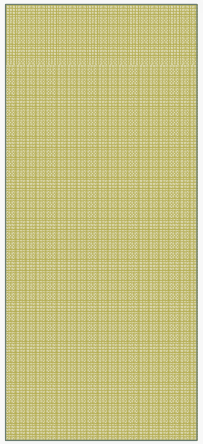
RECONSTRUCTION OF TREES USING METRIC PROPERTIES

Kr. Manev, N. Nikolov, M. Markov



A NECESSARY REMARK

Because of the presence of colleagues form schools of Thailand, **this speech is going beyond the frame of usual**

So, we will not present the results of the published in the Journal research

We will try to demonstrate **the knowledge and the abilities** necessary to **create/solve** competitive tasks

CONDICIO SINE QUA NON

- **Programming language** is the condition without which it is *impossible* to do serious work in Informatics
- We will not comment here what **kind of programming language** to use
- By different reasons the most popular nowadays are the **C-like** programming languages

CONDICIO SINE QUA NON

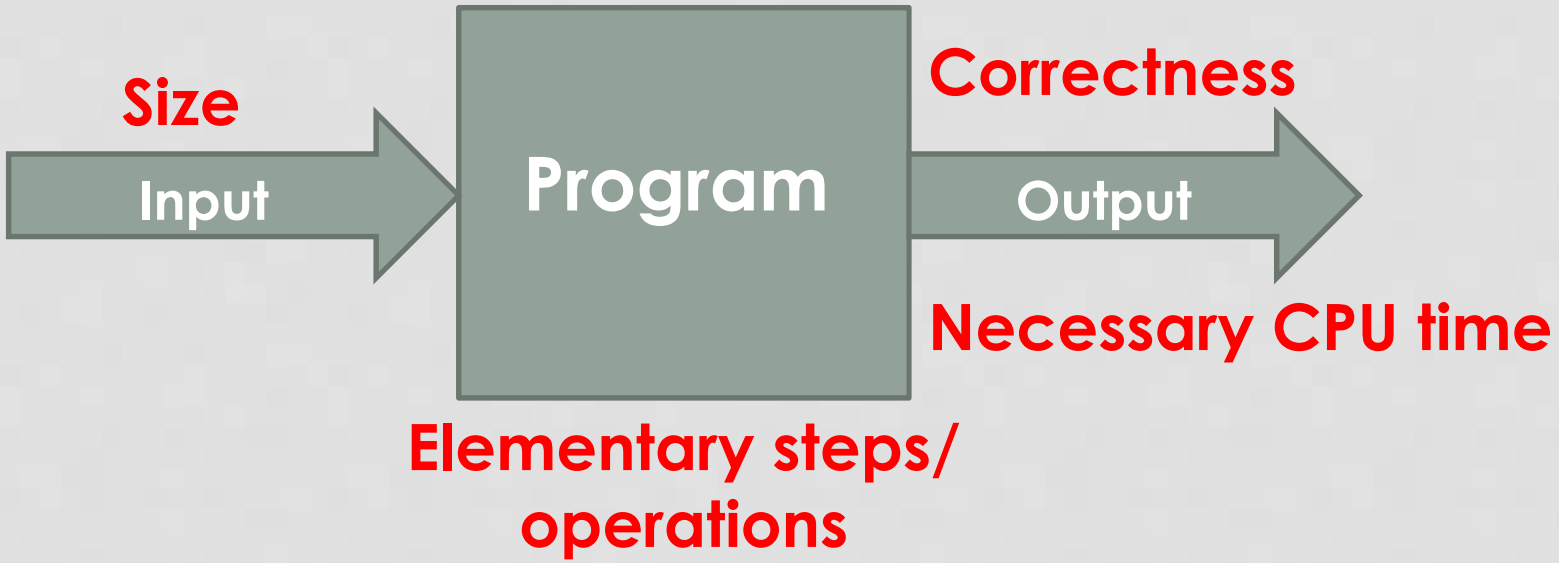
- **Programming environment (IDE)** is another condition without which it is ***very difficult*** to do serious work in Informatics
  - *Editor* (syntactically oriented)
  - *Compiler*
  - *Linkage editor/Loader*
  - *Debugger*
- DEV C++, CodeBlocks, MS Visual Studio, Eclipse, etc.

TO BE PRESENTED HERE ....

- Knowledge from the Theory of **algorithms** and the ability to apply it in practice
- Knowledge of some **discrete mathematical objects** and the ability to apply it for **modeling** of tasks situations
- Knowledge of some **algorithmic approaches** (**schemes**) which simplify creation of algorithms

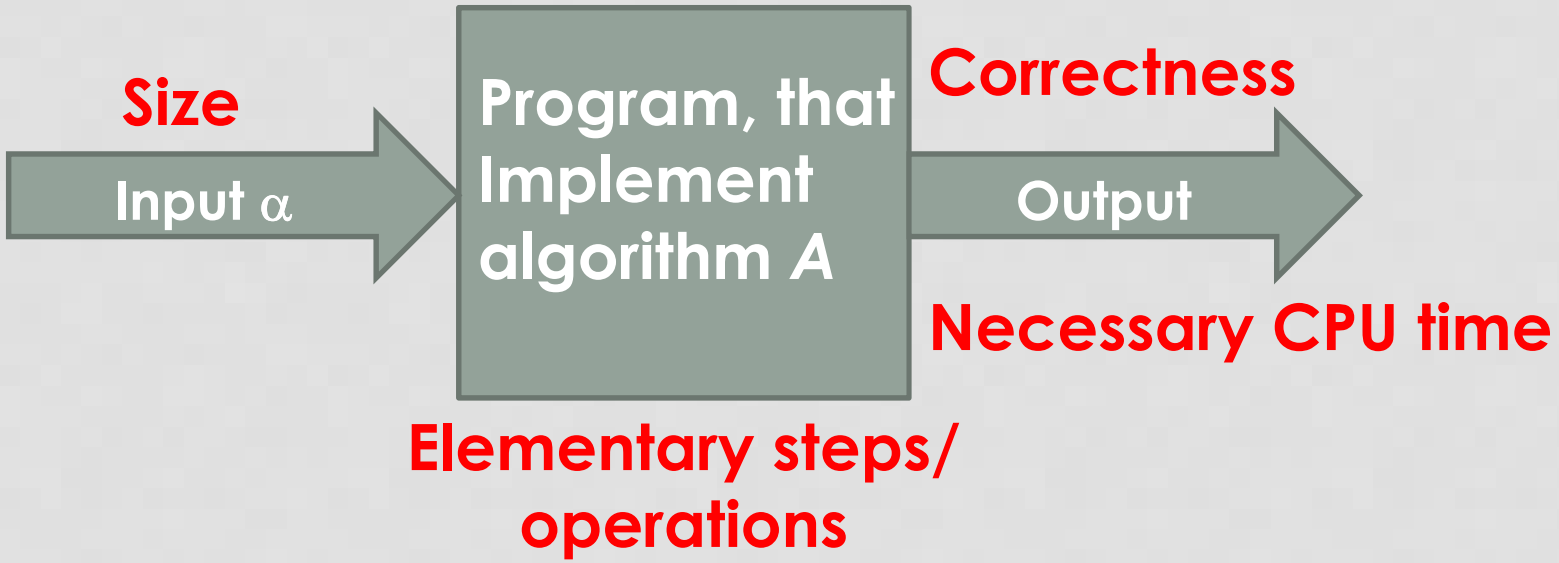
THEORY OF ALGORITHMS

- No mathematical definition of the notion **algorithm**
- **Algorithm ≅ Program**



