# Apple Store and Google Play Store profitable app analysis

This project is to find out which mobile apps are profitable in Apple Store and Google Play Store.

The goal for this proejct is to drow a data-driven decisions with respect to the kind of apps will be made in the future. We are going to find out what kind of apps attract more users on Apple Store and Google Play Store.

# Opening and Exploring the Data

googleplaystore.csv and AppleStore.csv are:

- A data set containing data about approximately 10,000 Android apps from Google Play; the data was collected in August 2018.
- A data set containing data about approximately 7,000 iOS apps from the App Store; the data was collected in July 2017.

The open_file() function takes in two parameters:

- filename: a file name that needs to be read
- header: 'True' by default if the file contains a header row, 'False' if the file doesn't contain a header row

In [1]:

```
def open_file(filename, header=True):
    opened_file = open(filename, encoding='utf8')
    from csv import reader
    read_file = reader(opened_file)
    data = list(read_file)

    if header:
        return data[0], data[1:]
    else:
        return data

apple_header, apple_data = open_file('AppleStore.csv')
google_header, google_data = open_file('googleplaystore.csv')
```

The explore_data() function takes in four parameters:

- dataset: expected to be a list of lists
- start and end: expected to be integers and represents the starting and the ending indices of a slice from the data set
- rows_and_columns: expected to be boolean and has false as a default argument. Ture if you want to see number of rows and columns.

In [2]:

```
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]))
```

In [3]:

```
print('This is Apple Store data:\n')
print(apple_header)
print('\n')
explore_data(apple_data, 0, 3, True)
```

```
This is Apple Store data:

['id', 'track_name', 'size_bytes', 'currency', 'price', 'rating_count_tot', 'rating_count_ver', 'u
```

```
ser_rating', '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+',
'Social Networking', '37', '1', '29', '1']


['389801252', 'Instagram', '113954816', 'USD', '0.0', '2161558', '1289', '4.5', '4.0', '10.23', '1
2+', '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
```

With a quick glance of the data, the columns that might be useful for the purpose of our analysis are 'track_name', 'currency', 'price', 'rating_count_tot' and 'prime_genre'. Details about each column can be found at documentation.

In [4]:

```
print('\nThis is Google Play Store data:\n')
print(google_header)
print('\n')
explore_data(google_data, 0, 3, True)
```

```
This is Google Play Store data:

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


['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,000+
', 'Free', '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', 'Art & 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
```

With a quick glance of the data, the columns that might be useful for the purpose of our analysis are 'App', 'Category', 'Reviews', 'Installs', 'Type', 'Price' and 'Genre'. Details about each column can be found at documentation.

# Data Cleaning

Data cleaning is about removing or correcting wrong data, removing duplicate data, and modifying the data to fit the purpose of analysis.

In this section, I am going to perform Data Cleaning which:

- Detect inaccurate data, and correct or remove it
- Detect duplicate data, and remove the duplicates To narrow down the targets towards an English-Speaking audience and free apps, I will:
- Remove non-English apps
- Remove apps that aren't free

# Removing Wrong Data

In [5]:

```
for row in google_data:
    column_length = len(google_header)
    row_length = len(row)
    if row_length != column_length:
        print(google_header)  # header
        print('\n')
        print('Wrong row: ')
        print(row)
        print('\n')
        print('Index of the wrong row: ')
        print(google_data.index(row))

print('\n')
print('Correct one: ')
print(google_data[0]) # correct row
print('\n')
```

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


Wrong row:
['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']


Index of the wrong row:
10472


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

Column 10472 is shifted to the left by one from 'Category' to 'Android Ver' which should be deleted from the data set.

Delete the wrong row from the data set.

In [6]:

```
print('Length of the data before deletion: ', len(google_data))
#del android[10472]  # don't run this more than once
print('Length of the data after deletion: ', len(google_data))
```

```
Length of the data before deletion:  10841
Length of the data after deletion:  10841
```

Performing same process with Apple Store data set

In [7]:

```
for row in apple_data:
    column_length = len(apple_header)
    row_length = len(row)
    if row_length != column_length:
        print(apple_header) # header
        print('\n')
        print('Wrong row: ')
        print(row)
        print('\n')
        print('Index of the wrong row: ')
        print(apple_data.index(row))
```

