CSC 520 Summer 2024
HW #2 — Complexity
25 points

Submit a .zip archive containing 3 files, or each file separately:

- A plain text .txt file that affirms the honor code. For individuals: `On my honor as an SFSU student, I, <name>, have neither given or received inappropriate help with this homework assignment.` For groups: `On our honor as SFSU students, we, <names>, have neither given or received inappropriate help with this homework assignment. All group members participated in this work, and all concur with the submission.`
- A .py Python source code file with your problem 1 solution.
- A .py Python source code file with your problem 2 solution.

1. (12.5 points). An instance of StartEndSameColor is a white-space delimited list of edges, separated by a semi-colon (';') from a white-space delimited list of node colorings. An edge is formatted as two node names separated by a comma (e.g., `"a,b"`), and a node coloring as a node name and a color separated by a colon (e.g., `"b:blue"`). For example, `'a,b b,c c,d d,a ; a:red b:blue c:yellow d:blue'` is an instance of StartEndSameColor.

A StartEndSameColor instance is a positive instance iff it encodes a graph with a directed cycle that goes through a node of every color in the colorings; and no color occurs twice except for the color of the first and last nodes in the cycle. For example:

If *I* = `'a,b  b,c  c,d  d,a; a:blue b:red c:yellow d:blue'` then `'a,b,c,d'` verifies *I* as a positive instance of StartEndSameColor.

If *I* = `'a,b  b,c  c,d  d,e e,f f,a; a:red b:blue c:yellow d:blue e:red f:red'` then `'a,b,c,d,e,f'` cannot verify *I* as a positive instance of StartEndSameColor because there are three red nodes.

If *I* = `''a,b  b,c  c,d  d,a; a:red b:blue c:blue d:red'` then `'a,b,c,d,e,f'` cannot verify *I* as a positive instance of StartEndSameColor because there are 3 duplicate blues in the interior of the cycle.

Complete VfyStartEndSameColor-template.py, included with the test materials, so that it verifies StartEndSameColor in polynomial time. The only required import is graph.py from the *WCBC* library, also included with the test materials. An important indication that your changes are successful is that none of the test cases included with the module are flagged with **, meaning that the actual result differed from the expected one.

See below for the test harness output from VfyStartEndSameColor-template.py. The explanations are hand coded in the test harness:

```
VERBOSE: VfyStartEndSameColor() unreasonable length hint or solution
test #1 VfyStartEndSameColor("a,b  b,c  c,d  d,a; a:red b:blue c:yellow
d:blue","maybe","a,b,c,d"): expected "unsure", received "unsure"
test #1 Explanation: solution too long

VERBOSE: VfyStartEndSameColor() solution != "yes"
test #2 VfyStartEndSameColor("a,b  b,c  c,d  d,a; a:red b:blue c:yellow
d:blue","no","a,b,c,d"): expected "unsure", received "unsure"
test #2 Explanation: can't verify negative instance

VERBOSE: VfyStartEndSameColor() Cycles must have at least 2 nodes.
test #3 VfyStartEndSameColor("a,a; a:red","yes","a"): expected "unsure", received
"unsure"
test #3 Explanation: One node does not a cycle make
```

```
VERBOSE: VfyStartEndSameColor() "e" in hint but not graph
test #4 VfyStartEndSameColor("a,b  b,c  c,d  d,a ; a:red b:blue c:yellow
d:red","yes","e,a,b,c,d"): expected "unsure", received "unsure"
test #4 Explanation: e not in graph

VERBOSE: VfyStartEndSameColor() "a,b  b,a  c,d  d,a ; a:red b:blue c:yellow d:red" is a
positive instance, all verifications succeeded
** test #5 VfyStartEndSameColor("a,b  b,a  c,d  d,a ; a:red b:blue c:yellow
d:red","yes","a,b,a,d"): expected "unsure", received "correct"
test #5 Explanation: "a" occurs twice

VERBOSE: VfyStartEndSameColor() "a,b  b,d  c,d  d,a; a:red b:blue c:yellow d:red" is a
positive instance, all verifications succeeded
** test #6 VfyStartEndSameColor("a,b  b,d  c,d  d,a; a:red b:blue c:yellow
d:red","yes","a,b,c,d"): expected "unsure", received "correct"
test #6 Explanation: No b-c edge

VERBOSE: VfyStartEndSameColor() "a,b  b,c  c,d  d,a; a:red b:blue c:blue d:red" is a
positive instance, all verifications succeeded
** test #7 VfyStartEndSameColor("a,b  b,c  c,d  d,a; a:red b:blue c:blue
d:red","yes","a,b,c,d"): expected "unsure", received "correct"
test #7 Explanation: duplicate blues in interior of cycle

VERBOSE: VfyStartEndSameColor() "a,b  b,c  c,a  d,a; a:red b:blue c:red d:yellow" is a
positive instance, all verifications succeeded
** test #8 VfyStartEndSameColor("a,b  b,c  c,a  d,a; a:red b:blue c:red
d:yellow","yes","a,b,c"): expected "unsure", received "correct"
test #8 Explanation: no yellow in cycle

VERBOSE: VfyStartEndSameColor() "a,b  b,c  c,d  d,e e,f f,a; a:red b:blue c:yellow
d:blue e:red f:red" is a positive instance, all verifications succeeded
** test #9 VfyStartEndSameColor("a,b  b,c  c,d  d,e e,f f,a; a:red b:blue c:yellow
d:blue e:red f:red","yes","a,b,c,d,e,f"): expected "unsure", received "correct"
test #9 Explanation: 3 reds

VERBOSE: VfyStartEndSameColor() red starts cycle and yellow ends it.
test #10 VfyStartEndSameColor("a,b  b,c  c,d  d,a; a:red b:blue c:white
d:yellow","yes","a,b,c,d"): expected "unsure", received "unsure"
test #10 Explanation: start/end color not same

VERBOSE: VfyStartEndSameColor() "a,b  b,c  c,d  d,a; a:blue b:red c:yellow d:blue" is a
positive instance, all verifications succeeded
test #11 VfyStartEndSameColor("a,b  b,c  c,d  d,a; a:blue b:red c:yellow
d:blue","yes","a,b,c,d"): expected "correct", received "correct"
test #11 Explanation: a-b-c-d traverses every color; only first and last have same
color
```

2. (12.5 points) A **Graph** for this problem is defined as an unweighted, undirected graph, encoded as a white space delimited sequence of edges.

A **node cover** is a subset of the nodes in a Graph, such that every edge in the Graph has one or both endpoints in a node in the subset.

An instance of **HalfNodeCover** is a Graph.

A HalfNodeCover instance is a positive instance iff no more than half the nodes in the Graph form a node cover subset. For example, `'a,b a,c a,d'` is a positive instance of HalfNodeCover, because {`a`} covers the graph, and |{`a`}| ≤ 2 = 0.5 * the number of nodes in the graph; and `'a,b a,c b,d c,d d,e'` is a negative instance of HalfNodeCover, because the smallest node cover subset is 3 > 2.5 = 0.5 * the number of nodes in the graph.

An **independent set** is a subset of the nodes in a Graph, such that there are no edges between any two nodes in the subset. For example, `'a,a a,c a,d a,e b,c b,d b,e c,d c,e d,e'` has an independent set of size 2, {`a`,`b`}, since there is no edge connecting nodes `a` and `b`.

An instance of **QuarterIndependentSet** is a Graph.

A QuarterIndependentSet instance is a positive instance iff at least 1/4 of the nodes in the Graph form an independent set. For example, `'a,c a,d a,e b,c b,d b,e c,d, c,e, d,e'` is a positive instance because {`a`, `b`} forms an independent set of size 2, out of a total of 5 nodes. But `'a,b a,c a,d a,e b,c b,d b,e c,d, c,e, d,e'` is a negative instance, because it has no independent set larger than 1.

Complete PolyreduceHalfNodeCoverToQuarterIndependentSet-template.py, following the instructions in the source code, to demonstrate the polyreduction HalfNodeCover $\leq_P$ QuarterIndependentSet. The template, along with all needed imports, is included with the test materials; getting the expected results from all the test cases is always a positive sign.

```
VERBOSE: PolyReduceHalfNodeCoverToQuarterIndpenedentSet() -
quarterIndependentSet_instance = "a,b  a,c a,d"
test #1 vfyHalfNodeCoverViaVfyQuarterIndepedentSet("a,b  a,c a,d","yes","a"): expected
"correct", received "correct"
test #1 Explanation: {a} covers all 4 nodes

VERBOSE: PolyReduceHalfNodeCoverToQuarterIndpenedentSet() -
quarterIndependentSet_instance = "a,b  a,c c,d"
test #2 vfyHalfNodeCoverViaVfyQuarterIndepedentSet("a,b  a,c c,d","yes","a c"):
expected "correct", received "correct"
test #2 Explanation: {a,c} covers all 4 nodes

VERBOSE: PolyReduceHalfNodeCoverToQuarterIndpenedentSet() -
quarterIndependentSet_instance = "a,b  b,c c,d e,f"
test #3 vfyHalfNodeCoverViaVfyQuarterIndepedentSet("a,b  b,c c,d e,f","yes","a c e"):
expected "correct", received "correct"
test #3 Explanation: {a,c} covers all 4 nodes

VERBOSE: PolyReduceHalfNodeCoverToQuarterIndpenedentSet() -
quarterIndependentSet_instance = "a,b  a,c b,d"
VERBOSE: VfyQuarterIndependentSet() "b d" not independent set, edge "b,d" exists
test #4 vfyHalfNodeCoverViaVfyQuarterIndepedentSet("a,b  a,c b,d","yes","a c"):
expected "unsure", received "unsure"
   test #4 Explanation: {a,c} does not cover nodes

VERBOSE: PolyReduceHalfNodeCoverToQuarterIndpenedentSet() -
quarterIndependentSet_instance = "a,b a,c a,d b,c c,e"
** test #5 vfyHalfNodeCoverViaVfyQuarterIndepedentSet("a,b a,c a,d b,c c,e","yes"," a b
c"): expected "unsure", received "correct"
test #5 Explanation: {a,b,c} is over half of 5 nodes
```