name: kevin wong
filename: Math115 homework11
date: 04/12/2022
desc: https://courses.csail.mit.edu/6.042/spring18/mcs.pdf (Links to an external site.) please do these problems: 12.6, 12.7, 12.8, 12.10, additional problem

12.6

For each of the following pairs of simple graphs, either define an isomorphism between them, or prove that there is none. (We write ab as shorthand for $(a—b)$.)

- (a)

  G_1 with $V_1 = \{1, 2, 3, 4, 5, 6\}, E_1 = \{12, 23, 34, 14, 15, 35, 45\}$

  G_2 with $V_2 = \{1, 2, 3, 4, 5, 6\}, E_2 = \{12, 23, 34, 45, 51, 24, 25\}$

```
G_1

          _________
         |         |
         |         |
1----2----3----4-----5      6
||_____________|     |
|_____________________|

1 degree is 3
2 degree is 2
3 degree is 3
4 degree is 3
5 degree is 3
6 degree is 0


G_2


      _______________
     | ________      |
     ||        |     |
1----2----3----4----5       6
|                   |
|___________________|

1 degree is 2
2 degree is 4
3 degree is 2
4 degree is 3
5 degree is 3
6 degree is 0
```

The pair of simples graphs above aren't isomorphic because they don't share equal number of vertices with the same degree. ✓

- (b)

  G_3 with $V_3 = \{1, 2, 3, 4, 5, 6\}, E_3 = \{12, 23, 34, 14, 45, 56, 26\}$

G_4 with $V_4 = \{a, b, c, d, e, f\}, E_4 = \{ab, bc, cd, de, ae, ef, cf\}$

```
G_3

     ___________________
    |                   |
1----2----3----4----5----6
|_____________|

1 degree is 2
2 degree is 3
3 degree is 2
4 degree is 3
5 degree is 2
6 degree is 2

G_4

 ___________________
|                   |
a----b----c----d----e----f
          |_____________|

a degree is 2
b degree is 2
c degree is 3
d degree is 2
e degree is 3
f degree is 2

note: G_4 looks like a mirror image of G_3
```

$f_3 : V_{G_3} \to V_{G_4}$ $f = (1, f), (2, e), (3, d), (4, c), (5, b), (6, a)$

Adjacencies matrix for G_3

G_3:	**1**	**2**	**3**	**4**	**5**	**6**
1	0	1	0	1	0	0
2	1	0	1	0	0	1
3	0	1	0	1	0	0
4	1	0	1	0	1	0
5	0	0	0	1	0	1
6	0	1	0	0	1	0

Adjacencies matrix for $f_3(G3)$ (matches G_3)

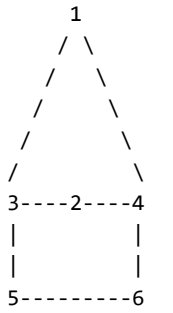
$f_3(G_3)$:	**f**	**e**	**d**	**c**	**b**	**a**
f	0	1	0	1	0	0
e	1	0	1	0	0	1
d	0	1	0	1	0	0
c	1	0	1	0	1	0
b	0	0	0	1	0	1

$f_3(G_3)$:	**f**	**e**	**d**	**c**	**b**	**a**
a	0	1	0	0	1	0

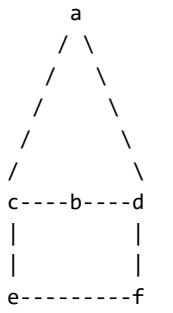
f_3 respects the number of vertices, edges, degree of vertex, adjacency, and cycles of a particular length, so it is an isomorphic pair of graphs. ✓

12.7

List all the isomorphisms between the two graphs given in Figure 12.23. Explain why there are no others.



1 degree is 2
2 degree is 2
3 degree is 3
4 degree is 3
5 degree is 2
6 degree is 2



a degree is 2
b degree is 2
c degree is 3
d degree is 3
e degree is 2
f degree is 2

Figure 12.23

- Adjacency matrix for top graph with the numbered vertices.

	1	**2**	**3**	**4**	**5**	**6**
1	0	0	1	1	0	0

	1	**2**	**3**	**4**	**5**	**6**
2	0	0	1	1	0	0
3	1	1	0	0	1	0
4	1	1	0	0	0	1
5	0	0	1	0	0	1
6	0	0	0	1	1	0

- no swaps ✓

f_1 $f = (1, a), (2, b), (3, c), (4, d), (5, e), (6, f)$

Adjacency matrix for f_1

f_1	**a**	**b**	**c**	**d**	**e**	**f**
a	0	0	1	1	0	0
b	0	0	1	1	0	0
c	1	1	0	0	1	0
d	1	1	0	0	0	1
e	0	0	1	0	0	1
f	0	0	0	1	1	0

