# Problem 7

Bookmark this page

## Problem 7

0.0/20.0 points (graded)

You are given the following two classes.

```
### Do not change the Location or Campus classes. ###
### Location class is the same as in lecture.     ###
class Location(object):
    def __init__(self, x, y):
        self.x = x
        self.y = y
    def move(self, deltaX, deltaY):
        return Location(self.x + deltaX, self.y + deltaY)
    def getX(self):
        return self.x
    def getY(self):
        return self.y
    def dist_from(self, other):
        xDist = self.x - other.x
        yDist = self.y - other.y
        return (xDist**2 + yDist**2)**0.5
    def __eq__(self, other):
        return (self.x == other.x and self.y == other.y)
    def __str__(self):
        return '<' + str(self.x) + ',' + str(self.y) + '>'

class Campus(object):
    def __init__(self, center_loc):
        self.center_loc = center_loc
    def __str__(self):
        return str(self.center_loc)
```

Implement a class that meets the specifications below.

```
class MITCampus(Campus):
    """ A MITCampus is a Campus that contains tents """
    def __init__(self, center_loc, tent_loc = Location(0,0)):
        """ Assumes center_loc and tent_loc are Location objects
        Initializes a new Campus centered at location center_loc
        with a tent at location tent_loc """
```

```
        # Your code here

    def add_tent(self, new_tent_loc):
        """ Assumes new_tent_loc is a Location
        Adds new_tent_loc to the campus only if the tent is at least 0.5 distance
        away from all other tents already there. Campus is unchanged otherwise.
        Returns True if it could add the tent, False otherwise. """
        # Your code here

    def remove_tent(self, tent_loc):
        """ Assumes tent_loc is a Location
        Removes tent_loc from the campus.
        Raises a ValueError if there is not a tent at tent_loc.
        Does not return anything """

        # Your code here

    def get_tents(self):
        """ Returns a list of all tents on the campus. The list should contain
        the string representation of the Location of a tent. The list should
        be sorted by the x coordinate of the location. """
        # Your code here
```

For example, if `c = MITCampus(Location(1,2))` then executing the following sequence of commands:

`c.add_tent(Location(2,3))` should return `True`
`c.add_tent(Location(1,2))` should return `True`
`c.add_tent(Location(0,0))` should return `False`
`c.add_tent(Location(2,3))` should return `False`
`c.get_tents()` should return `['<0,0>', '<1,2>', '<2,3>']`

Paste your entire class `MITCampus` in the box below. Do not leave any debugging print statements.

```
1 # Paste your class here
```

Press ESC then TAB or click outside of the code editor to exit

Unanswered

**Note:There is no error with "test 5". The sorted list you see is because we sort it for you with** `print(sorted(c.get_tents()))`