1. (4 points)

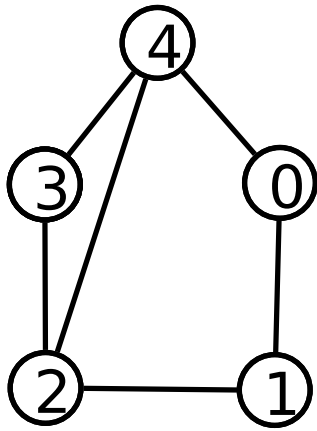   (a) Draw a minimum spanning tree of the graph below.
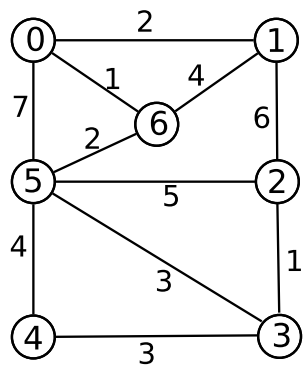
   (b) What is the weight of the minimum spanning tree?

   (c) Is this graph Eulerian? Explain.

2. (5 points)
   Represent the following graph by an adjacency matrix.

3. (3 points)
   Circle the correct Python evaluation (True or False) for each of the following Boolean statements:

   (a) $2 ** 2 ** 3 == 4 ** 3$
      - A. True
      - B. False

   (b) True and not False
      - A. True
      - B. False

   (c) 4 not in range(4)
      - A. True
      - B. False

   (d) "hello" == ['h', 'e', 'l', 'l', 'o']
      - A. True
      - B. False

   (e) $10/3 == 3$
      - A. True
      - B. False

   (f) len([1,2,3])==2
      - A. True
      - B. False

4. (8 points)
   For each of the following fragments of Python code, write what would be printed by the program.

   (a)
   ```
   total=0
   for i in range(3,7,2):
       total=total+i
   print(str(total))
   ```

   (b)
   ```
   s="hello world"
   s.upper()
   print(s)
   ```

   (c)
   ```
   for k in [1, 2, 3]:
       print(k*str(k))
   ```

   (d)
   ```
   s = "1045"
   t = "FIT"
   for k in range(len(t)):
       s += t[k]
   print(s)
   ```

5. (8 points)

   (a) Give a definition of an algorithm.

   (b) Explain why the following pseudocode is, or is not, an algorithm.
   ```
   input x
   while True:
       x=x+1
   ```

   (c) Explain why the following words are, or are not, valid Python variable names.
   ```
   break
   min_val
   ```

6. (4 points)

Write a Python program that given an unsorted list of integers and prints the number in the list with the smallest absolute value.

For example, if the list is $[3, -5, 2, -6, -1]$ the program should print -1,

and if the list is $[4, -21, 2, -20]$ your program should print 2.

You can assume that the list is called numList and already contains at least one number: you do not need to write code to read the numbers from a file or ask the user to input them.

7. (5 points)

(a) Sort the list $[3, 16, 25, 32, 1, 20]$ into increasing order using **Insertion sort**. You should write the list in the table below after each iteration of the main loop in this algorithm.

3	16	25	32	1	20

(b) Give a loop invariant for the main loop in the Insertion Sort Algorithm.

8. (5 points)

   Write Python code to find the maximum entry in a $n \times m$ table T. Your code should print the value of the maximum entry and its location (row and column). You can assume that all the entries are distinct.

This is the end of the test.