**Heroes Of Pymoli - *A Fictional RPG Mobile Game***

Heroes Of Pymoli is a fictional RPG (Role Playing Game) Mobile Game. As in many RPG’s, the game is free for the user to play, and the players have the ability to purchase additional items to increase their game playing fun.

In our analysis, we will look at a set of synthesized stats (as though they had been scraped from this game) and attempt to glean information about the game and those who play it.

From our analysis, the following information can be found…

**Summary:**

In our analysis of 576 players stats, more than 4/5ths (484) of the players were **male** (84%) and less than 1/5th (81) were **female** (14%), with only 11 players not-identifying/other. These numbers vary greatly from the yearly ESA (Entertainment Software Association) reported numbers of 38-48% female participation rates (see <https://www.theesa.com/esa-research/2019-essential-facts-about-the-computer-and-video-game-industry/>).

It should be noted however, that if we look at By-Genre numbers, comparing this RPG’s numbers to other Western & Action RPGs, ours are much closer to the “norm” (see <https://quanticfoundry.com/2017/01/19/female-gamers-by-genre/>).

It is interesting that when it comes to in-game purchases, Gender doesn’t seem to matter for Dollars spent. We see that the Per-Person Average spent lines up with the above proportion of players (male/female/other 82.7 / 15.2 / 2.1). Likewise, the By-Gender per-person Total purchases also lines up with the proportion of players (male/female/other 83.6 / 14.5 / 1.9).

When it comes to the age of players, the majority (76.74%) of all players fall into that of 15-29 years old, with almost half of players (44.79%), falling into the 20-24 years old bracket. These middle bracket players are framed by the 15-19 (18.58%) and 25-29 (13.37%). Likewise, the dollars spent in-game seem to be independent of age, with only slight differences by age.

**Note**

* Instructions have been included for each segment. You do not have to follow them exactly, but they are included to help you think through the steps.

In [ ]:

*# Dependencies and Setup*

**import** pandas **as** pd

**import** numpy **as** np

​

*# File to Load (Remember to Change These)*

file\_to\_load **=** "Resources/purchase\_data.csv"

​

*# Read Purchasing File and store into Pandas data frame*

purchase\_data **=** pd.read\_csv(file\_to\_load)

purchase\_data.head(20)

​

**Player Count**

* Display the total number of players

In [ ]:

TotRows **=** purchase\_data["Price"].count()

print ("Total rows = ", TotRows)

print(" ")

TotPlayers **=** len(purchase\_data["SN"].value\_counts())

print ("Total Players = ", TotPlayers)

​

**Purchasing Analysis (Total)**

* Run basic calculations to obtain number of unique items, average price, etc.
* Create a summary data frame to hold the results
* Optional: give the displayed data cleaner formatting
* Display the summary data frame

In [3]:

UniqItms **=** len(purchase\_data["Item ID"].value\_counts())

AvePrc **=** purchase\_data["Price"].mean()

ItemsPurch **=** purchase\_data["Price"].count()

TotRev **=** purchase\_data["Price"].sum()

*#*

print (

f" UniqItms= {UniqItms} AvePrc= ${AvePrc:,.2f} ItemsPurch= {ItemsPurch} TotRev= ${TotRev:,.2f} "

)

*# Convert the summary data into a DataFrame*

SummaryData\_df **=** pd.DataFrame({"Number of Unique Items":[UniqItms],

"Average Price":[AvePrc],

"Number of Purchases":[ItemsPurch],

"Total Revenue":[TotRev]

})

SummaryData\_df.head()

​

UniqItms= 183 AvePrc= $3.05 ItemsPurch= 780 TotRev= $2,379.77

Out[3]:

|  | **Number of Unique Items** | **Average Price** | **Number of Purchases** | **Total Revenue** |
| --- | --- | --- | --- | --- |
| **0** | 183 | 3.050987 | 780 | 2379.77 |

**Gender Demographics**

* Percentage and Count of Male Players purchase\_data["Gender" = "Male"].count()
* Percentage and Count of Female Players
* Percentage and Count of Other / Non-Disclosed

In [4]:

CntMale **=** purchase\_data["Gender"].value\_counts()['Male']

CntFemale **=** purchase\_data["Gender"].value\_counts()['Female']

CntOther **=** purchase\_data["Gender"].value\_counts()['Other / Non-Disclosed']