Can't seem to find wrong data row in apple data set

# Removing Duplicate Data

In [8]:

```
print(google_header)
for app in google_data:
    name = app[0]
    if name == '10 Best Foods for You':
        print(app)
```

```
['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type', 'Price', 'Content Rating', 'G
enres', 'Last Updated', 'Current Ver', 'Android Ver']
['10 Best Foods for You', 'HEALTH_AND_FITNESS', '4.0', '2490', '3.8M', '500,000+', 'Free', '0', 'E
veryone 10+', 'Health & Fitness', 'February 17, 2017', '1.9', '2.3.3 and up']
['10 Best Foods for You', 'HEALTH_AND_FITNESS', '4.0', '2490', '3.8M', '500,000+', 'Free', '0', 'E
veryone 10+', 'Health & Fitness', 'February 17, 2017', '1.9', '2.3.3 and up']
```

Now try to find all of the duplicate data.

In [9]:

```
unique_apps = []
duplicate_apps = []

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

print('Number of duplicate apps in Google Play Store : ',len(duplicate_apps))
```

```
Number of duplicate apps in Google Play Store :  1181
```

In [10]:

```
reviews_max = {}
for app in google_data:
    name = app[0]
    review = app[3]
    if name in reviews_max and reviews_max[name] < review:
        reviews_max[name] = review
    elif name not in reviews_max:
        reviews_max[name] = review

print('Data including duplicates: ',len(google_data))
print('Expected data length: ',len(google_data) - 1181)
print('Data after clean up: ',len(reviews_max))
```

```
Data including duplicates:  10841
Expected data length:  9660
Data after clean up:  9660
```

In [11]:

```
google_clean_data = []
google_already_added_app = []

for app in google_data:
    name = app[0]
    review = app[3]
    if (reviews_max[name] == review) and (name not in google_already_added_app):
        google_clean_data.append(app)
        google_already_added_app.append(name)

print(google_header,'\n')
explore_data(google_clean_data, 0, 3, True)
```

```
['App'  'Category'  'Rating'  'Reviews'  'Size'  'Installs'  'Type'  'Price'  'Content Rating'  'G
```

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

['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,000+
', 'Free', '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', 'T
een', 'Art & Design', 'June 8, 2018', 'Varies with device', '4.2 and up']


Number of rows:  9660
Number of columns:  13
```

In [12]:

```
duplicate_apps = []
unique_apps = []
for app in apple_data:
    name = app[1]
    if name in unique_apps:
        duplicate_apps.append(name)
    else:
        unique_apps.append(name)

print(duplicate_apps)
print('Number of duplicate apps in Apple App Store : ',len(duplicate_apps))
```

```
['Mannequin Challenge', 'VR Roller Coaster']
Number of duplicate apps in Apple App Store :  2
```

In [13]:

```
rating_count_max = {}
for app in apple_data:
    name = app[1]
    rating_count_tot = app[5]
    if name in rating_count_max and rating_count_max[name] < rating_count_tot:
        rating_count_max[name] = rating_count_tot
    elif name not in rating_count_max:
        rating_count_max[name] = rating_count_tot

print('Apple Store Data before clean up: ', len(apple_data))
print('Expected number of data after clean up: ',len(apple_data) - 2)
print('Apple Store Data after clean up: ', len(rating_count_max))
```

```
Apple Store Data before clean up:  7197
Expected number of data after clean up:  7195
Apple Store Data after clean up:  7195
```

In [14]:

```
apple_clean_data = []
already_added_apple_data = []
for app in apple_data:
    name = app[1]
    rating_count_tot = app[5]
    if name not in already_added_apple_data and rating_count_max[name] == rating_count_tot:
        apple_clean_data.append(app)
        already_added_apple_data.append(name)

print(apple_header,'\n')
explore_data(apple_clean_data, 0, 3, True)
```

```
['id', 'track_name', 'size_bytes', 'currency', 'price', 'rating_count_tot', 'rating_count_ver', 'u
ser_rating', '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+'
```

```
'Social Networking', '37', '1', '29', '1']


['389801252', 'Instagram', '113954816', 'USD', '0.0', '2161558', '1289', '4.5', '4.0', '10.23', '1
2+', '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:  7195
Number of columns:  16
```

# Isolating the non-English Apps

I am going to isolate the non-English Apps from the data since the target for this project is 'English Speakers'.

In [15]:

```
def is_english(string):
    for character in string:
        if ord(character) > 127:
            return False
    return True

print(is_english('Min Oh'))
print(is_english('오승민'))
```

```
True
False
```

In [16]:

```
def modified_is_english(string):
    possible_errors = 0
    for character in string:
        if ord(character) > 127:
            possible_errors += 1
    if possible_errors > 3:
        return False
    else:
        return True

print(modified_is_english('Instagram'))
print(modified_is_english('爱奇艺PPS -《欢乐颂2》电视剧热播'))
print(modified_is_english('Docs To Go™ Free Office Suite'))
print(modified_is_english('Instachat 😜'))
```

```
True
False
True
True
```

This modified version help us to minimize mistakes on eliminating non-english apps by allowing upto 3 non-english words in title.

In [33]:

```
google_english_data = []
google_non_english_data = []
for app in google_clean_data:
    name = app[0]
    if modified_is_english(name):
        google_english_data.append(app)
    else:
        google_non_english_data.append(app)

explore_data(google_english_data, 0, 3, True)
```

```
['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,000+
', 'Free', '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', 'T
een', 'Art & Design', 'June 8, 2018', 'Varies with device', '4.2 and up']


Number of rows:  9615
Number of columns:  13
```

Out of 9660 Apps in Google Play Store, 45 Apps contain more than 3 non-English words in them.

In [18]:

```
apple_english_data = []
apple_non_english_data = []
for app in apple_clean_data:
    name = app[1]
    if modified_is_english(name):
        apple_english_data.append(app)
    else:
        apple_non_english_data.append(app)

explore_data(apple_english_data, 0, 3, True)
```

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


['389801252', 'Instagram', '113954816', 'USD', '0.0', '2161558', '1289', '4.5', '4.0', '10.23', '1
2+', '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:  6181
Number of columns:  16
```

Out of 7195 Apps in Apple App Store, 1014 Apps contain more than 3 non-English words in them.

# Isolating the Free Apps

In [19]:

```
google_final = []

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

print('There are ',len(google_final), ' free, english, non_duplicate, and correct data in Google P
lay Store data.')
```

```
There are  8862  free, english, non_duplicate, and correct data in Google Play Store data.
```

In [20]:

```
apple_final = []

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

```
    if price == '0.0':
        apple_final.append(app)

print('There are ',len(apple_final), ' free, english, non_duplicate, and correct data in Apple App
Store data.')
```

```
There are  3220  free, english, non_duplicate, and correct data in Apple App Store data.
```

# Frequency Tables

In [21]:

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

    for app in dataset:
        total += 1
        value = app[index]
        if value in table:
            table[value] += 1
        else:
            table[value] = 1
    percentage_table = {}
    for app in table:
        percentage = (table[app] / total) * 100
        percentage_table[app] = percentage
    return percentage_table

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])
```

In [22]:

```
print('Genre')
display_table(google_final, 9)
print('\n')
print('Category')
display_table(google_final, 1)
```

```
Genre
Tools : 8.440532611148726
Entertainment : 6.070864364703228
Education : 5.348679756262695
Business : 4.5926427443015125
Productivity : 3.8930264048747465
Lifestyle : 3.8930264048747465
Finance : 3.7011961182577298
Medical : 3.5206499661475967
Sports : 3.4642292936131795
Personalization : 3.3175355450236967
Communication : 3.238546603475513
Action : 3.1031369893929135
Health & Fitness : 3.080568720379147
Photography : 2.945159106296547
News & Magazines : 2.798465357707064
Social : 2.663055743624464
Travel & Local : 2.324531708417964
Shopping : 2.2455427668697814
Books & Reference : 2.143985556307831
Simulation : 2.0424283457458814
Dating : 1.8618821936357481
Arcade : 1.8505980591288649
Video Players & Editors : 1.7716091175806816
```