- swapped $a \leftrightarrow b$:✓

f_2 $f = (1, b), (2, a), (3, c), (4, d), (5, e), (6, f)$

Adjacency matrix for f_2

f_2	**b**	**a**	**c**	**d**	**e**	**f**
b	0	0	1	1	0	0
a	0	0	1	1	0	0
c	1	1	0	0	1	0
d	1	1	0	0	0	1
e	0	0	1	0	0	1
f	0	0	0	1	1	0

- swapped $c \leftrightarrow d$, $e \leftrightarrow f$:✓

f_3 $f = (1, a), (2, b), (3, d), (4, c), (5, f), (6, e)$

Adjacency matrix for f_3

f_3	**a**	**b**	**d**	**c**	**f**	**e**
a	0	0	1	1	0	0
b	0	0	1	1	0	0
d	1	1	0	0	1	0
c	1	1	0	0	0	1

f_3	a	b	d	c	f	e
f	0	0	1	0	0	1
e	0	0	0	1	1	0

- swapped $a \leftrightarrow b$, $c \leftrightarrow d$, $e \leftrightarrow f$:✓

f_4 $f = (1, b), (2, a), (3, d), (4, c), (5, f), (6, e)$

Adjacency matrix for f_4

f_4	b	a	d	c	f	e
b	0	0	1	1	0	0
a	0	0	1	1	0	0
d	1	1	0	0	1	0
c	1	1	0	0	0	1
f	0	0	1	0	0	1
e	0	0	0	1	1	0

- Below are a few that aren't isomorphisms:

  ~~swapped $a \leftrightarrow b$, $c \leftrightarrow d$:~~

  f_5 $f = (1, b), (2, a), (3, d), (4, c), (5, e), (6, f)$

  ~~swapped $a \leftrightarrow b$, $e \leftrightarrow f$:~~

  f_6 $f = (1, b), (2, a), (3, c), (4, d), (5, f), (6, e)$

  ~~swapped $c \leftrightarrow d$:~~

  f_7 $f = (1, a), (2, b), (3, d), (4, c), (5, e), (6, f)$

  ~~swapped $e \leftrightarrow f$:~~

  f_8 $f = (1, a), (2, b), (3, c), (4, d), (5, f), (6, e)$

- An observation about the two graphs is that they are clearly isomorphic.

The isomorphic mappings appear to be those that mirror one side to the other along an imaginary axis splitting $(1, 2)$ or (a, b). In other words, if a vertex on one side is mapped to the other side, all vertices from that same side have to also be mapped to its equivalent vertex on the other side. For example $f(3, 5) \to (c, e)$ or $f(3, 5) \to (d, f)$ and $f(4, 6) \to (d, f)$ or $f(4, 6) \to (c, e)$. Since (a, b) exists on the "center line", so its isomorphic properties preserved regardless if they are swapped. ✓

12.8

Determine which among the four graphs pictured in Figure 12.24 are isomorphic. For each pair of isomorphic graphs, describe an isomorphism between them. For each pair of graphs that are not isomorphic, give a property that is preserved under isomorphism such that one graph has the property, but the other does not. For at least one of the properties you choose, *prove* that it is indeed preserved under isomorphism (you only need prove one of them).

