"""Problem 8 COMPLETE
    If you wanted to find one “best” answer but had run MDS 10 times, how would you pick
    the best? Why? Show a plot of the best and any code you used to find it.

    ANSWER: The 'best' answer would be the MDS array that achieved the lowest stress. There is
    accurate way to choose the best MDS simply by looking at the 10 subplots. To find the best,
    I changed my traceGradient algorigthm (actually it's the first one I used as it's more accurate
    and I didn't realize the assignment wanted a less accurate version) to do the following:

    1) Create a new random array of x,y points.
    2) Adjust the points using the gradient until a threshold near 0 is reached.
    3) Check the stress for the point array and save the array if it's stress score is less than
        the last best point array.
    4) Repeat 1-4 n iterations.
    5) Return the best array and plot.

    This method will help ensure that a global minimum is found because we increase the chances
    that we get a random point array that is near the global minimum which should result in the
    lowest possible stress.

    I have included a plot using this method which shows a better clustering of sports than
    doing N iterations on one position array. """

def traceGradientOptimal(psycho_array, learn_rate=.01, gradient_threshold=0.0005, n=1000, dimensions=2):
    """Takes a psychological distance array, returns a position array with min(stress) after N iterations,
        it's stress value, and the total stress for each position array over N iterations"""

    grad_x, grad_y = 10000, 10000
    grad_total = 10000
    min_stress = float('inf')
    stress_value = float('inf')
    best_positions = None
    stressList = []
    bestIter = 0

    for i in range(n):

```
        x, y = 0, 1

        position_array = getRandPositions(psycho_array.shape[0], dimensions)
        while (grad_x > gradient_threshold) and (grad_y > gradient_threshold):
            for point in range(0, len(position_array)):
                grad_x, grad_y = gradient(point, psycho_array, position_array, h=.001)
                position_array[point][x] += (-grad_x * learn_rate)
                position_array[point][y] += (-grad_y * learn_rate)

        grad_x, grad_y = 10000, 10000

        stress_value = stress(psycho_array, position_array)
        stressList += [stress_value]

        if stress_value < min_stress:
            min_stress = stress_value
            best_positions = position_array
            bestIter = i

    return best_positions, min_stress, stressList, bestIter
"""END PROBLEM 8"""
#end
```