Casual : 1.7490408485669149
Maps & Navigation : 1.399232678853532
Food & Drink : 1.2412547957571656
Puzzle : 1.128413450688332
Racing : 0.9930038366057323
Role Playing : 0.9365831640713158
Libraries & Demo : 0.9365831640713158
Auto & Vehicles : 0.9252990295644324
Strategy : 0.9140148950575491
House & Home : 0.8237418190024826
Weather : 0.8011735499887158
Events : 0.7109004739336493
Adventure : 0.6770480704129994
Comics : 0.6093432633716994
Beauty : 0.598059128864816
Art & Design : 0.598059128864816
Parenting : 0.49650191830286616
Card : 0.4400812457684496
Casino : 0.4287971112615662
Trivia : 0.41751297675468296
Educational;Education : 0.3949447077409162
Educational : 0.3723764387271496
Board : 0.3723764387271496
Education;Education : 0.3385240352064997
Word : 0.2595350936583164
Casual;Pretend Play : 0.23696682464454977
Music : 0.2031144211238998
Puzzle;Brain Games : 0.18054615211013314
Racing;Action & Adventure : 0.16926201760324985
Entertainment;Music & Video : 0.16926201760324985
Casual;Brain Games : 0.13540961408259986
Casual;Action & Adventure : 0.13540961408259986
Arcade;Action & Adventure : 0.12412547957571654
Action;Action & Adventure : 0.1015572105619499
Educational;Pretend Play : 0.09027307605506657
Board;Brain Games : 0.09027307605506657
Simulation;Action & Adventure : 0.07898894154818326
Parenting;Education : 0.07898894154818326
Entertainment;Brain Games : 0.07898894154818326
Parenting;Music & Video : 0.06770480704129993
Educational;Brain Games : 0.06770480704129993
Casual;Creativity : 0.06770480704129993
Art & Design;Creativity : 0.06770480704129993
Education;Pretend Play : 0.056420672534416606
Role Playing;Pretend Play : 0.045136538027533285
Education;Creativity : 0.045136538027533285
Role Playing;Action & Adventure : 0.033852403520649964
Puzzle;Action & Adventure : 0.033852403520649964
Entertainment;Creativity : 0.033852403520649964
Entertainment;Action & Adventure : 0.033852403520649964
Educational;Creativity : 0.033852403520649964
Educational;Action & Adventure : 0.033852403520649964
Education;Music & Video : 0.033852403520649964
Education;Brain Games : 0.033852403520649964
Education;Action & Adventure : 0.033852403520649964
Adventure;Action & Adventure : 0.033852403520649964
Video Players & Editors;Music & Video : 0.022568269013766643
Sports;Action & Adventure : 0.022568269013766643
Simulation;Pretend Play : 0.022568269013766643
Puzzle;Creativity : 0.022568269013766643
Music;Music & Video : 0.022568269013766643
Entertainment;Pretend Play : 0.022568269013766643
Casual;Education : 0.022568269013766643
Board;Action & Adventure : 0.022568269013766643
Video Players & Editors;Creativity : 0.011284134506883321
Trivia;Education : 0.011284134506883321
Travel & Local;Action & Adventure : 0.011284134506883321
Tools;Education : 0.011284134506883321
Strategy;Education : 0.011284134506883321
Strategy;Creativity : 0.011284134506883321
Strategy;Action & Adventure : 0.011284134506883321
Simulation;Education : 0.011284134506883321
Role Playing;Brain Games : 0.011284134506883321
Racing;Pretend Play : 0.011284134506883321
Puzzle;Education : 0.011284134506883321
Parenting;Brain Games : 0.011284134506883321
Music & Audio;Music & Video : 0.011284134506883321

```
Lifestyle;Pretend Play : 0.011284134506883321
Lifestyle;Education : 0.011284134506883321
Health & Fitness;Education : 0.011284134506883321
Health & Fitness;Action & Adventure : 0.011284134506883321
Entertainment;Education : 0.011284134506883321
Communication;Creativity : 0.011284134506883321
Comics;Creativity : 0.011284134506883321
Casual;Music & Video : 0.011284134506883321
Card;Brain Games : 0.011284134506883321
Card;Action & Adventure : 0.011284134506883321
Books & Reference;Education : 0.011284134506883321
Art & Design;Pretend Play : 0.011284134506883321
Art & Design;Action & Adventure : 0.011284134506883321
Arcade;Pretend Play : 0.011284134506883321
Adventure;Education : 0.011284134506883321


Category
FAMILY : 18.934777702550214
GAME : 9.693071541412774
TOOLS : 8.451816745655607
BUSINESS : 4.5926427443015125
LIFESTYLE : 3.9043105393816293
PRODUCTIVITY : 3.8930264048747465
FINANCE : 3.7011961182577298
MEDICAL : 3.5206499661475967
SPORTS : 3.39652448657188
PERSONALIZATION : 3.3175355450236967
COMMUNICATION : 3.238546603475513
HEALTH_AND_FITNESS : 3.080568720379147
PHOTOGRAPHY : 2.945159106296547
NEWS_AND_MAGAZINES : 2.798465357707064
SOCIAL : 2.663055743624464
TRAVEL_AND_LOCAL : 2.335815842924848
SHOPPING : 2.2455427668697814
BOOKS_AND_REFERENCE : 2.143985556307831
DATING : 1.8618821936357481
VIDEO_PLAYERS : 1.7941773865944481
MAPS_AND_NAVIGATION : 1.399232678853532
FOOD_AND_DRINK : 1.2412547957571656
EDUCATION : 1.1735499887158656
ENTERTAINMENT : 0.9591514330850823
LIBRARIES_AND_DEMO : 0.9365831640713158
AUTO_AND_VEHICLES : 0.9252990295644324
HOUSE_AND_HOME : 0.8237418190024826
WEATHER : 0.8011735499887158
EVENTS : 0.7109004739336493
PARENTING : 0.6544798013992327
ART_AND_DESIGN : 0.6431956668923494
COMICS : 0.6206273978785828
BEAUTY : 0.598059128864816
```