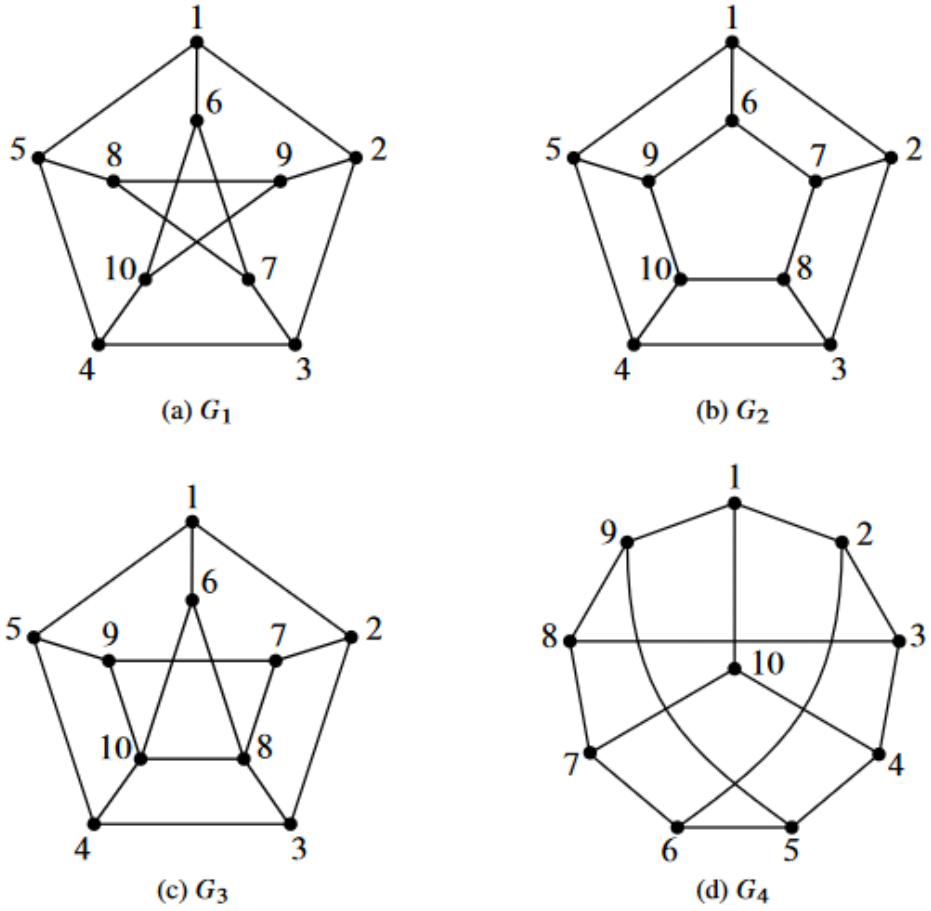


Figure 12.24 Which graphs are isomorphic?

- Adjacency matrices are too tedious for this problem.

	vertex →	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Edges**
G_1	degrees	3	3	3	3	3	3	3	3	3	3	15
G_2	degrees	3	3	3	3	3	3	3	3	3	3	15
G_3	degrees	3	3	3	3	3	3	3	4	3	4	16
G_4	degrees	3	3	3	3	3	3	3	3	3	3	15

- We eliminate G_3 because two of the vertices doesn't share the same degree compared to the other three graphs.
- Having ten vertices is a preserved property amongst G_1, G_2, G_3.
- Having vertices with a degree of three is a preserved property amongst G_1, G_2, G_3.

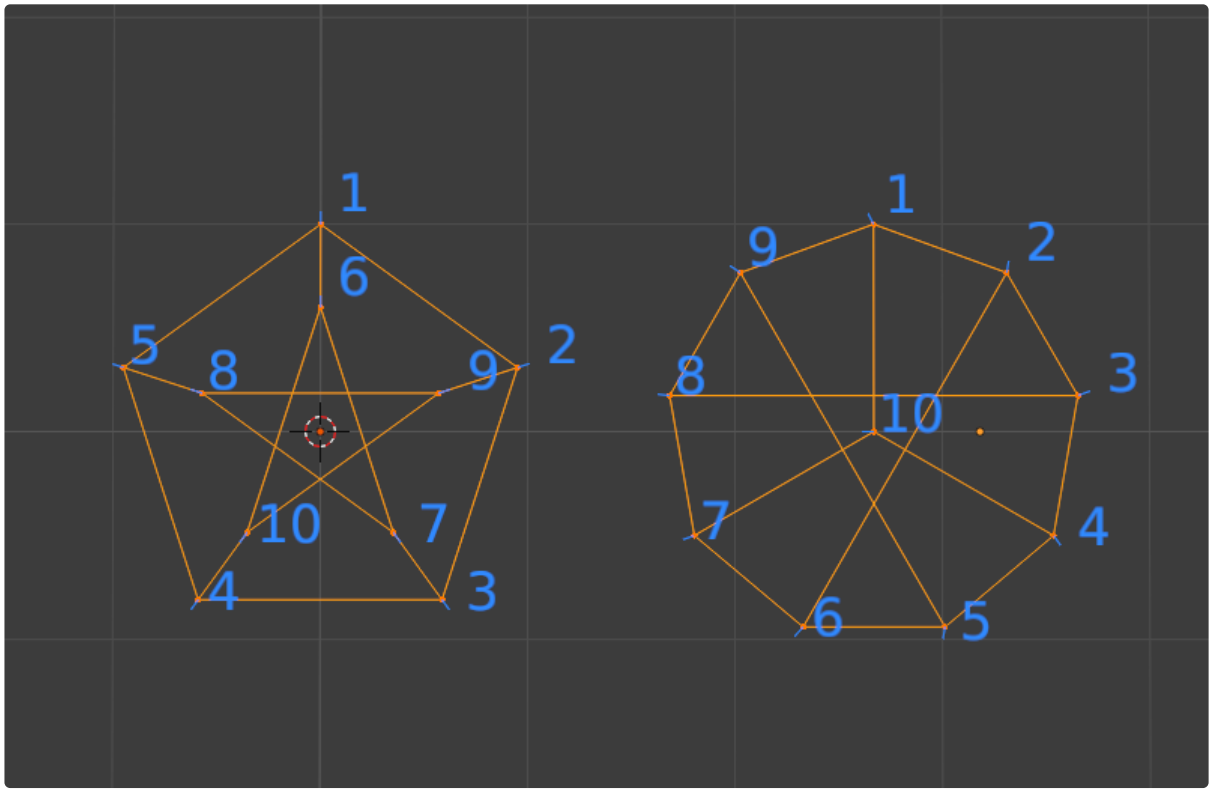
- Having fifteen edges is a preserved property amongst G_1, G_2, G_3, the remaining possible isomorphisms.
- A Hamiltonian path is a preserved property amongst the G_1, G_2, G_3, G_4.