THEORY OF ALGORITHMS

- ***Size*** of α $s(\alpha)$ = number ***N*** of characters
- $T_A(N)=\max_{\forall \alpha, s(\alpha)=N}$ {number of steps that *A* execute when working on input α} – ***time complexity*** of the algorithm *A*



THEORY OF ALGORITHMS

```
int a[],N;/* each int is a character */
bsort()
{ int i,j,t;
        3       2    3
  for(i=N-1;i>0;i--)   3+N*5+Σ_{i=1,...,N-1}f(i)
          2    3     3
    for(j=1;j<=i;j++) 2+(i+1)*6+i*30 =
           10                = 36*i+8 = f(i)
 30   if(a[j]>a[j+1])
        5        9                6
 20   {t=a[j];a[j]=a[j+1];a[j+1]=t;}
  }
```

THEORY OF ALGORITHMS

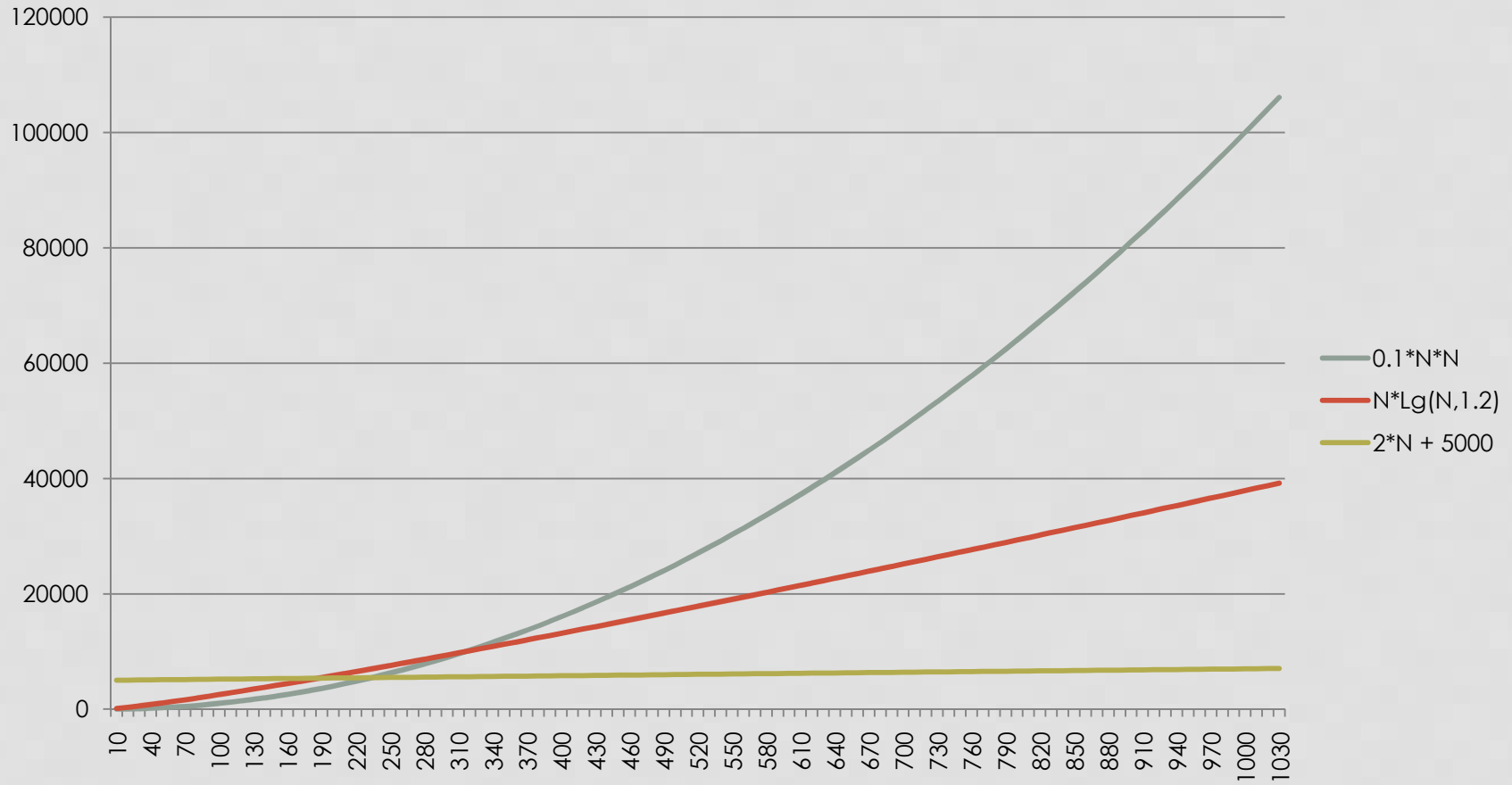
$$T_{bsort}(N) = 3 + 5N + \sum_{i=1,2,\ldots,N-1} f(i) =$$
$$= 3 + 5N + \sum_{i=1,2,\ldots,N-1} (36i + 8) =$$
$$= 3 + 5N + 8(N - 1) + 36 \sum_{i=1,2,\ldots,N-1} i =$$
$$= 13N - 5 + 18N(N - 1) =$$
$$= 18N^2 - 5N - 5 = O(N^2)$$

- **Asymptotical notation**
  - $O(g(N))$ – grows no faster than g(N) ($\leq$)
  - $o(g(N))$ – grows slower than g(N) ($<$)
  - $\Omega(g(N))$ – grows no slower than g(N) ($\geq$)
  - $\omega(g(N))$ – grows faster than g(N) ($>$)
  - $\Theta(g(N))$ – grows like g(N) ($=$)

