In [1]:

```
# Profitable app profiles for the App Store and Google play store

# For this project, we'll pretend we're working as data analysts for a
# company that builds Android and iOS mobile apps. We make our apps
# available on Google Play and the App Store.
# Our aim in this project is to find mobile app profiles that are
# profitable for the App Store and Google Play markets

# At our company, we only build apps that are free to download and install,
# and our main source of revenue consists of in-app ads. This means that
# our revenue for any given app is mostly influenced by the number of users
# that use our app. Our goal for this project is to analyze data to help
# our developers understand what kinds of apps are likely to attract more
# users.
```

In [2]:

```
# As of September 2018, there were approximately 2 million iOS apps
# available on the App Store, and 2.1 million Android apps on Google Play.

# Collecting data for over four million apps requires a significant amount
# of time and money, so we'll try to analyze a sample of data instead.
# To avoid spending resources with collecting new data ourselves, we should
# first try to see whether we can find any relevant existing data at no cost.
# Luckily, these are two data sets that seem suitable for our purpose:

#  - A data set containing data about approximately ten thousand
#      Android apps from Google Play
#  - A data set containing data about approximately seven thousand iOS apps
#      from the App Store

# Let's start by opening the two data sets and then continue with exploring
# the data

from csv import reader

# Data from Google play store
opened_file = open('googleplaystore.csv')
read_file = reader(opened_file)
android = list(read_file)
android_header = android[0]
android = android[1:]

# Data from the App store
opened_file = open('AppleStore.csv')
read_file = reader(opened_file)
ios = list(read_file)
ios_header = ios[0]
ios = ios[1:]
```

In [3]:

```
# The explore_data function will be useful for explore rows in a more
# readable way.

def explore_data(dataset, start, end, rows_and_columns=False):
    dataset_slice = dataset[start:end]
    for row in dataset_slice:
        print(row)
        print('\n')


    if rows_and_columns:
        print('Number of rows:', len(dataset))
        print('Number of columns:', len(dataset[0]))

print(android_header)
print('\n')
explore_data(android, 0, 3, True)
```

```
['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type', 'Price', 'Content Rating', 'Genres
', 'Last Updated', 'Current Ver', 'Android Ver']
```

```
['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,000+', 'F
ree', '0', 'Everyone', 'Art & Design', 'January 7, 2018', '1.0.0', '4.0.3 and up']


['Coloring book moana', 'ART_AND_DESIGN', '3.9', '967', '14M', '500,000+', 'Free', '0', 'Everyone', 'Ar
t & Design;Pretend Play', 'January 15, 2018', '2.0.0', '4.0.3 and up']


['U Launcher Lite – FREE Live Cool Themes, Hide Apps', 'ART_AND_DESIGN', '4.7', '87510', '8.7M', '5,000
,000+', 'Free', '0', 'Everyone', 'Art & Design', 'August 1, 2018', '1.2.4', '4.0.3 and up']


Number of rows: 10841
Number of columns: 13
```

In [4]:

```
print(ios_header)
print('\n')
explore_data(ios, 0, 3, True)
```

```
['id', 'track_name', 'size_bytes', 'currency', 'price', 'rating_count_tot', 'rating_count_ver', 'user_r
ating', 'user_rating_ver', 'ver', 'cont_rating', 'prime_genre', 'sup_devices.num', 'ipadSc_urls.num', '
lang.num', 'vpp_lic']


['284882215', 'Facebook', '389879808', 'USD', '0.0', '2974676', '212', '3.5', '3.5', '95.0', '4+', 'Soc
ial Networking', '37', '1', '29', '1']


['389801252', 'Instagram', '113954816', 'USD', '0.0', '2161558', '1289', '4.5', '4.0', '10.23', '12+',
'Photo & Video', '37', '0', '29', '1']


['529479190', 'Clash of Clans', '116476928', 'USD', '0.0', '2130805', '579', '4.5', '4.5', '9.24.12', '
9+', 'Games', '38', '5', '18', '1']


Number of rows: 7197
Number of columns: 16
```

In [5]:

```
# The Google Play data set has a dedicated discussion section, and we can
# see that one of the discussions outlines an error for row 10472.
# The row 10472 corresponds to the app Life Made WI-Fi Touchscreen Photo
# Frame, and we can see that the rating is 19. This is clearly off because
# the maximum rating for a Google Play app is 5. As a consequence,
# we'll delete this row.

print(android[10472])
```

```
['Life Made WI-Fi Touchscreen Photo Frame', '1.9', '19', '3.0M', '1,000+', 'Free', '0', 'Everyone', '',
'February 11, 2018', '1.0.19', '4.0 and up']
```