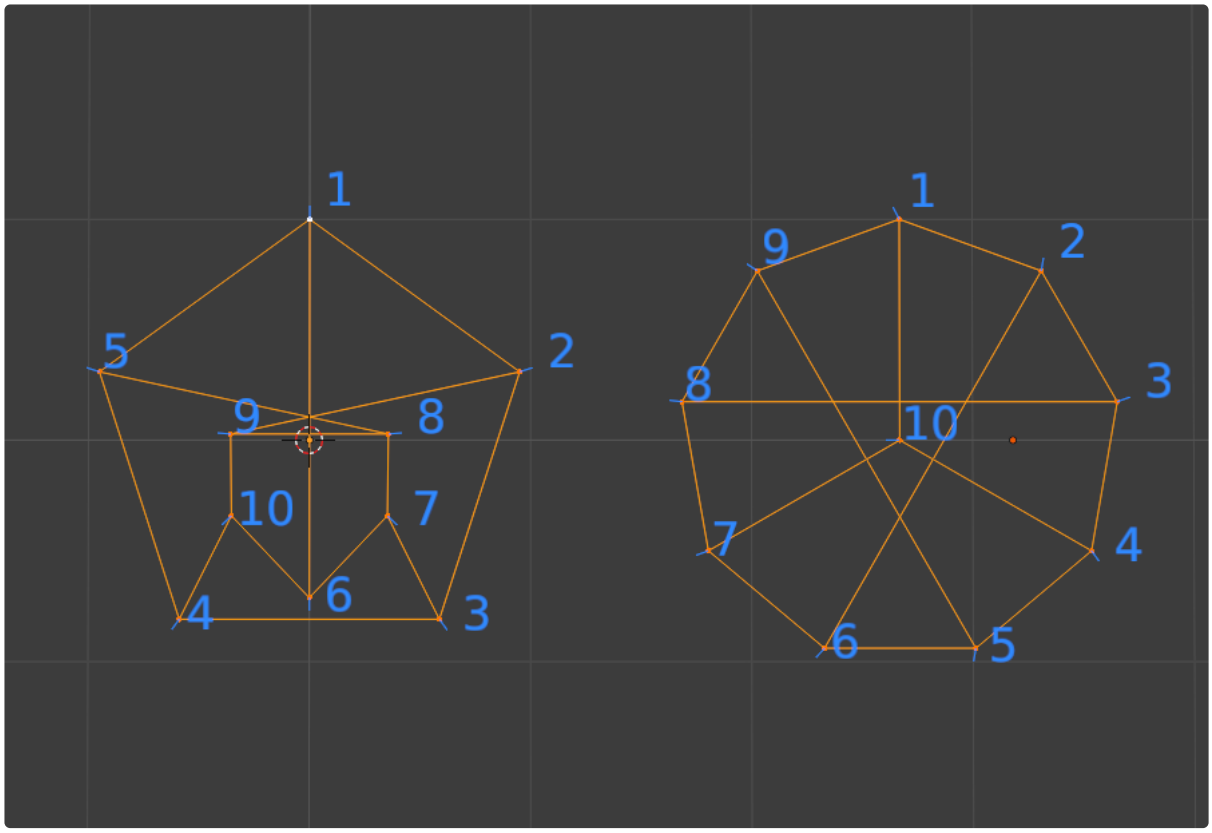
  $G_1 Hamiltonian Path = 1, 2, 3, 4, 5, 8, 7, 6, 10, 9$