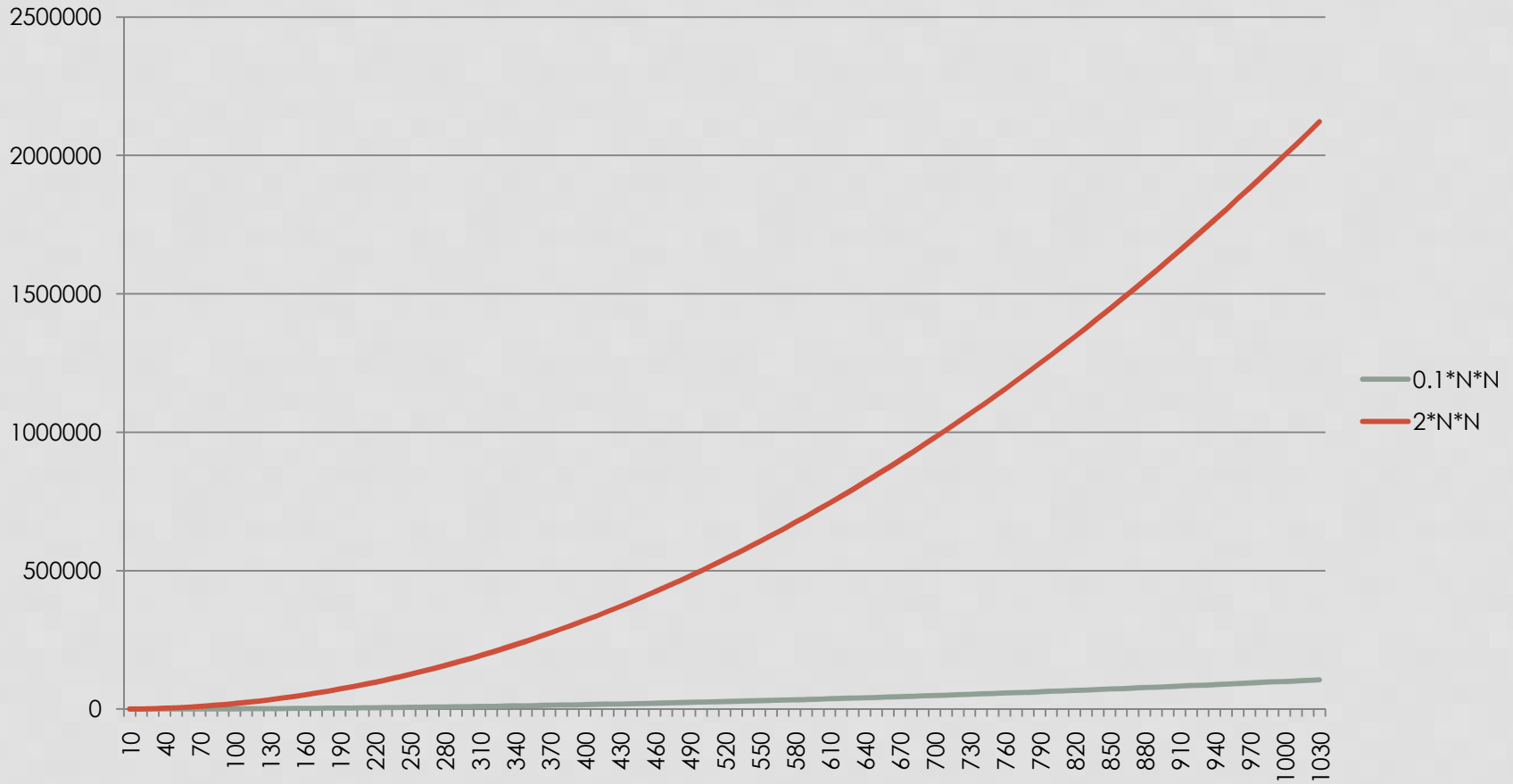
THEORY OF ALGORITHMS

- **Partial** order of classes of algorithm's time complexity
- $O(\lg N), O(N), O(N.\lg N), O(N^2), O(N^2.\lg N), \ldots, O(2^N), \ldots, O(N^N), \ldots$
- Two algorithms for the same task with comparable different complexity classes – **chose the better**
- Two algorithms for the same task within same complexity class – **chose those with better constants**

THEORY OF ALGORITHMS



THEORY OF ALGORITHMS



GRAPH THEORY

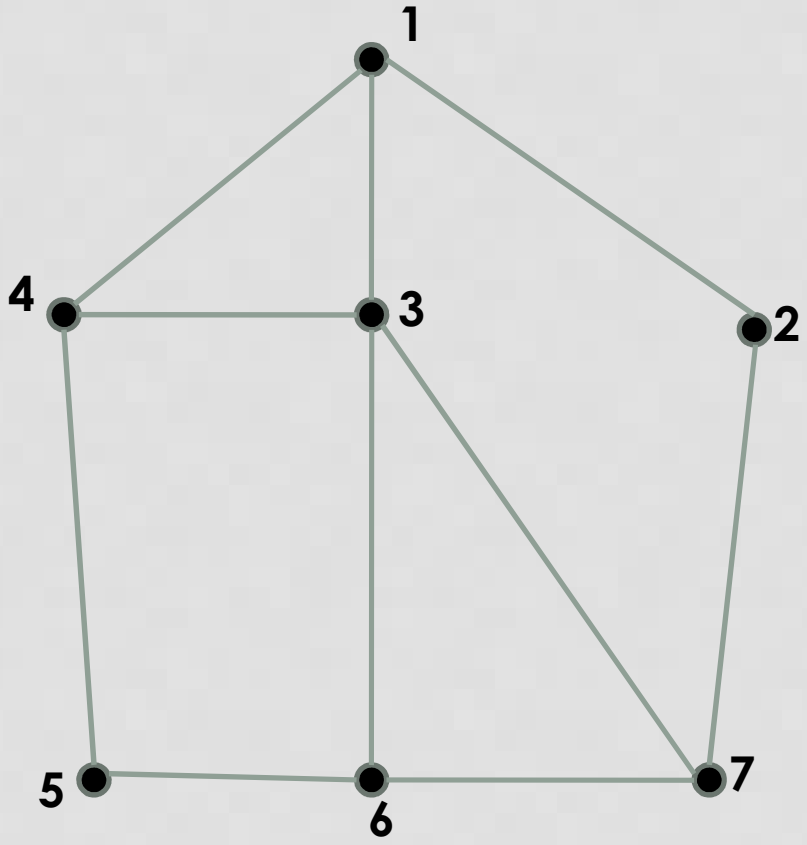
- ***Graphs*** are very simple DM objects
- Set of ***vertices***

$V = \{v_1, v_2, \ldots, v_N\}$

- Set of ***edges***

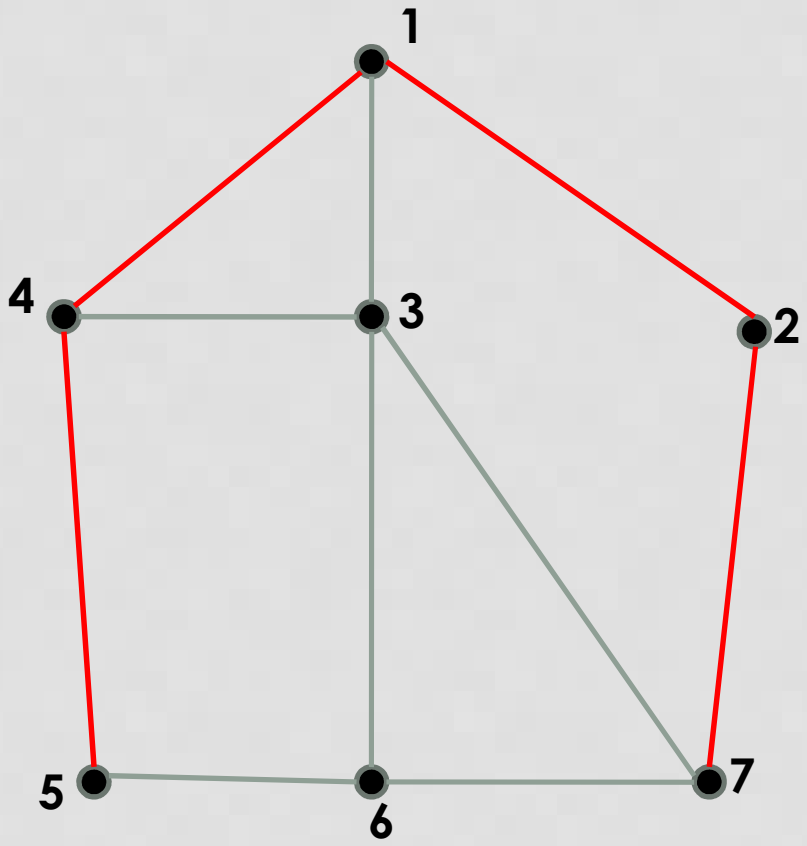
$E = \{e_1, e_2, \ldots, e_M\}$

- Each edge e_k ***links*** some couple $\{v_i, v_j\}$ of vertices



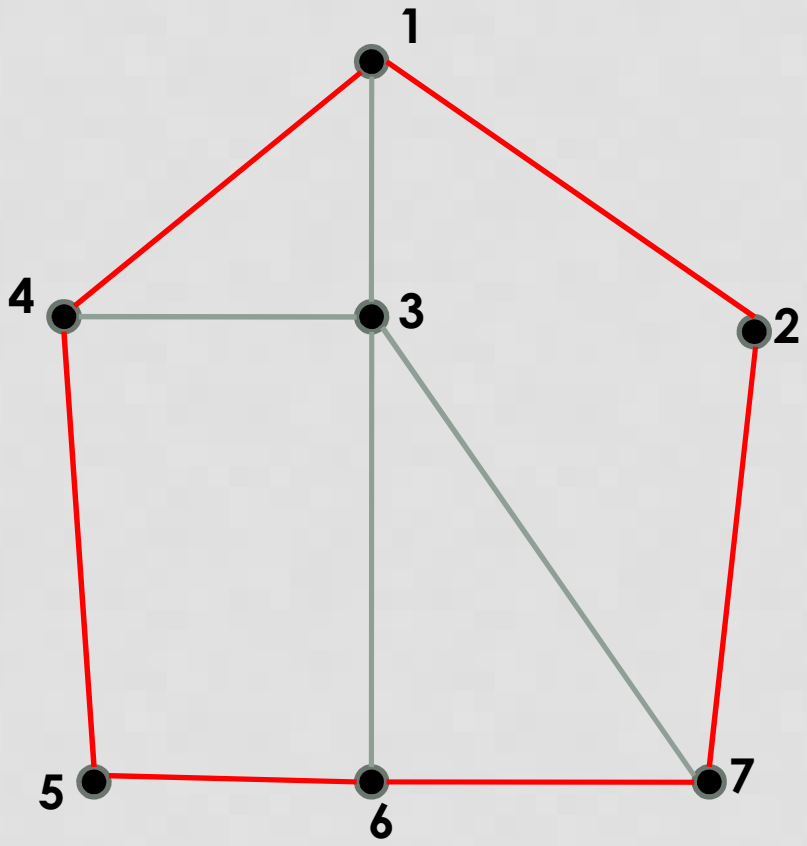
GRAPH THEORY

- ***Path*** is a sequence of vertices, each two consecutive vertices linked by an edge
- The sequence (5, 4, 1, 2, 7) is a path **from *5* to *7***
- Number of edges – ***length*** of the path



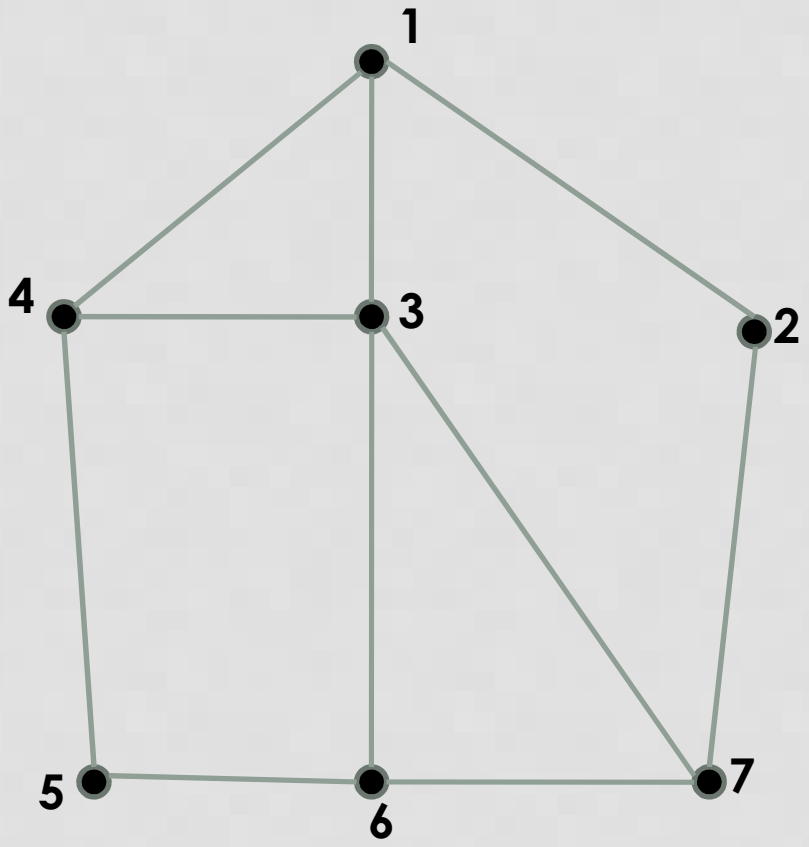
GRAPH THEORY

- ***Cycle*** in the graph is a path, such that starts and finishes in a same vertex
- The sequence (5, 4, 1, 2, 7, 6, 5) is a cycle