In Google Play Store data set, the difference between 'Genre' and 'Category' is not clear. However, using 'Category' will fit our purpose since it is more general compare to 'Genre'.

Category of Google Play Store is overally well spread. Productivity Apps (Family, Tools, Business, Life Style, etc) are majority beside 9.6% of the 'Game' category.

In [23]:

```
print(apple_header)
display_table(apple_final, 11)
```

```
['id', 'track_name', 'size_bytes', 'currency', 'price', 'rating_count_tot', 'rating_count_ver', 'u
ser_rating', 'user_rating_ver', 'ver', 'cont_rating', 'prime_genre', 'sup_devices.num',
'ipadSc_urls.num', 'lang.num', 'vpp_lic']
Games : 58.13664596273293
Entertainment : 7.888198757763975
Photo & Video : 4.968944099378882
Education : 3.6645962732919255
Social Networking : 3.291925465838509
Shopping : 2.608695652173913
Utilities : 2.515527950310559
Sports : 2.142857142857143
Music : 2.049689440993789
```

```
Music : 2.049689440993789
Health & Fitness : 2.018633540372671
Productivity : 1.7391304347826086
Lifestyle : 1.5838509316770186
News : 1.3354037267080745
Travel : 1.2422360248447204
Finance : 1.1180124223602486
Weather : 0.8695652173913043
Food & Drink : 0.8074534161490683
Reference : 0.5590062111801243
Business : 0.5279503105590062
Book : 0.43478260869565216
Navigation : 0.18633540372670807
Medical : 0.18633540372670807
Catalogs : 0.12422360248447205
```

Apple App Store is heavily skewed with 58.1% of the 'Game' category. The first impression of Apple App Store compare to Google Play Store is Apps that are desinged for fun (Game, Entertainment, Photo & Video, Music, Sports, etc) is dominating.

However, the fact that fun apps are the most numerous doesn't also imply that they also have the greatest number of users — the demand might not be the same as the offer. So I will create freqency table to examine that.

# Most Popular Apps by Genre

I am going to use 'average number of installations' to find out which genre is more popular than others. Number of installs in Google Play Store data is not percise, however, I do not need percise data for our purpose. To use the data as is, '+' and ',' need to be cleaned and it needs to be float to calculate the average install number.