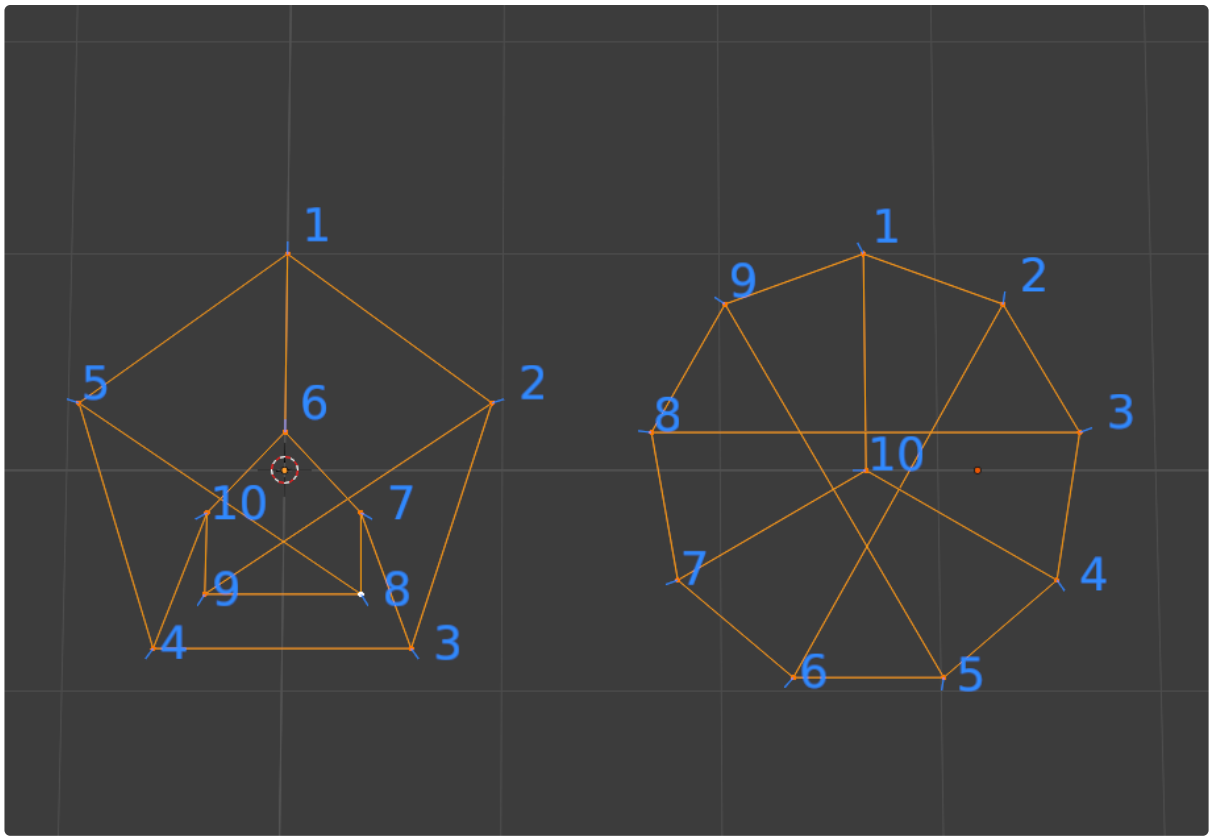
  $G_2 Hamiltonian Path = 1, 2, 3, 4, 5, 9, 10, 8, 7, 6$

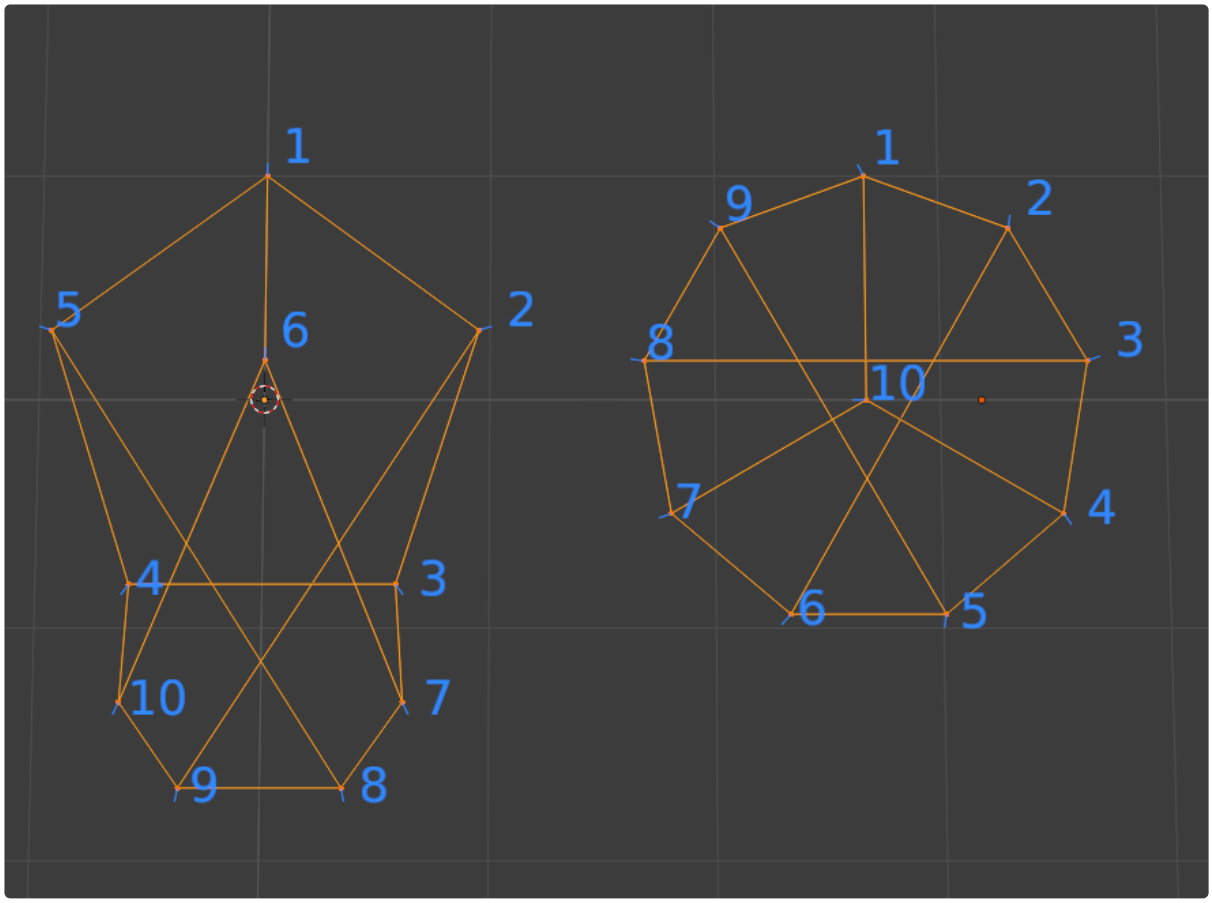
  $G_3 Hamiltonian Path = 1, 2, 3, 4, 5, 9, 10, 6, 8, 7$