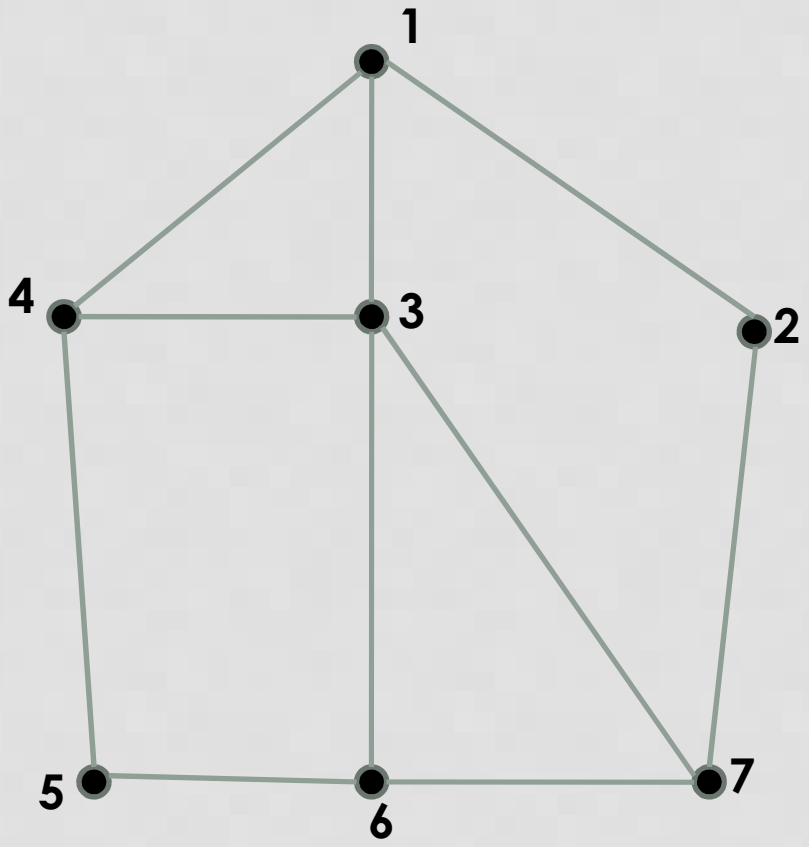
GRAPH THEORY

- Graph $G(V,E)$ is ***connected*** if each 2 vertices are connected by a path
- The graph on the Figure is connected
- Deleting edges we could obtain ***not connected*** graph



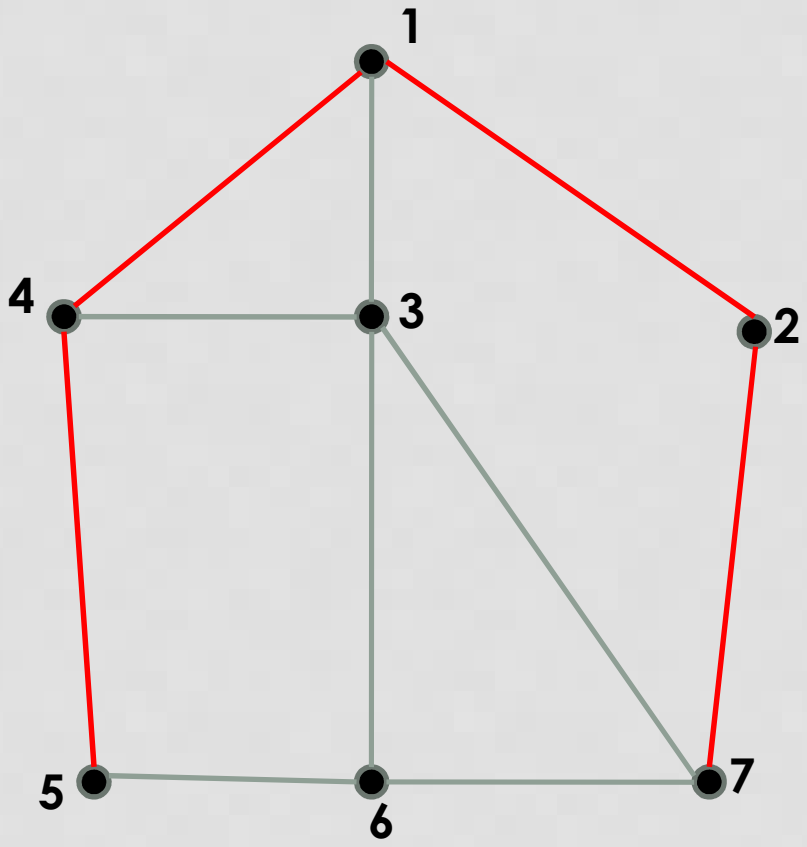
GRAPH THEORY

- Connected graph $G(V,E)$ without cycles is called ***tree***
- To turn the graph on the Figure in tree we have to **cut each cycle**.
- After deleting an edge of a cycle **the graph remains connected**



GRAPH THEORY

- *A **path** of minimal length from v_i to v_j among all such pats is called **shortest path***
- (5, 4, 1, 2, 7) is not a shortest path from 5 to 7. A shortest path from 5 to 7 is (5,6,7)



ALGORITHMIC SCHEMES

- An ***algorithmic scheme*** (AS) is a procedure very similar to an algorithm but, in some sense, **uncompleted**, and so - not solving any task
- If we have a task for which an algorithmic scheme is appropriate, we **could complete the procedure to an algorithm**
- Using algorithmic schemes **simplify**, in many cases, **development** of algorithms

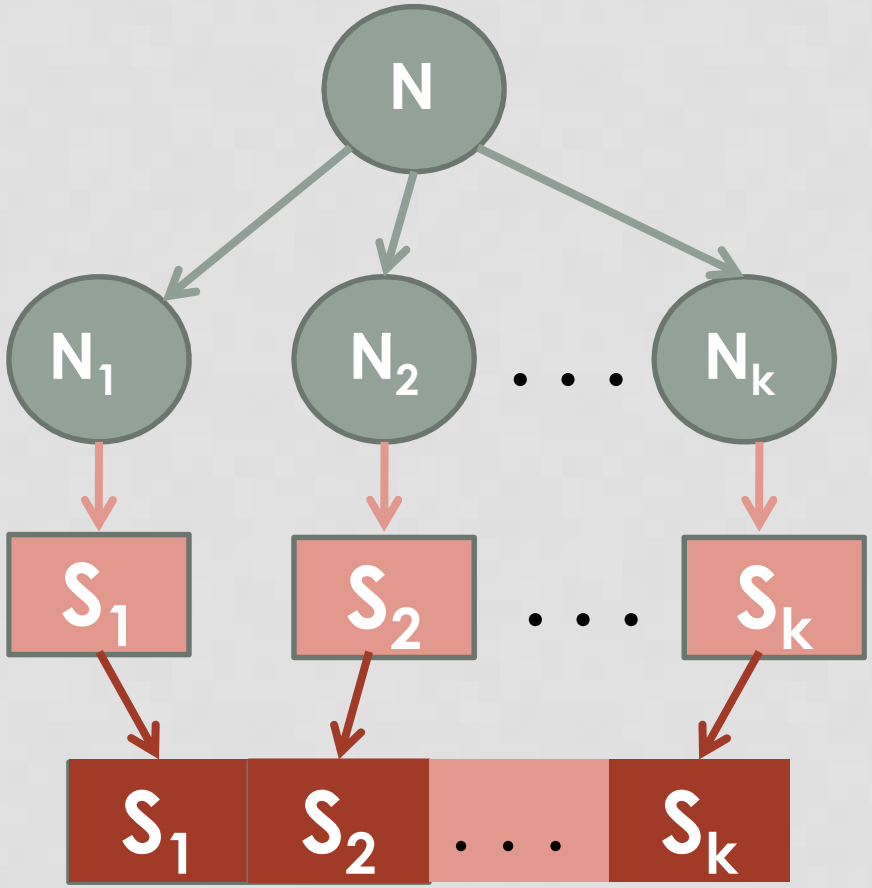
ALGORITHMIC SCHEMES

- Very popular AS are:
  - Divide and conquer
  - Breadth-first traversal of graph
  - Breadth-first traversal of graph
  - Euler traversal of graph
  - Dynamic programming
  - Greedy
  - Backtracking