In [38]:

```
freq_category = freq_table(google_final, 1)
category_dict = {}
for genre in freq_category:
    total = 0
    len_genre = 0
    for app in google_final:
        genre_app = app[1]
        if genre_app == genre:
            n_install = app[5]
            n_install = n_install.replace('+', '')
            n_install = n_install.replace(',', '')
            total += float(n_install)
            len_genre += 1
    avg_install = total / len_genre
    category_dict[genre] = avg_install
category_dict = sorted(category_dict.items(), key = lambda t: t[1], reverse = True)
for i in category_dict:
    print(i)
```

```
('COMMUNICATION', 38456119.167247385)
('VIDEO_PLAYERS', 24727872.452830188)
('SOCIAL', 23253652.127118643)
('PHOTOGRAPHY', 17805627.643678162)
('PRODUCTIVITY', 16787331.344927534)
('GAME', 15560965.599534342)
('TRAVEL_AND_LOCAL', 13984077.710144928)
('ENTERTAINMENT', 11640705.88235294)
('TOOLS', 10682301.033377837)
('NEWS_AND_MAGAZINES', 9549178.467741935)
('BOOKS_AND_REFERENCE', 8767811.894736841)
('SHOPPING', 7036877.311557789)
('PERSONALIZATION', 5201482.6122448975)
('WEATHER', 5074486.197183099)
('HEALTH_AND_FITNESS', 4188821.9853479853)
('MAPS_AND_NAVIGATION', 4056941.7741935486)
('FAMILY', 3694276.334922527)
('SPORTS', 3638640.1428571427)
('ART_AND_DESIGN', 1986335.0877192982)
('FOOD_AND_DRINK', 1924897.7363636363)
('EDUCATION', 1820673.076923077)
('BUSINESS', 1712290.1474201474)
('LIFESTYLE', 1437816.2687861272)
```

```
('FINANCE', 1387692.475609756)
('HOUSE_AND_HOME', 1331540.5616438356)
('DATING', 854028.8303030303)
('COMICS', 817657.2727272727)
('AUTO_AND_VEHICLES', 647317.8170731707)
('LIBRARIES_AND_DEMO', 638503.734939759)
('PARENTING', 542603.6206896552)
('BEAUTY', 513151.88679245283)
('EVENTS', 253542.22222222222)
('MEDICAL', 120616.48717948717)
```

'Communication', 'Video_players', 'Social', and 'Photography' are the major categories that users installed the most.

Let's look at one of the major categories in detail.

In [27]:

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

```
WhatsApp Messenger : 1,000,000,000+
imo beta free calls and text : 100,000,000+
Android Messages : 100,000,000+
Google Duo - High Quality Video Calls : 500,000,000+
Messenger – Text and Video Chat for Free : 1,000,000,000+
imo free video calls and chat : 500,000,000+
Skype - free IM & video calls : 1,000,000,000+
Who : 100,000,000+
GO SMS Pro - Messenger, Free Themes, Emoji : 100,000,000+
LINE: Free Calls & Messages : 500,000,000+
Google Chrome: Fast & Secure : 1,000,000,000+
Firefox Browser fast & private : 100,000,000+
UC Browser - Fast Download Private & Secure : 500,000,000+
Gmail : 1,000,000,000+
Hangouts : 1,000,000,000+
Messenger Lite: Free Calls & Messages : 100,000,000+
Kik : 100,000,000+
KakaoTalk: Free Calls & Text : 100,000,000+
Opera Mini - fast web browser : 100,000,000+
Opera Browser: Fast and Secure : 100,000,000+
Telegram : 100,000,000+
Truecaller: Caller ID, SMS spam blocking & Dialer : 100,000,000+
UC Browser Mini -Tiny Fast Private & Secure : 100,000,000+
Viber Messenger : 500,000,000+
WeChat : 100,000,000+
Yahoo Mail – Stay Organized : 100,000,000+
BBM - Free Calls & Messages : 100,000,000+
```

Communication category is dominated by big named companies(Skype, Google Duo, WhatsApp, Android Messages, etc) that are hard to compete with.

In [28]:

```
under_100_m = []

for app in google_final:
    n_installs = app[5]
    n_installs = n_installs.replace(',', '')
    n_installs = n_installs.replace('+', '')
    if (app[1] == 'COMMUNICATION') and (float(n_installs) < 100000000):
        under_100_m.append(float(n_installs))

sum(under_100_m) / len(under_100_m)
```

Out[28]:

```
3603485.3884615386
```

After removing apps that have over 100 million installs, the average reduced roughly ten times.