In [6]:

```
print(len(android))
del android[10472]
print(len(android))
```

```
10841
10840
```

In [7]:

```
# If we explore the Google Play data set long enough, we'll find that some
# apps have more than one entry. For instance, the application Instagram
# has four entries

for app in android:
    name = app[0]
    if name == 'Instagram':
        print(app)
```

```
['Instagram', 'SOCIAL', '4.5', '66577313', 'Varies with device', '1,000,000,000+', 'Free', '0', 'Teen',
```

```
'Social', 'July 31, 2018', 'Varies with device', 'Varies with device']
['Instagram', 'SOCIAL', '4.5', '66577446', 'Varies with device', '1,000,000,000+', 'Free', '0', 'Teen',
'Social', 'July 31, 2018', 'Varies with device', 'Varies with device']
['Instagram', 'SOCIAL', '4.5', '66577313', 'Varies with device', '1,000,000,000+', 'Free', '0', 'Teen',
'Social', 'July 31, 2018', 'Varies with device', 'Varies with device']
['Instagram', 'SOCIAL', '4.5', '66509917', 'Varies with device', '1,000,000,000+', 'Free', '0', 'Teen',
'Social', 'July 31, 2018', 'Varies with device', 'Varies with device']
```

In [8]:

```
duplicate_apps = []
unique_apps = []

for app in android:
    name = app[0]
    if name in unique_apps:
        duplicate_apps.append(name)
    else:
        unique_apps.append(name)

print('Number of duplicate apps:', len(duplicate_apps))
print('\n')
print('Examples of duplicate apps:', duplicate_apps[:15])

# In total, there are 1,181 cases where an app occurs more than once
```

```
Number of duplicate apps: 1181


Examples of duplicate apps: ['Quick PDF Scanner + OCR FREE', 'Box', 'Google My Business', 'ZOOM Cloud M
eetings', 'join.me - Simple Meetings', 'Box', 'Zenefits', 'Google Ads', 'Google My Business', 'Slack',
'FreshBooks Classic', 'Insightly CRM', 'QuickBooks Accounting: Invoicing & Expenses', 'HipChat - Chat B
uilt for Teams', 'Xero Accounting Software']
```

In [9]:

```
# The Google play data has duplicate entries for instance Instagram

for app in android:
    name = app[0]
    if name == 'Instagram':
        print(app)
```

```
['Instagram', 'SOCIAL', '4.5', '66577313', 'Varies with device', '1,000,000,000+', 'Free', '0', 'Teen',
'Social', 'July 31, 2018', 'Varies with device', 'Varies with device']
['Instagram', 'SOCIAL', '4.5', '66577446', 'Varies with device', '1,000,000,000+', 'Free', '0', 'Teen',
'Social', 'July 31, 2018', 'Varies with device', 'Varies with device']
['Instagram', 'SOCIAL', '4.5', '66577313', 'Varies with device', '1,000,000,000+', 'Free', '0', 'Teen',
'Social', 'July 31, 2018', 'Varies with device', 'Varies with device']
['Instagram', 'SOCIAL', '4.5', '66509917', 'Varies with device', '1,000,000,000+', 'Free', '0', 'Teen',
'Social', 'July 31, 2018', 'Varies with device', 'Varies with device']
```

In [10]:

```
# We will gonna keep only one entrie per app and
# remove duplicate depending on the number
# of reviews, indeed, the higher it is, the more
# recent the data should be

duplicate_apps = []
unique_apps = []

for app in android:
    name = app[0]
    if name in unique_apps:
        duplicate_apps.append(name)
    else:
        unique_apps.append(name)

print('Number of duplicate apps:', len(duplicate_apps))
print('\n')
print('Examples of duplicate apps:', duplicate_apps[:15])
```

```
Number of duplicate apps: 1181


Examples of duplicate apps: ['Quick PDF Scanner + OCR FREE', 'Box', 'Google My Business', 'ZOOM Cloud M
```

```
eetings', 'join.me - Simple Meetings', 'Box', 'Zenefits', 'Google Ads', 'Google My Business', 'Slack',
'FreshBooks Classic', 'Insightly CRM', 'QuickBooks Accounting: Invoicing & Expenses', 'HipChat - Chat B
uilt for Teams', 'Xero Accounting Software']
```