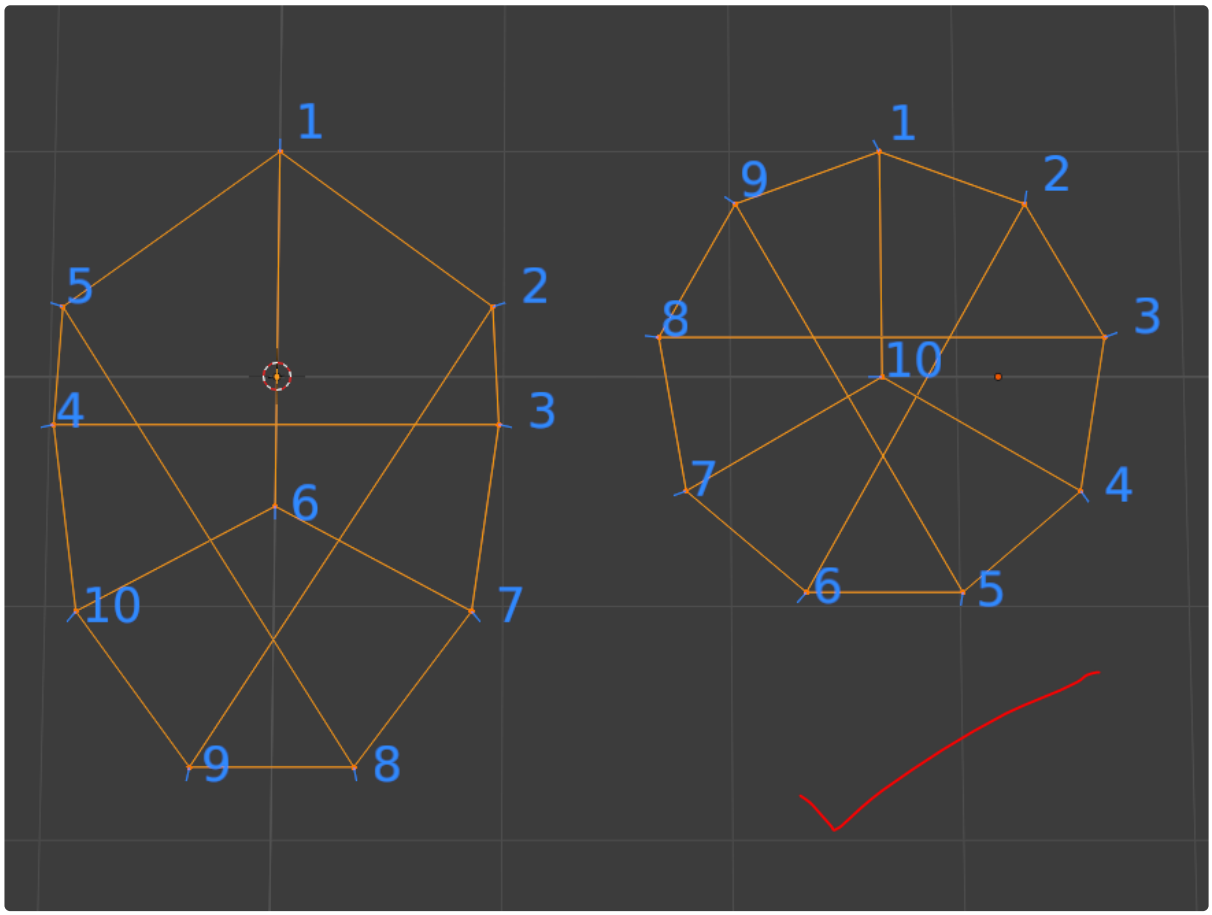
  $G_4 Hamiltonian Path = 1, 2, 6, 5, 9, 8, 3, 4, 10, 7$
- A cycles argument appears to be pretty tedious as well.
- Conclusion:
  - I believe G_1 and G_4 is an isomorphic pair, the work is shown below. ✓
  - I don't know for sure, but I believe G_2 can't form an isomorphic pair with G_1 and G_4 because of the same reason we couldn't connect the utilities in the house in the lectures notes. It has something to do with crossing edges.
  - THe work: The G_1 and G_4 pair was found by untying G_1 into G_4




1
5
2
9
8
10
7
6
4
3
1
9
2
8
3
10
7
4
6
5


1
5
6
2
10
7
9
8
4
3
1
9
2
8
3
10
7
4
6
5


1
2
3
4
5
6
7
8
9
10
1
2
3
4
5
6
7
8
9
10


1
2
3
4
5
6
7
8
9
10
1
2
3
4
5
6
7
8
9
10

$G_4 = G_4 = G_4$	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1		1							1	1
2	1		1			1				
3		1		1				1		
4			1		1					1
5				1		1			1	
6		1			1		1			
7						1		1		1
8			1				1		1	
9	1				1			1		
10	1			1			1			

$f(G_4) \to G_1$	**1**	**2**	**3**	**7**	**8**	**9**	**10**	**4**	**5**	**6**
1		1							1	1
2	1		1			1				
3		1		1				1		
7			1		1					1
8				1		1			1	
9		1			1		1			
10						1		1		1
4			1				1		1	
5	1				1			1		
6	1			1			1			

12.10

Problem 12.10.
Let's say that a graph has "two ends" if it has exactly two vertices of degree 1 and all its other vertices have degree 2. For example, here is one such graph:

(a) A *line graph* is a graph whose vertices can be listed in a sequence with edges between consecutive vertices only. So the two-ended graph above is also a line graph of length 4.

Prove that the following theorem is false by drawing a counterexample.
False Theorem. *Every two-ended graph is a line graph.*

(b) Point out the first erroneous statement in the following bogus proof of the false theorem and describe the error.

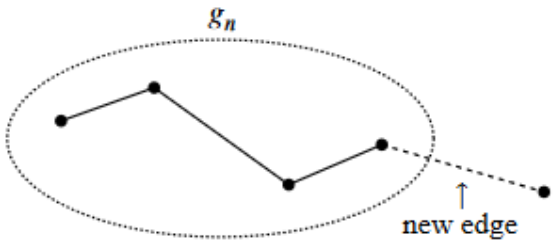
Bogus proof. We use induction. The induction hypothesis is that every two-ended graph with n edges is a line graph.

Base case ($n = 1$)**:** The only two-ended graph with a single edge consists of two vertices joined by an edge:

Sure enough, this is a line graph.

Inductive case: We assume that the induction hypothesis holds for some $n \geq 1$ and prove that it holds for $n + 1$. Let G_n be any two-ended graph with n edges. By the induction assumption, G_n is a line graph. Now suppose that we create a two-ended graph G_{n+1} by adding one more edge to G_n. This can be done in only one way: the new edge must join one of the two endpoints of G_n to a new vertex;

otherwise, G_{n+1} would not be two-ended.



Clearly, G_{n+1} is also a line graph. Therefore, the induction hypothesis holds for all graphs with $n + 1$ edges, which completes the proof by induction.

■

- a)

```
2---------2         1
|         |         |
|         |         |
|         |         |
2---------2         1
```

A graph with 6 vertices. The numbers represent the degree for each vertex.

- b) The incorrect proof comes in during the base case when "**Base case** $(n = 1)$: The only two-ended graph with a single edge consists of two vertices joined by an edge:"
  - a counter example would be:

```
1
|            3
|
|
2
```

A two-ended graph with vertices 1, 2, and 3. An edge joins vertices 1 and 2.

Vertices 1, 2 form a two-ended graph with a single edge, but there are 3 points. ✓

RSA encryption problem

You have intercepted some messages sent on a popular dating site. The person sending these messages has the address $(n_1, e_1) = (96403, 31)$. The person receiving these messages has the address $(n_2, e_2) = (405319, 29)$.

Message1:

[227631, 175041, 262106, 49893, 67508, 265405, 112056, 15517, 346109, 269901, 363337, 112056, 80936, 175102, 266472]

Message2:

[227631, 175041, 269901, 131833, 298318, 304995, 334924, 264584, 107159, 173412, 296042, 222009, 293529]

Message3:

[227631, 175041, 238656, 269901, 88066, 173412, 202240, 191529, 102893, 15585, 175448, 67508, 265405, 222009, 136179, 259370, 139802]

You've discovered that the modulus in the receiver's public key, $n = 405319$, is the product of the primes 409 and 991. Use this information to break the RSA encryption. Pick one of the above questions (or more if you want), decrypt it, and send a reply to the sender.

- Going to decrypt message 3:

  Modulus recievers public key, $n = 405319$.

  Product of primes 409 and 991.

  $(n, e) = (409, 991)$

  To decrypt this, we need to computer the inverse of "$e^{-1}\ mod (p-1)(q-1)$"

  $(p-1)(q-1) = 408 * 990 = 403920$

- extended Euclidean algorithm:

  $a = r_{-1}, b = r_0$

  $a * s_i + b * t_i = r_i$, then iterate until r_i equals to zero.

  i is the index

  q_i is the quotient

  r_i is the remainder

  s_i, t_i are the Bezout coefficients

i	q_i	r_i	s_i	t_i
-1	-	403920	1	0
0	-	29	0	1
1	403920 // 29 = 13928	8	1 - (13928)(0) = 1	0 - (13928)(1) = -13928
2	29 // 8 = 3	29 % 8 = 5	0 - (3)(1) = -3	1 - (3)(-13928) = 41785
3	8 // 5 = 1	8 % 5 = 3	1 - (1)(-3) = 4	-13928 - (1)(41785) = -55713
4	5 // 3 = 1	5 % 3 = 2	-3 - (1)(4) = -7	41785 - (1)(-55713) = 97498
5	3 // 2 = 1	3 % 2 = 1	4 - (1)(-7) = 11	-55713 - (1)(97498) = -153211
6	2 // 1 = 2	0	-	-

$gcd = (11)403920 + (-153211)29$

An inverse of $29\ mod 403920$ is -153211, which is congruent to $250709\ mod 403920$

So the description exponent, $e^{-1} = 250709$

Then take the $MessageElement^{e^{-1}}\ mod\ n$ for each element in the message.

- Finished the rest via python below:

```
# reciever's private key: (n, e^-1 = e)
n = 405319
e = 250709

message_encrypted = [227631, 175041, 238656, 269901, 88066, 173412, 202240, 191529, 102893,
15585, 175448, 67508, 265405, 222009, 136179, 259370, 139802]

message_decrypted = []

# decryption loop
for i in range(len(message_encrypted)):
    message_decrypted.append(str(message_encrypted[i]**e % n))

message_decrypted_wZero = []

# add zeros loop
for message in message_decrypted:
    while len(message) < 4:
        message = "0" + message

    message_decrypted_wZero.append(message)

# print decrypted message with zeros
print("message_decrypted_wZero:", message_decrypted_wZero)

cipher = {
"00":"A", "01":"B", "02":"C", "03":"D", "04":"E", "05":"F", "06":"G", "07":"H", "08":"I",
"09":"J", "10":"K", "11":"L", "12":"M", "13":"N", "14":"O", "15":"P", "16":"Q", "17":"R",
"18":"S", "19":"T", "20":"U", "21":"V", "22":"W", "23":"X", "24":"Y", "25":"Z", "26":"-"}

decrypted_message = []

# mapping decrypted message to a letter via cipher
for message in message_decrypted_wZero:
    first_key = message[0] + message[0+1]
    second_key = message[0+2] + message[0+3]

    if first_key in cipher:
        decrypted_message.append(cipher[first_key])

    if second_key in cipher:
        decrypted_message.append(cipher[second_key])

message = ""

# print string
for letters in decrypted_message:
    message += letters

print("message3:", message)

############### OUTPUT ################
# message_decrypted_wZero: ['2207', '0019', '2608', '1826', '1907', '0426', '0104', '1819',
'2612', '1421', '0804', '2624', '1420', '2104', '2618', '0404', '1326']
# message3: WHAT-IS-THE-BEST-MOVIE-YOUVE-SEEN-
```

- Reply to senders address $(n_1, e_1) = (96403, 31)$:

```
original text converted (not by compiler)
G O  O D  - W  I L  L -  H U  N T  I N  G -
0614 1403 2622 0811 1126 0720 1319 0813 0626

# sender's public encryption key: (n, e)
n = 96403
e = 31

message = "GOOD-WILL-HUNTING-"

cipher = {
"00":"A", "01":"B", "02":"C", "03":"D", "04":"E", "05":"F", "06":"G", "07":"H", "08":"I",
"09":"J", "10":"K", "11":"L", "12":"M", "13":"N", "14":"O", "15":"P", "16":"Q", "17":"R",
"18":"S", "19":"T", "20":"U", "21":"V", "22":"W", "23":"X", "24":"Y", "25":"Z", "26":"-"}

reverse_cipher = {}

# reversing cipher
for key in cipher:
    reverse_cipher[cipher[key]] = key

encrypted_message_string = ""

# mapping letters to numbers via reverse_cipher
    # note: only works for this particular message
for letter in message:
    if letter in reverse_cipher:
        encrypted_message_string += reverse_cipher[letter]

encrypted_message = []

# converting numbers to four number groupings
for i in range(0, len(encrypted_message_string), 4):
    four_number_set = encrypted_message_string[i] + encrypted_message_string[i + 1] +
encrypted_message_string[i + 2] + encrypted_message_string[i + 3]

    encrypted_message.append(int(four_number_set))

print("encrypted_message", encrypted_message)

encrypted_message_to_sender = []

# encrypt message using sender's public key
for number in encrypted_message:
    encrypted_message_to_sender.append(number**e % n)

print("encrypted_message_to_sender", encrypted_message_to_sender)

############### OUTPUT ################
# encrypted_message [614, 1403, 2622, 811, 1126, 720, 1319, 813, 626]
# encrypted_message_to_sender [40918, 48263, 70873, 39011, 83765, 51088, 28246, 1906, 57941]
```

-The encrypted message to the sender is [40918, 48263, 70873, 39011, 83765, 51088, 28246, 1906, 57941]

END
