Brute force algorithm implementation

- The brute implementation of the the 'k' size vertex cover problem is in **python**.
- The program uses three external libraries for computations:
    1. **sys** for standard input reading from and writing into a file
    2. **itertools** for generating 'k' sized subsets from the vertices set
    3. **time** for measuring the time to run the main function (time for generating the graph, vertices and edges lists from the input is excluded)

Functions:

1. checkIfVertexCover

Input: a list of vertices (lst) and the edge list (edges)

Computation: A loop iterates over the vertices list and removes all the edges in the edge list that the particular vertex covers. When the loop ends, if the edge list is empty, it returns True (the vertices cover the entire edge set). Otherwise, it returns False.

Output: True / False

2. Main function

The program accepts the input file name as a command line input. It then reads through the file line by line and generates the graph, edges and vertices lists.

It then starts the timer and uses the itertools library and generates a list (allPossibleKCombinations) of all possible subsets of k-size from the vertices set. Then it passes each set of allPossibleKCombinations to the checkIfVertexCover function and checks if each of the k-size vertices set is a vertex cover of the original graph. As soon as it encounters that a particular instance of the k-sized vertices is a vertex cover, it breaks out of the loop and prints 'True' along with the solution (set of vertices). It also displays the time (in seconds) for which which the main function got executed.

Execution:

To execute the program:

```
python bruteForce.py 'input.txt' > 'output.txt'
```