In [11]:

```
# We won't remove rows randomly, but rather we'll keep the rows that have
# the highest number of reviews because the higher the number of reviews,
# the more reliable the ratings.

# We will create a dictionary where each key is a unique app name, and the
# value is the highest number of reviews of that app.
# And then use the dictionary to create a new data set, which will have only one
# entry per app (and we only select the apps with the highest number of
# reviews)

reviews_max = {}

for app in android:
    name = app[0]
    n_reviews = float(app[3])

    if name in reviews_max and reviews_max[name] < n_reviews:
        reviews_max[name] = n_reviews

    elif name not in reviews_max:
        reviews_max[name] = n_reviews
```

In [12]:

```
android_clean = []
already_added = []

for app in android:
    name = app[0]
    n_reviews = float(app[3])

    if (reviews_max[name] == n_reviews) and (name not in already_added):
        android_clean.append(app)
        already_added.append(name)
```

In [13]:

```
# We now verify that android_clean contain only 9659
# beacause there were 1181 duplicates and 10840 rows all in all
explore_data(android_clean, 0, 3, True)
```

```
['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,000+', 'F
ree', '0', 'Everyone', 'Art & Design', 'January 7, 2018', '1.0.0', '4.0.3 and up']


['U Launcher Lite – FREE Live Cool Themes, Hide Apps', 'ART_AND_DESIGN', '4.7', '87510', '8.7M', '5,000
,000+', 'Free', '0', 'Everyone', 'Art & Design', 'August 1, 2018', '1.2.4', '4.0.3 and up']


['Sketch - Draw & Paint', 'ART_AND_DESIGN', '4.5', '215644', '25M', '50,000,000+', 'Free', '0', 'Teen',
'Art & Design', 'June 8, 2018', 'Varies with device', '4.2 and up']


Number of rows: 9659
Number of columns: 13
```

In [14]:

```
# We're gonna write a function which will let us to know if an app's
# name contain more than 3 non-english characters
def is_english(string):
    non_ascii = 0

    for character in string:
        if ord(character) > 127:
            non_ascii += 1

    if non_ascii > 3:
        return False
    else:
        return True
```

```
# We now verify if it works
print(is_english('Docs To Go™ Free Office Suite'))
print(is_english('爱奇艺PPS -《欢乐颂2》电视剧热播'))
```

```
True
False
```

In [15]:

```
android_english = []
ios_english = []

for app in android_clean:
    name = app[0]
    if is_english(name):
        android_english.append(app)

for app in ios:
    name = app[1]
    if is_english(name):
        ios_english.append(app)

explore_data(android_english, 0, 3, True)
print('\n')
explore_data(ios_english, 0, 3, True)
```

```
['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,000+', 'F
ree', '0', 'Everyone', 'Art & Design', 'January 7, 2018', '1.0.0', '4.0.3 and up']


['U Launcher Lite – FREE Live Cool Themes, Hide Apps', 'ART_AND_DESIGN', '4.7', '87510', '8.7M', '5,000
,000+', 'Free', '0', 'Everyone', 'Art & Design', 'August 1, 2018', '1.2.4', '4.0.3 and up']


['Sketch - Draw & Paint', 'ART_AND_DESIGN', '4.5', '215644', '25M', '50,000,000+', 'Free', '0', 'Teen',
'Art & Design', 'June 8, 2018', 'Varies with device', '4.2 and up']


Number of rows: 9614
Number of columns: 13


['284882215', 'Facebook', '389879808', 'USD', '0.0', '2974676', '212', '3.5', '3.5', '95.0', '4+', 'Soc
ial Networking', '37', '1', '29', '1']


['389801252', 'Instagram', '113954816', 'USD', '0.0', '2161558', '1289', '4.5', '4.0', '10.23', '12+',
'Photo & Video', '37', '0', '29', '1']


['529479190', 'Clash of Clans', '116476928', 'USD', '0.0', '2130805', '579', '4.5', '4.5', '9.24.12', '
9+', 'Games', '38', '5', '18', '1']


Number of rows: 6183
Number of columns: 16
```

In [16]:

```
# Now , we are going to isolate free apps because we build only free to
# download and install apps
android_final = []
ios_final = []

for app in android_english:
    price = app[7]
    if price == '0':
        android_final.append(app)

for app in ios_english:
    price = app[4]
    if price == '0.0':
```

```
        ios_final.append(app)

print(len(android_final))
print(len(ios_final))
```

```
8864
3222
```