ALGORITHMIC SCHEMES

AS Divide and conquer

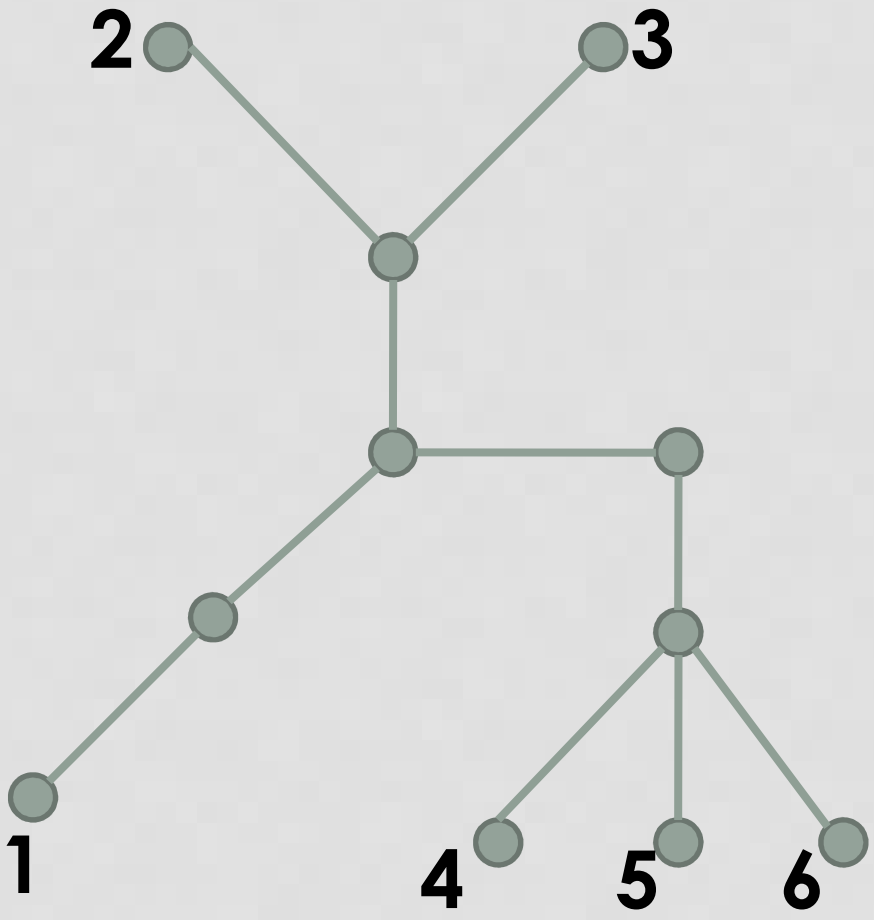
- **Split** the task of size *N to* **smaller** subtasks of same kind
- **Solve** subtasks
- **Compose solution** from the solutions of the subtask



OUR TASK

- For the NO in Math one of the co-authors – prof. N. Nikolov, leader of Bulgarian team for IOM, proposed the following (in following the vertices will be labeled with 1, 2, 3, …)
- TASK: **A tree *D* has *L* vertices of degree 1 (i.e. that are linked to exactly one other vertex) – 1, 2, …, L. The length l_{ij} of the single path from *i* to *j* is given for all *i* and *j*, $1 \leq i < j \leq L$. Reconstruct the tree!**

OUR TASK



	1	2	3	4	5	6
1		4	4	5	5	5
2	4		2	5	5	5
3	4	2		5	5	5
4	5	5	5		2	2
5	5	5	5	2		2
6	5	5	5	2	2	

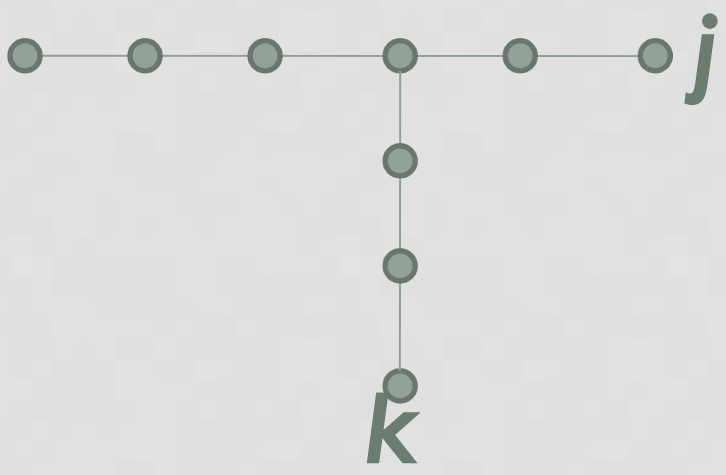
OUR TASK

- First, let us remark that the **length l_{ij} of the shortest path** between 2 vertices is a ***distance*** in the usual math sense (like the Euclidean), because:
  - *Axiom 1*: $l_{ij} \geq 0$, $\forall$ *i,j,* and $l_{ij} = 0$ iff $i = j$;
  - *Axiom 2*: $l_{ij} = l_{ji}$, $\forall$ *i* and *j*;
  - Axiom 3 (triangle inequality): For $\forall$ *i, j* and *k,* the inequality $l_{ij} + l_{jk} \geq l_{ik}$
- So, we will call l_{ij} a ***distance*** between *i and j* and will use the existing analogy to the Euclidean distance

OUR TASK

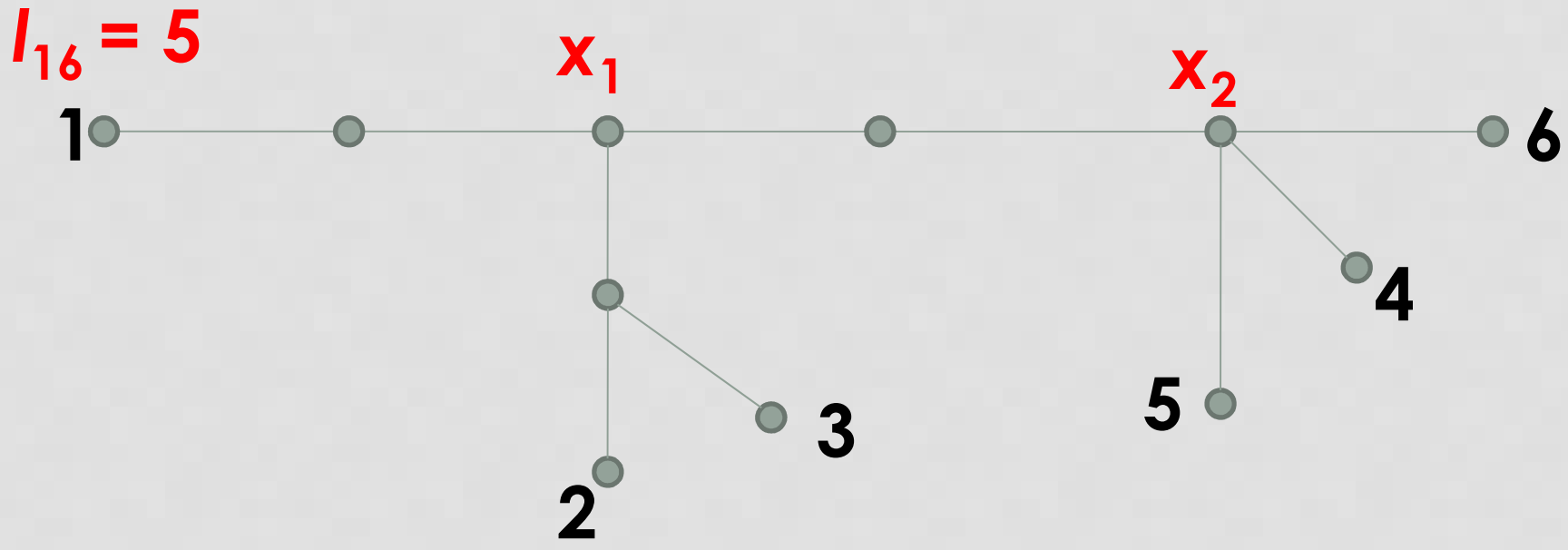
Lemma. Let $D = (V, E)$ be a tree and i, j and k be pairwise distinct vertices. There is a unique vertex x that belongs to the three pats– from i to j, from j to k, and from i to k. From the Proof (omitted) follows that:

- $l_{ix} = (l_{ij} + l_{ik} - l_{jk})/2$,
- $l_{jx} = (l_{ij} + l_{jk} - l_{ik})/2$,
- $l_{kx} = (l_{ik} + l_{jk} - l_{ij})/2$.



OUR TASK

- We will try the *AS Divide and conquer*
- Decomposition in subtasks, solving the subtasks and composition will be done **in parallel**



OUR TASK

Divide and conquer algorithm

- Chose two pending vertices ***i*** and ***j*** and build a path of length l_{ij} between them (***composition/reconstruction***)
- For each ***k*** on the path, $k \neq i$ *and* $k \neq j$,
  - identify (by the Lemma) the vertex **x** and the sub-tree T_x rooted in x (**splitting in subtasks**)
  - solve recursively the task for T_x. Use the Lemma **to find** l_{ip}, when necessary, for each pending vertex *p* (**decomposition**)

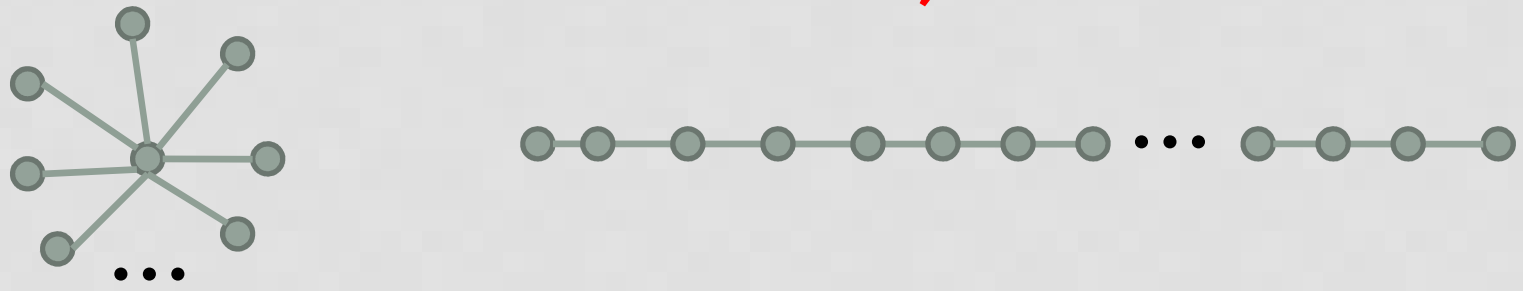
OUR TASK

- Last step in preparation of the task for a contest, before creating the test cases, is the estimation of *time complexity* of the algorithm
- This is important step, because without the estimation of the complexity is difficult **to choose the size of test cases**
- Unfortunately, **finding the time complexity** of algorithms for reconstruction of graphs, and especially of tree by its metric properties **could be not easy**

OUR TASK

- The usual expectation is that $T_A(N)$ **is growing** with growing of N
- It is not the same with the considered task – here the complexity depend too much of **the number M of the inner vertces**

