In [1]:
```python
'Week 4 Practice Programming Assignment'

def orangecap(data):
    'Finding batsman with highest total score in all matches'
    scores = {}
    for match in data:
        for player in data[match]:
            if player not in scores:
                scores[player] =  data[match][player]
            else:
                scores[player] += data[match][player]
    top_player = max(scores,key = lambda x: scores[x])
    return (top_player,scores[top_player])

def addpoly(p1,p2):
    'Adding polynomial given in (coeff,degree) tuple list form'
    degree_coeff = {}
    for c,d in p1:
        degree_coeff[d] = c
    for c,d in p2:
        if d in degree_coeff:
            degree_coeff[d] += c
        else:
            degree_coeff[d] = c
    return [ (degree_coeff[d],d) for d in sorted(degree_coeff,reverse=True) if degree_coeff[d] ]

def multpoly(p1,p2):
    'Multiplying polynomial given in (coeff,degree) tuple list form'
    degree_coeff = {}
    for c1,d1 in p1:
        for c2,d2 in p2:
            if d1+d2 in degree_coeff:
                degree_coeff[d1+d2] += c1*c2
            else:
                degree_coeff[d1+d2] = c1*c2
    return [ (degree_coeff[d],d) for d in sorted(degree_coeff,reverse=True) if degree_coeff[d] ]
```

In [2]:
```
_ = '''DO NOT COPY THIS CODE'''

a = '''Test Case 1 orangecap({'match1':{'player1':57,'player2':38},'match2':{'player3':9,'player1':42},'match3':{
Test Case 2 orangecap({'test1':{'Ashwin':84,'Kohli':120},'test2':{'ashwin':59,'Pujara':42}}) ('Kohli',120)
Test Case 3 orangecap({'match1':{'player1':57,'player2':38},'match2':{'player3':9,'player1':42},'match3':{'player
Test Case 4 orangecap({'match1':{'player1':57,'player2':38}}) ('player1',57)
Test Case 5 multpoly([(1,1),(-1,0)],[(1,2),(1,1),(1,0)]) [(1,3),(-1,0)]
Test Case 6 multpoly([(2,1)],[(-2,1)]) [(-4,2)]
Test Case 7 multpoly([(4,3),(3,0)],[(-4,3),(2,1)]) [(-16,6),(8,4),(-12,3),(6,1)]
Test Case 8 addpoly([(4,3),(3,0)],[(-4,3),(2,1)]) [(2,1),(3,0)]
Test Case 9 addpoly([(2,1)],[(-2,1)]) []
Test Case 10 addpoly([(1,1),(-1,0)],[(1,2),(1,1),(1,0)]) [(1,2),(2,1)]'''


for i1,i2,i3,j,k in [x.split(' ') for x in a.split('\n')]:
    print(eval(j),eval(k))

_ = '''NON-COPYING REGION END'''
```

```
('player3', 100) ('player3', 100)
('Kohli', 120) ('Kohli', 120)
('Kohli', 120) ('Kohli', 120)
('player1', 57) ('player1', 57)
[(1, 3), (-1, 0)] [(1, 3), (-1, 0)]
[(-4, 2)] [(-4, 2)]
[(-16, 6), (8, 4), (-12, 3), (6, 1)] [(-16, 6), (8, 4), (-12, 3), (6, 1)]
[(2, 1), (3, 0)] [(2, 1), (3, 0)]
[] []
[(1, 2), (2, 1)] [(1, 2), (2, 1)]
```

In [ ]:

In [3]:
```
'Week 4 Programming Assignment'

def rainaverage(L):
    'Average rain per city'
    city_raincount = {}
    for city,amount in L:
        if city in city_raincount:
            city_raincount[city][0] += amount
            city_raincount[city][1] += 1
        else:
            city_raincount[city] = [amount,1]
    city_order = sorted(city_raincount)
    return [(city,float(city_raincount[city][0]/city_raincount[city][1])) for city in city_order]

def flatten(D):
    'Recursive, Unrolling only list data type'
    ret = []
    for i in D:
        if (type(i) == type([])):
            ret.extend( flatten(i) )
        else:
            ret.append(i)
    return ret
```

```
In [4]: _ = '''DO NOT COPY THIS CODE'''

        a = '''Test Case 1 rainaverage([(1,2),(1,3),(2,3),(1,1),(3,8)]) [(1,2.0),(2,3.0),(3,8.0)]
        Test Case 2 rainaverage([('Bombay',848),('Madras',103),('Bombay',923),('Bangalore',201),('Madras',128)]) [('Banga
        Test Case 3 flatten([1,2,[3],[4,[5,6]]]) [1,2,3,4,5,6]
        Test Case 4 flatten([1,2,3,(4,5,6)]) [1,2,3,(4,5,6)]
        Test Case 5 flatten(["hello",True,3]) ['hello',True,3]
        Test Case 6 flatten([1,2,[3,["hello",True]],[4,[5,6]]]) [1,2,3,'hello',True,4,5,6]'''


        for i1,i2,i3,j,k in [x.split(' ') for x in a.split('\n')]:
            print(eval(j),eval(k))

        _ = '''NON-COPYING REGION END'''
```

```
[(1, 2.0), (2, 3.0), (3, 8.0)] [(1, 2.0), (2, 3.0), (3, 8.0)]
[('Bangalore', 201.0), ('Bombay', 885.5), ('Madras', 115.5)] [('Bangalore', 201.0), ('Bombay', 885.5), ('Madra
s', 115.5)]
[1, 2, 3, 4, 5, 6] [1, 2, 3, 4, 5, 6]
[1, 2, 3, (4, 5, 6)] [1, 2, 3, (4, 5, 6)]
['hello', True, 3] ['hello', True, 3]
[1, 2, 3, 'hello', True, 4, 5, 6] [1, 2, 3, 'hello', True, 4, 5, 6]
```

```
In [ ]:
```