In [17]:

```
# Our end goal is to add the app on both Google Play and the App Store,
# we need to find app profiles that are successful on both markets. For
# instance, a profile that works well for both markets might be a
# productivity app that makes use of gamification.

def freq_table(dataset, index):
    table = {}
    total = 0

    for row in dataset:
        total += 1
        value = row[index]
        if value in table:
            table[value] += 1
        else:
            table[value] = 1

    table_percentages = {}
    for key in table:
        percentage = (table[key] / total) * 100
        table_percentages[key] = percentage

    return table_percentages


def display_table(dataset, index):
    table = freq_table(dataset, index)
    table_display = []
    for key in table:
        key_val_as_tuple = (table[key], key)
        table_display.append(key_val_as_tuple)

    table_sorted = sorted(table_display, reverse = True)
    for entry in table_sorted:
        print(entry[1], ':', entry[0])

# now, we are going to use the display_table funtion on the Category and
# prime_genre column for each data set

display_table(android_final, 1)
print('\n')
display_table(ios_final, -5)
```

```
FAMILY : 18.907942238267147
GAME : 9.724729241877256
TOOLS : 8.461191335740072
BUSINESS : 4.591606498194946
LIFESTYLE : 3.9034296028880866
PRODUCTIVITY : 3.892148014440433
FINANCE : 3.7003610108303246
MEDICAL : 3.531137184115524
SPORTS : 3.395758122743682
PERSONALIZATION : 3.3167870036101084
COMMUNICATION : 3.2378158844765346
HEALTH_AND_FITNESS : 3.0798736462093865
PHOTOGRAPHY : 2.944494584837545
NEWS_AND_MAGAZINES : 2.7978339350180503
SOCIAL : 2.6624548736462095
TRAVEL_AND_LOCAL : 2.33528880866426
SHOPPING : 2.2450361010830324
BOOKS_AND_REFERENCE : 2.1435018050541514
DATING : 1.861462093862816
VIDEO_PLAYERS : 1.7937725631768955
MAPS_AND_NAVIGATION : 1.3989169675090252
FOOD_AND_DRINK : 1.2409747292418771
EDUCATION : 1.1620036101083033
```

```
ENTERTAINMENT : 0.9589350180505415
LIBRARIES_AND_DEMO : 0.9363718411552346
AUTO_AND_VEHICLES : 0.9250902527075812
HOUSE_AND_HOME : 0.8235559566787004
WEATHER : 0.8009927797833934
EVENTS : 0.7107400722021661
PARENTING : 0.6543321299638989
ART_AND_DESIGN : 0.6430505415162455
COMICS : 0.6204873646209386
BEAUTY : 0.5979241877256317


Games : 58.16263190564867
Entertainment : 7.883302296710118
Photo & Video : 4.9658597144630665
Education : 3.662321539416512
Social Networking : 3.2898820608317814
Shopping : 2.60707635009311
Utilities : 2.5139664804469275
Sports : 2.1415270018621975
Music : 2.0484171322160147
Health & Fitness : 2.0173805090006205
Productivity : 1.7380509000620732
Lifestyle : 1.5828677839851024
News : 1.3345747982619491
Travel : 1.2414649286157666
Finance : 1.1173184357541899
Weather : 0.8690254500310366
Food & Drink : 0.8069522036002483
Reference : 0.5586592178770949
Business : 0.5276225946617008
Book : 0.4345127250155183
Navigation : 0.186219739292365
Medical : 0.186219739292365
Catalogs : 0.12414649286157665
```

In [18]:

```
# Now, we want to find out what genres are the most popular
# (have the most users). To do so, we will calculate the average
# number of installs for each app genre. For the Google Play data
# set, we find this information in the Installs column, but
# this information is missing for the App Store data set. So we'll
# take the total number of user ratings as a proxy, which we
# can find in the rating_count_tot app.

# for the App Store

genres_ios = freq_table(ios_final, -5)

for genre in genres_ios:
    total = 0
    len_genre = 0
    for app in ios_final:
        genre_app = app[-5]
        if genre_app == genre:
            n_ratings = float(app[5])
            total += n_ratings
            len_genre += 1
    avg_n_ratings = total / len_genre
    print(genre, ':', avg_n_ratings)
```

```
Utilities : 18684.456790123455
Productivity : 21028.410714285714
Finance : 31467.944444444445
Weather : 52279.892857142855
Social Networking : 71548.34905660378
Food & Drink : 33333.92307692308
Navigation : 86090.33333333333
Entertainment : 14029.830708661417
Medical : 612.0
Games : 22788.6696905016
News : 21248.023255813954
Sports : 23008.898550724636
Book : 39758.5
Reference : 74942.11111111111
```

```
Business : 7491.117647058823
Lifestyle : 16485.764705882353
Photo & Video : 28441.54375
Education : 7003.983050847458
Travel : 28243.8
Music : 57326.530303030304
Health & Fitness : 23298.015384615384
Shopping : 26919.690476190477
Catalogs : 4004.0
```