$L = 2, M = 10000000000$

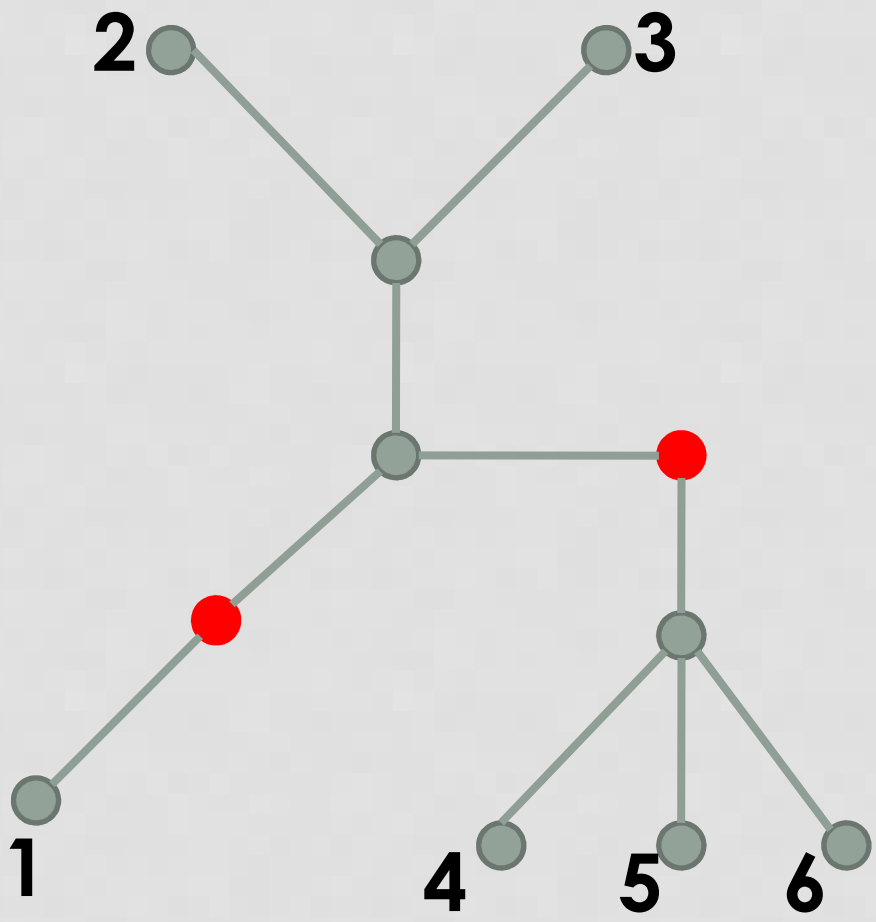


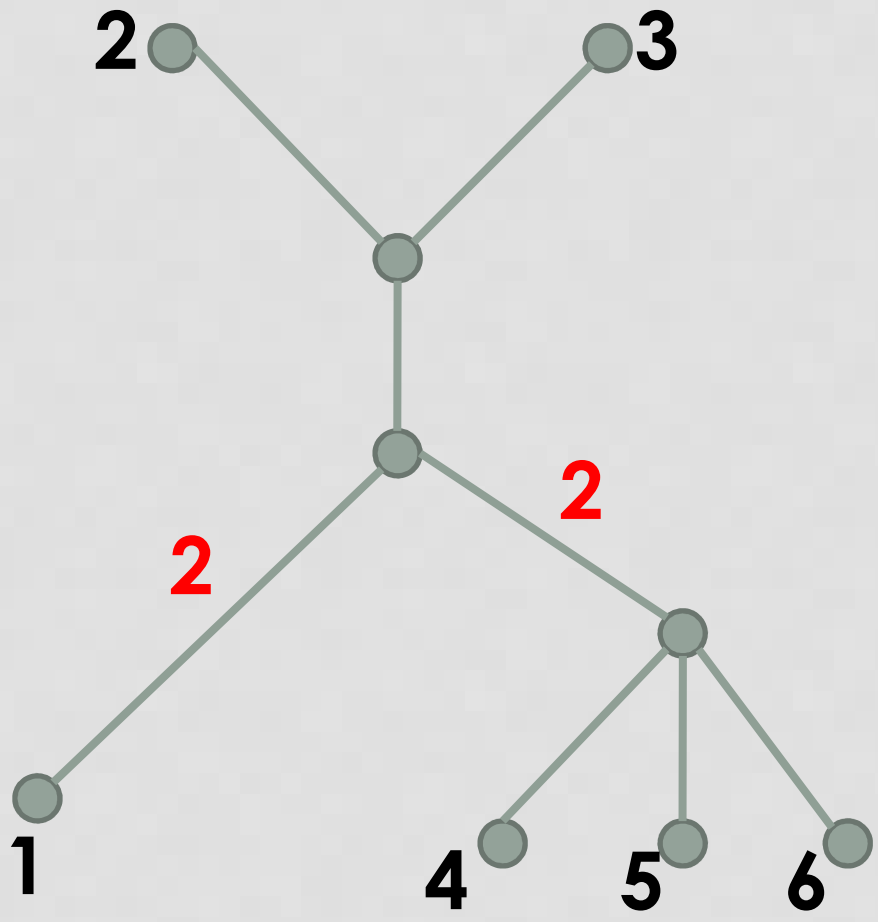
$L = 10000000000, M = 1$

OUR TASK

- One way to manage the problem is to ask not reconstruction, but only the num-'ber of inner vertices (see the algorithm)
- The other is to eliminate the vertices of degree 2. Looking again the algorithm we could see that these vertices do not generate subtasks and so – could be reduced.

OUR TASK





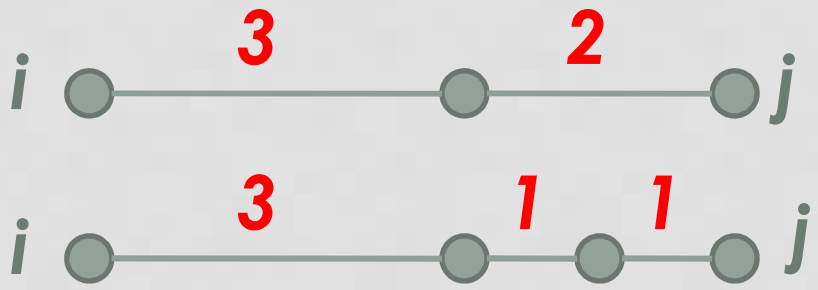
OUR TASK

- Using this simplification we succeed to estimate the complexity of the algorithm
- It happens that our algorithm, let us call it *MNM* make $O(L^2)$ steps solving the task for three with ***L* pending vertices**
- Because the size ***N*** of the input of the task is $\Theta(L^2)$, the complexity of our algorithm is $T_{MNM}(N) = O(N)$ – nice linear algorithm

OUR TASK

- It is interesting to try to solve the task when the edges of the tree are **weighted** and the length of a path is a **sum the weights** of edges
- But it is **impossible**. Trivial counterexample is the tree with 2 pending vertices

$l_{ij} = 5$



OUR TASK

- The reason for the impossibility to solve the task for weighted graphs is again in the vertices of degree 2.
- Theorem. If the graph is **without vertices of degree 2** than it could be reconstruc-ted by the distances among the pending veritces
- The algorithm is practically the same and its complexity is again **linear**

CONCLUSIONS

- Task was included un the test set of **SEE Regional contest of ACM ICPC** – 10.2010
- 55 teams, **33** of which from traditionally **strong** Ukraine, Romania and Bulgaria
- Only **11 teams submitted correct solution** (out of 10 problem - one was not solved and 1 solved by only 5 teams)
- So, we could classify the difficulty of the tasks as **above the average**

CONCLUSIONS

- We try to solve another tasks with the same approach
- **TASK: Let *D*(*V*,*E*) is a weighted tree and the distances l_{ij} among all of its vertices. Reconstruct the tree.**
- Trivial modification *MNMW* of *MNM* solved the problem (solution is in the paper)
- The time complexity of *MNMW* is again linear

CONCLUSIONS

- Task was included un the test set of Bulgarian **Autumn tournament**, 11.2010, in the second age group – 16-17 years old
- **53 students** from Bulgaria, Croatia, Greece, Macedonia, Serbia and Romania,
- **43 students** submitted solution, **15** obtained at least 70%, and 10 of them – 100%
- So, we could classify the difficulty of the this tasks as **below the average**

CONCLUSIONS

- Dependence of the time complexity from a **hidden parameter** of the graph make the first task unattractive
- Some efforts have to be spent during the training of contestants in order to be able **to manage tasks with hidden parameters**
- Graph reconstruction problems seem to be a good way to **enlarge the scope of the tasks** given at programming contests.

CONCLUSIONS

- **Open problem:** to reconstruct the tree from the distances between all vertices (or between the outer vertices) when it is **not given which distance is between which two vertices**.
- For solutions of the discussed two problems **this knowledge is crucial**.

THANKS FOR YOUR ATTENTION!

ANY QUESTIONS?