In [37]:

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

under_100_m = []

for app in google_final:
    n_installs = app[5]
    n_installs = n_installs.replace(',', '')
    n_installs = n_installs.replace('+', '')
    if (app[1] == 'VIDEO_PLAYERS') and (float(n_installs) < 100000000):
        under_100_m.append(float(n_installs))

sum(under_100_m) / len(under_100_m)
```

```
YouTube : 1,000,000,000+
Motorola Gallery : 100,000,000+
VLC for Android : 100,000,000+
Google Play Movies & TV : 1,000,000,000+
MX Player : 500,000,000+
Dubsmash : 100,000,000+
VivaVideo - Video Editor & Photo Movie : 100,000,000+
VideoShow-Video Editor, Video Maker, Beauty Camera : 100,000,000+
Motorola FM Radio : 100,000,000+
```

Out[37]:

```
5544878.133333334
```

Similar trend showed with 'VIDEO_PLAYERS' category, the average was reduced by roughly four times. From this information, we can extract the fact that these app categories might seem more popular than they really are. Moreover, it is vital to avoid these categories for start up company to build a new App to succeed in the Google Play Store.

Number of installation doesn't exist in the Apple App Store's data. One alternate way to find users' interest would be number of ratings.

In [41]:

```
freq_category = freq_table(apple_final, 11)
category_dict = {}
for genre in freq_category:
    total = 0
    len_genre = 0
    for app in apple_final:
        genre_app = app[11]
        if genre_app == genre:
            n_rating = app[5]
            total += float(n_rating)
            len_genre += 1
    avg_rating = total / len_genre
    category_dict[genre] = avg_rating
category_dict = sorted(category_dict.items(), key = lambda t: t[1], reverse = True)
for i in category_dict:
    print(i)
```

```
('Navigation', 86090.33333333333)
('Reference', 74942.11111111111)
('Social Networking', 71548.34905660378)
('Music', 57326.530303030304)
('Weather', 52279.892857142855)
('Book', 39758.5)
('Food & Drink', 33333.92307692308)
('Finance', 31467.944444444445)
('Photo & Video', 28441.54375)
('Travel', 28243.8)
('Shopping', 26919.690476190477)
```

```
('Health & Fitness', 23298.015384615384)
('Sports', 23008.898550724636)
('Games', 22812.903311965812)
('News', 21248.023255813954)
('Productivity', 21028.410714285714)
('Utilities', 18684.456790123455)
('Lifestyle', 16485.764705882353)
('Entertainment', 14029.830708661417)
('Business', 7491.117647058823)
('Education', 7003.983050847458)
('Catalogs', 4004.0)
('Medical', 612.0)
```

Since we are looking to make an app that will succeed in both Apple Store and Google Play Store, we can eliminate some of these categories.

'Music' category only exists in Apple Store, we eliminated 'Social Networking' category from Google Play Store, 'Reference' is dominated by Bible and 'Book' category is dominated by big named comapanies' book reader.

However, 'Weather' category exists in both Google Play Store and Apple Store, not dominated by big named companies, and have decent number of user interest. (number of installations & number of ratings).

In [67]:

```
for app in google_final:
    if app[1] == 'WEATHER':
        print(app[0], ':', app[5])
```

```
The Weather Channel: Rain Forecast & Storm Alerts : 50,000,000+
Weather forecast : 1,000,000+
AccuWeather: Daily Forecast & Live Weather Reports : 50,000,000+
Live Weather Pro : 10,000+
Weather by WeatherBug: Forecast, Radar & Alerts : 10,000,000+
weather - weather forecast : 1,000,000+
MyRadar NOAA Weather Radar : 10,000,000+
SMHI Weather : 1,000,000+
Free live weather on screen : 1,000,000+
Weather Radar Widget : 1,000,000+
Weather -Simple weather forecast : 10,000,000+
Weather Crave : 5,000,000+
Klara weather : 500,000+
Yahoo Weather : 10,000,000+
Real time Weather Forecast : 1,000,000+
METEO FRANCE : 5,000,000+
APE Weather ( Live Forecast) : 5,000,000+
Live Weather & Daily Local Weather Forecast : 1,000,000+
Weather : 10,000,000+
Rainfall radar - weather : 5,000,000+
Yahoo! Weather for SH Forecast for understanding the approach of rain clouds Free : 1,000,000+
The Weather Network : 5,000,000+
Klart.se - Sweden's best weather : 1,000,000+
GO Weather - Widget, Theme, Wallpaper, Efficient : 50,000,000+
Info BMKG : 1,000,000+
Weather From DMI/YR : 100,000+
wetter.com - Weather and Radar : 10,000,000+
Storm Radar: Tornado Tracker & Hurricane Alerts : 1,000,000+
Yandex.Weather : 10,000,000+
Local Weather Forecast & Visual Widget : 500,000+
Wetter by t-online.de : 1,000,000+
HTC Weather : 10,000,000+
AEMET's time : 1,000,000+
New 2018 Weather App & Widget : 500,000+
Météociel : 500,000+
Climatempo Lite - 15 day weather forecast : 100,000+
ForecaWeather : 1,000,000+
HumorCast - Authentic Weather : 100,000+
W - Weather Forecast & Animated Radar Maps : 5,000+
Weather & Clock Widget for Android : 50,000,000+
WeatherClear - Ad-free Weather, Minute forecast : 50,000+
Ag Weather Tools : 500+
Wind & Weather Meter for Ag : 1,000+
Au Weather Free : 100,000+
Amber Weather : 10,000,000+
ByssWeather for Wear OS : 1,000,000+
```