In [19]:

```
# We have data about the number of installs for the Google Play market, so
# we should be able to get a clearer picture about genre popularity.
# However, the install numbers don't seem precise enough. We can see that
# most values are open-ended (100+, 1,000+, 5,000+, etc.
# we only want to find out which app genres attract the most users, and we
# don't need perfect precision with respect to the number of users.
display_table(android_final, 5)
```

```
1,000,000+ : 15.726534296028879
100,000+ : 11.552346570397113
10,000,000+ : 10.548285198555957
10,000+ : 10.198555956678701
1,000+ : 8.393501805054152
100+ : 6.915613718411552
5,000,000+ : 6.825361010830325
500,000+ : 5.561823104693141
50,000+ : 4.7721119133574
5,000+ : 4.512635379061372
10+ : 3.5424187725631766
500+ : 3.2490974729241873
50,000,000+ : 2.3014440433213
100,000,000+ : 2.1322202166064983
50+ : 1.917870036101083
5+ : 0.78971119133574
1+ : 0.5076714801444043
500,000,000+ : 0.2707581227436823
1,000,000,000+ : 0.22563176895306858
0+ : 0.04512635379061372
0 : 0.01128158844765343
```

In [20]:

```
categories_android = freq_table(android_final, 1)
for category in categories_android:
    total = 0
    len_category = 0
    for app in android_final:
        category_app = app[1]
        if category_app == category:
            n_installs = app[5]
            n_installs = n_installs.replace('+', '')
            n_installs = n_installs.replace(',', '')
            total += float(n_installs)
            len_category += 1
    avg_n_installs = total / len_category
    print(category, ':', avg_n_installs)
```

```
EDUCATION : 1833495.145631068
LIBRARIES_AND_DEMO : 638503.734939759
BUSINESS : 1712290.1474201474
BEAUTY : 513151.88679245283
TOOLS : 10801391.298666667
FINANCE : 1387692.475609756
PHOTOGRAPHY : 17840110.40229885
SPORTS : 3638640.1428571427
AUTO_AND_VEHICLES : 647317.8170731707
WEATHER : 5074486.197183099
LIFESTYLE : 1437816.2687861272
EVENTS : 253542.22222222222
FOOD_AND_DRINK : 1924897.7363636363
PERSONALIZATION : 5201482.6122448975
COMMUNICATION : 38456119.167247385
SHOPPING : 7036877.311557789
MEDICAL : 120550.61980830671
TRAVEL_AND_LOCAL : 13984077.710144928
```

```
ART_AND_DESIGN : 1986335.0877192982
ENTERTAINMENT : 11640705.88235294
DATING : 854028.8303030303
COMICS : 817657.2727272727
SOCIAL : 23253652.127118643
PARENTING : 542603.6206896552
MAPS_AND_NAVIGATION : 4056941.7741935486
FAMILY : 3695641.8198090694
HEALTH_AND_FITNESS : 4188821.9853479853
VIDEO_PLAYERS : 24727872.452830188
HOUSE_AND_HOME : 1331540.5616438356
GAME : 15588015.603248259
BOOKS_AND_REFERENCE : 8767811.894736841
NEWS_AND_MAGAZINES : 9549178.467741935
PRODUCTIVITY : 16787331.344927534
```

In [21]:

```
for app in android_final:
    if app[1] == 'HEALTH_AND_FITNESS' and (app[5] == '1,000,000,000+'
                                        or app[5] == '500,000,000+'
                                        or app[5] == '100,000,000+'):
        print(app[0], ':', app[5])


# Finally, our conclusion is that we could create a new sort of health app
# wich are very successful in each OS.
```

```
Period Tracker - Period Calendar Ovulation Tracker : 100,000,000+
Samsung Health : 500,000,000+
```