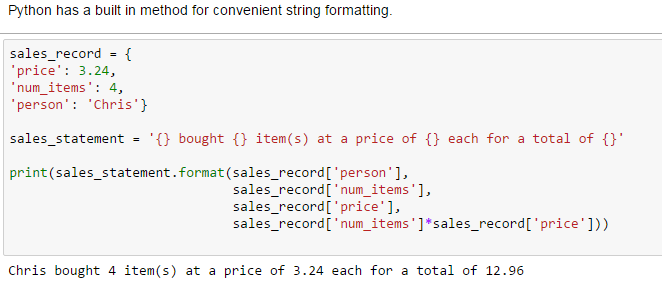
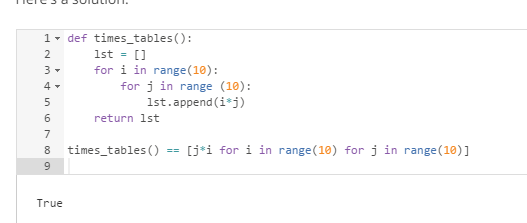
# Grammar

## String formatting

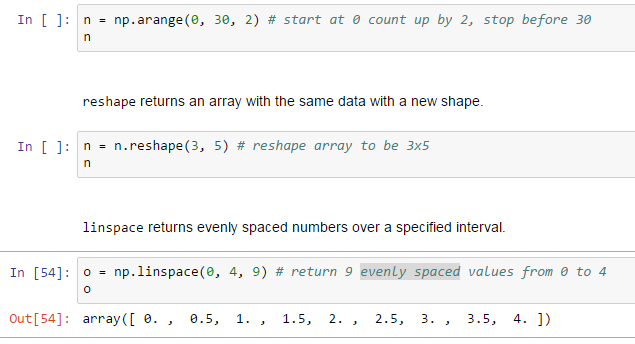


## Convert into list

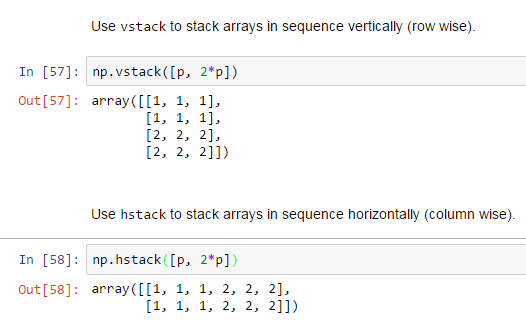


## Numpy basic functions

Generate arraies:



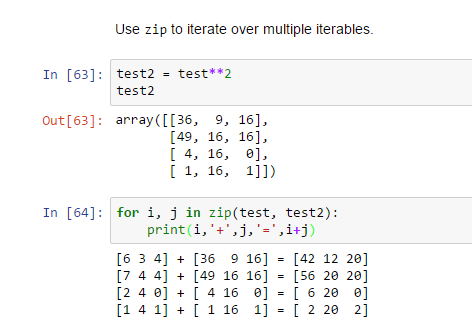
Mat stack functions:



Important array feature:

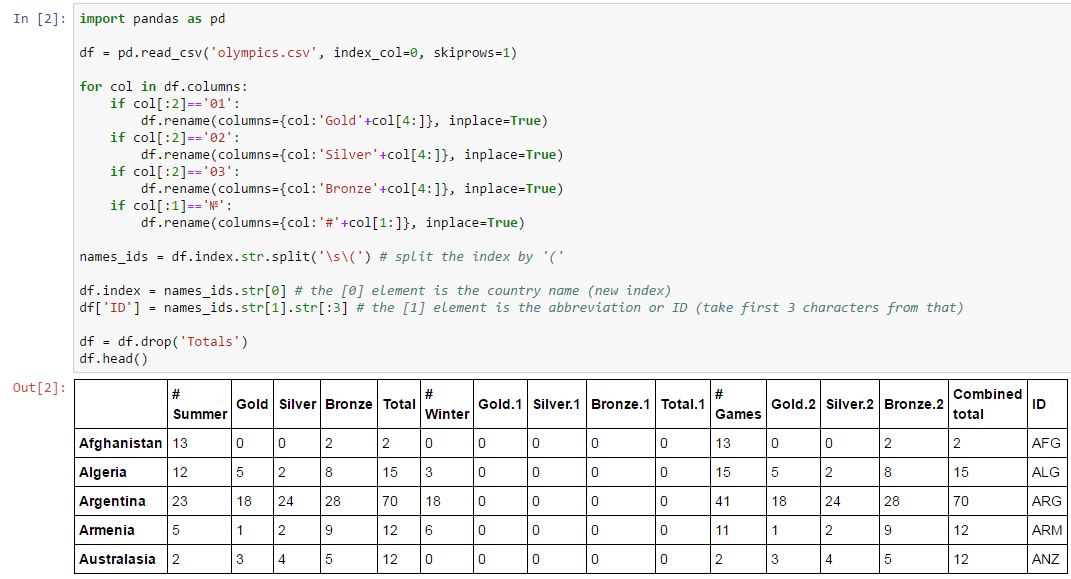


Zip to iterate multiple iterables:

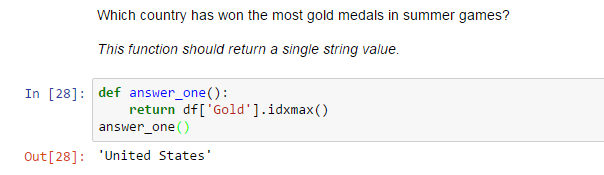


## Manipulate Series and Dataframe

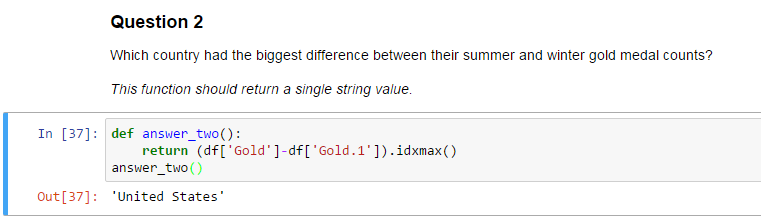
Data:



Select max id:



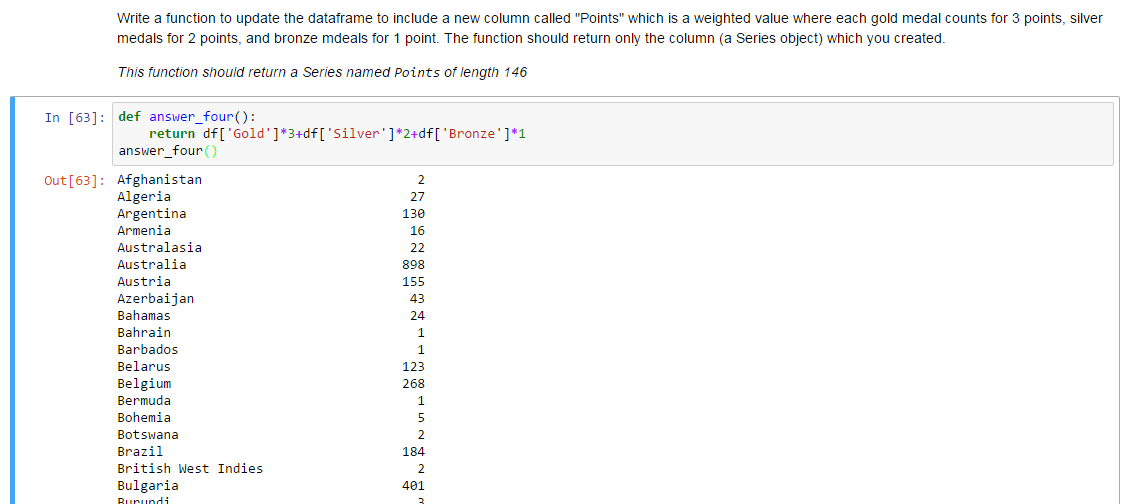
Max difference:



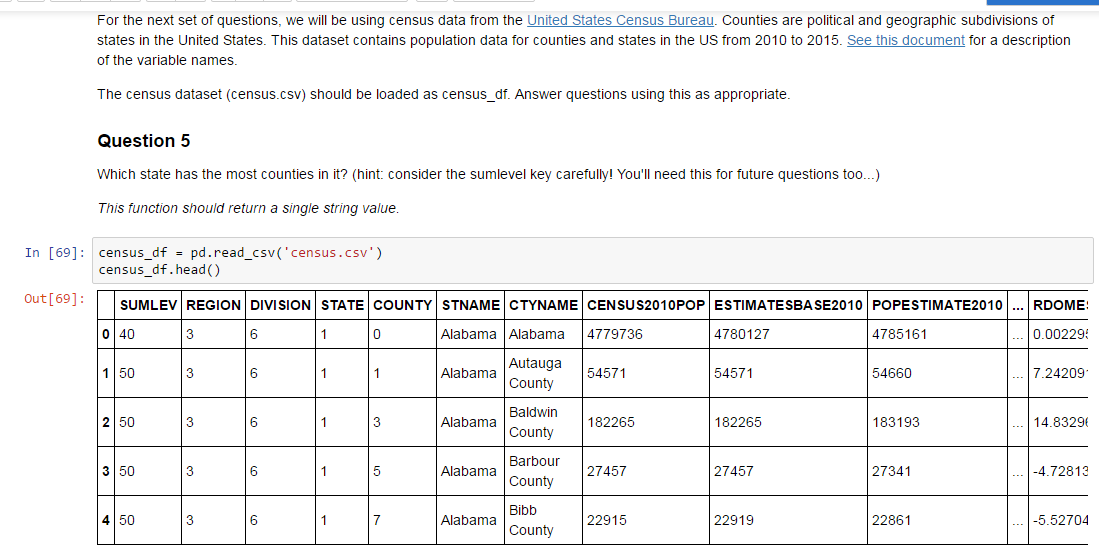
Max difference abs percent:



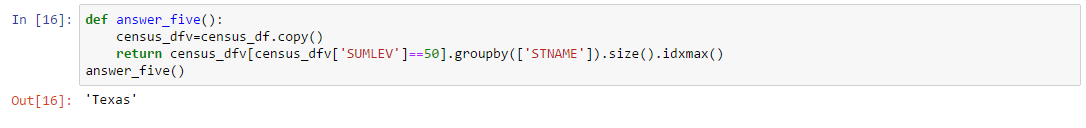
Weighted value:



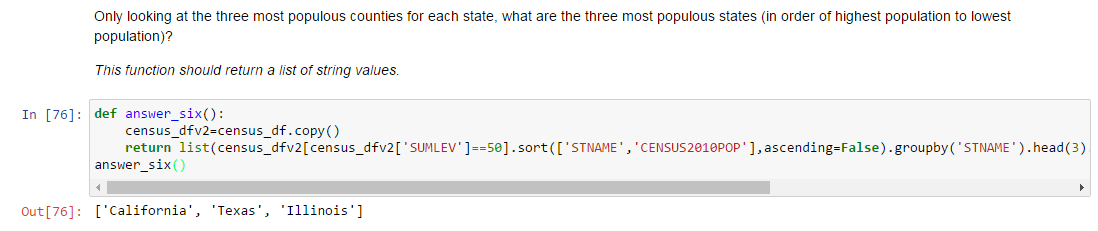
Data2:



Groupby Count Max:



Top 3 most populous states of top 3 counties:



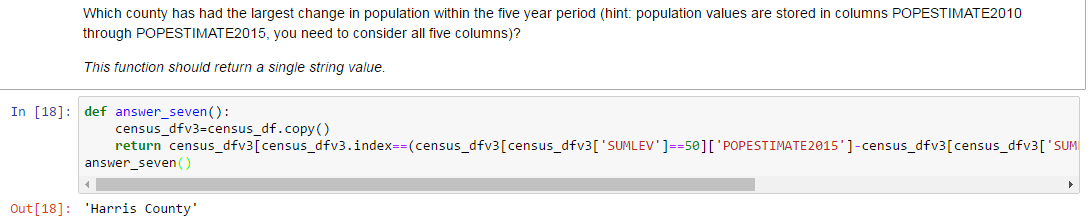
def answer\_six():

census\_dfv2=census\_df.copy()

return list(census\_dfv2[census\_dfv2['SUMLEV']==50].sort(['STNAME','CENSUS2010POP'],ascending=False).groupby('STNAME').head(3).groupby('STNAME')['CENSUS2010POP'].sum().nlargest(3).index)

answer\_six()

Largest difference in population:



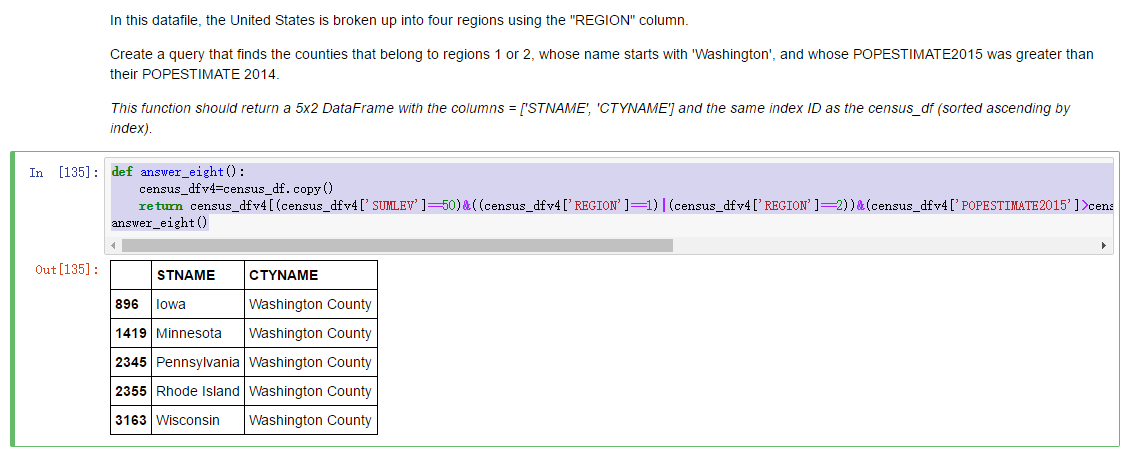
def answer\_seven():

census\_dfv3=census\_df.copy()

return census\_dfv3[census\_dfv3.index==(census\_dfv3[census\_dfv3['SUMLEV']==50]['POPESTIMATE2015']-census\_dfv3[census\_dfv3['SUMLEV']==50]['POPESTIMATE2010']).idxmax()]['CTYNAME'].values[0]

answer\_seven()

Multiply condition query:



def answer\_eight():

census\_dfv4=census\_df.copy()

return census\_dfv4[(census\_dfv4['SUMLEV']==50)&((census\_dfv4['REGION']==1)|(census\_dfv4['REGION']==2))&(census\_dfv4['POPESTIMATE2015']>census\_dfv4['POPESTIMATE2014'])&(census\_dfv4['CTYNAME'].str.startswith('Washington'))].loc[:,['STNAME','CTYNAME']]

answer\_eight()

Apply function:

