Fixed Parameter algorithm implementation

- The fixed parameter 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. checkNoEdges

Input: a graph (a dictionary of lists)

Computation: A loop iterates over the graph and checks if the length of each list of adjacent vertices of each vertex in the graph is zero. If lengths of all the vertex lists is zero, it returns True (Graph has no edges). Otherwise, it returns False.

Output: True / False

2. numberOfEdges

Input: a graph (a dictionary of lists)

Computation: A loop iterates over the graph and counts the length of each list of adjacent vertices of each vertex in the graph. If then sums up all the list lengths and returns half of that sum value (each edge is counted twice).

Output: An integer (number of edges in the graph)

3. pickRandomEdge

Input: a graph (a dictionary of lists)

Computation: A loop iterates over the graph and picks the first edge it encounters, stores the two end vertices of the graph as a list and returns it.

Output: Two end vertices of the edge (as a list)

4. removeVertex

Input: a graph (a dictionary of lists) and a list (list of vertices to be removed from the graph)

Computation: A loop iterates over the vertex list and searches for the vertex in the graph and adjacency lists of each vertex in the graph. If it is found, it is deleted from the graph. Similarly, the edges that are covered by the vertex are also deleted in the edge list. The resultant graph is returned

Output: a graph (a dictionary of lists)

5. VCDbst

Input: a graph (a dictionary of lists), integer k (VC parameter) and an integer originalK (VC parameter)

Computation: This function basically an implementation of the DBST after an f(k) kernel is found for the original graph. It picks an edge and branches on it by removing either of the vertices on the edge and reduces k by one as it travels down by one level. It will return True/False (whether the graph has a VC of size originalK or not.)

Output: True / False

6. def kernelize(graph, k):

Input: a graph (a dictionary of lists) and an integer k (VC parameter)

Computation: This function applies the rules of kernelization and removes all appropriate vertices to produce the kernel instance of the original graph. Then it passes the kernel of the graph and the resultant value of k to the function VCDbst.

7. 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 calls the kernelize function on the input graph and paramter k that returns True if the graph has a vertex cover of size K.

Execution:

To execute the program:

python parametrized.py 'input.txt' > 'output.txt'