```
Sun & Moon AR Locator : 10,000+
RadarNow! : 5,000,000+
Skywatch BL : 1,000+
Weather BZ : 100,000+
Fu*** Weather (Funny Weather) : 1,000,000+
WebCams : 100,000+
Windguru Lite : 1,000,000+
World Webcams : 1,000,000+
DS Barometer - Altimeter and Weather Information : 100,000+
DS Thermometer : 100,000+
Météo Algérie DZ : 100,000+
Local weather Forecast : 1,000,000+
Weather 14 Days : 10,000,000+
Weather by eltiempo.es : 5,000,000+
Storm Shield : 100,000+
GO Weather EX Theme White : 500,000+
MIUI Style GO Weather EX : 500,000+
Moonlight GO Weather EX : 1,000,000+
EZ Clock & Weather Widget : 1,000,000+
Florida Storms : 10,000+
Clearwater, FL - weather and more : 10+
St. Petersburg, FL - weather and more : 10+
WSVN • South Florida's Source for Weather : 1,000+
My Earthquake Alerts - US & Worldwide Earthquakes : 100,000+
FR Tides : 100,000+
```

Unlike other categories of Google Play Store, all the apps seem to have similar number of installations. Also, it is not dominated by few giant companies. This market shows potential.

In [61]:

```
for app in apple_final:
    genre = app[11]
    if genre == 'Weather':
        print(app[1], ':', app[5])
```

```
The Weather Channel: Forecast, Radar & Alerts : 495626
The Weather Channel App for iPad – best local forecast, radar map, and storm tracking : 208648
WeatherBug - Local Weather, Radar, Maps, Alerts : 188583
MyRadar NOAA Weather Radar Forecast : 150158
AccuWeather - Weather for Life : 144214
Yahoo Weather : 112603
Weather Underground: Custom Forecast & Local Radar : 49192
NOAA Weather Radar - Weather Forecast & HD Radar : 45696
Weather Live Free - Weather Forecast & Alerts : 35702
Storm Radar : 22792
QuakeFeed Earthquake Map, Alerts, and News : 6081
Moji Weather - Free Weather Forecast : 2333
Hurricane by American Red Cross : 1158
Forecast Bar : 375
Hurricane Tracker WESH 2 Orlando, Central Florida : 203
FEMA : 128
iWeather - World weather forecast : 80
Weather - Radar - Storm with Morecast App : 78
Yurekuru Call : 53
Weather & Radar : 37
WRAL Weather Alert : 25
Météo-France : 24
JaxReady : 22
Freddy the Frogcaster's Weather Station : 14
Almanac Long-Range Weather Forecast : 12
TodayAir : 0
wetter.com : 0
WarnWetter : 0
```

Similar to the Google Play Store, the number of rating seems well destributed in the Apple Store.

Apps in the 'Weather' category varies from weather forecast to natural disaster apps. Besides the normal weather forecast apps, people are looking for area specific weather apps as well. For example, 'Hurricane Tracker' and 'My Earthquake Alterts' targets certain areas in the U.S. that requires special weather forecast.

Conclusions In this project, we analyzed data about the App Store and Google Play mobile apps with the goal of recommending an app profile that can be profitable for both markets.

We concluded that making a weather app that is specialized to a certain area in U.S. could be profitable for both the Google Play Store and the Apple App Store markets. This category isn't full of Apps which gives some room to compete, so specializing the app to a certain area will be successful. This might include daily weather forecast, Earthquake, Hurricane, Storm, atmospheric particulate matter, etc.