# Problem 1 – Best Time to Buy and Sell Stock

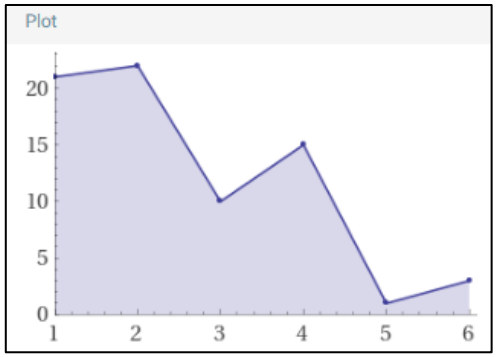
**Description:**

You are given a list called **prices** where **prices[i]** is the price of a given stock on the **i+1th** day i.e. price[0] represents price on day 1 (remember: list indices in python start with 0).

You want to maximise your profit by choosing a **single day** to buy one stock and choosing a **different day in the future** to sell that stock.

**Your task:**

Write a **Python** function that returns **the maximum profit you can achieve from this transaction. If you cannot achieve any profit, return 0.**




## My solution:

```python
def bestTimeToBuyAndSellStock(myList):
    maxDiff = 0
    minItem = myList[0]

    for i in range(len(myList)):
        if myList[i] < minItem:
            minItem = myList[i]

        maxDiff = max(maxDiff, myList[i] - minItem)

    return maxDiff

```

## Unit tests:

```python
import unittest
from BestTimeToBuyAndSellStock_final import bestTimeToBuyAndSellStock

test_cases = [
    ([22,1,5], 4),
    ([7,1,5,3,6,4], 5),
    ([7,6,4,3,1], 0),
    ([5,10,1,3,2,1,2], 5),
    ([21,22,1,5,7,2], 6),
    ([21,22,10,15,1,3], 5)
]

class TestValidClockTimes(unittest.TestCase):
    def test_all(self):
        for stock, expected in test_cases:
            with self.subTest(case=stock, expected=expected):
                result = bestTimeToBuyAndSellStock(stock)
                self.assertEqual(result, expected)

if __name__ == '__main__':
    unittest.main()

```