CntTot **=** CntMale **+** CntFemale **+** CntOther

print(

f" Total: {CntTot}\n Male: {CntMale}\n Female: {CntFemale}\n Non\_specfic: {CntOther}")

print(" ")

PctMale **=** (CntMale **/** CntTot) **\*** 100

PctFemale **=** (CntFemale **/** CntTot) **\*** 100

PctOther **=** (CntOther **/** CntTot) **\*** 100

print(

f" % Male: {PctMale}\n % Female: {PctFemale}\n % Non\_specifc: {PctOther}")

Total: 780

Male: 652

Female: 113

Non\_specfic: 15

% Male: 83.58974358974359

% Female: 14.487179487179489

% Non\_specifc: 1.9230769230769231

**Purchasing Analysis (Gender)**

* Run basic calculations to obtain purchase count, avg. purchase price, avg. purchase total per person etc. by gender
* Create a summary data frame to hold the results
* Optional: give the displayed data cleaner formatting
* Display the summary data frame

In [5]:

OnlyMale **=** purchase\_data.loc[purchase\_data["Gender"] **==** "Male", :]

SumMale **=** OnlyMale["Price"].sum()

AveMale **=** OnlyMale["Price"].mean()

CntMaleSN **=** len(OnlyMale["SN"].unique())

AvePerMale **=** SumMale**/**CntMaleSN

​

print(

f" Male Total: {CntMale}\n Male AvePrice: ${AveMale:,.2f}\n Male TotPrice: ${SumMale:,.2f}\n Male AvePerPerson: ${AvePerMale:,.2f}")

print(

f" Male Persons: {CntMaleSN}")

print(" ")

OnlyFemale **=** purchase\_data.loc[purchase\_data["Gender"] **==** "Female", :]

SumFemale **=** OnlyFemale["Price"].sum()

AveFemale **=** OnlyFemale["Price"].mean()

CntFemaleSN **=** len(OnlyFemale["SN"].unique())

AvePerFemale **=** SumFemale**/**CntFemaleSN

​

print(

f" Female Total: {CntFemale}\n Female AvePrice: ${AveFemale:,.2f}\n Female TotPrice: ${SumFemale:,.2f}\n Female AvePerPerson: ${AvePerFemale:,.2f}")

print(

f" Female Persons: {CntFemaleSN}")

print(" ")

OnlyOth **=** purchase\_data.loc[purchase\_data["Gender"] **==** "Other / Non-Disclosed", :]

SumOth **=** OnlyOth["Price"].sum()

AveOth **=** OnlyOth["Price"].mean()

CntOthSN **=** len(OnlyOth["SN"].unique())

AvePerOth **=** SumOth**/**CntOthSN

​

print(

f" Other Total: {CntOther}\n Other AvePrice: ${AveOth:,.2f}\n Other TotPrice: ${SumOth:,.2f}\n Other AvePerPerson: ${AvePerOth:,.2f}")

print(

f" Other Persons: {CntOthSN}")

print(" ")

*# Create a new frame consolodating above calculations*

​

*# Creating a Pandas DataFrame by passing in a LIST OF DICTIONARIES*

*# Each value in the list is a dictionary*

*# Imagine that each dictionary represents a row of data in our eventual purchase count, avg. purchase price, avg. purchase total per persontaframe*

*# Each dictionary should have the same keys, since these keys dictate the column headers of our dataframe*

*# purchase count, avg. purchase price, avg. purchase total per person*

my\_list **=** [{"Gender": "Male", "Purchase Count": CntMale, "Average Price Paid": AveMale, "Total Purchase Value": SumMale, "Average Purchase PerPerson": AvePerMale},

{"Gender": "Female", "Purchase Count": CntFemale, "Average Price Paid": AveFemale, "Total Purchase Value": SumFemale, "Average Purchase PerPerson": AvePerFemale},

{"Gender": "Other / Non-Disclosed", "Purchase Count": CntOther, "Average Price Paid": AveOth, "Total Purchase Value": SumOth, "Average Purchase PerPerson": AvePerOth}]

ConsolPdDf **=** pd.DataFrame(my\_list)

ConsolPdDf

Male Total: 652

Male AvePrice: $3.02

Male TotPrice: $1,967.64

Male AvePerPerson: $4.07

Male Persons: 484

Female Total: 113

Female AvePrice: $3.20

Female TotPrice: $361.94

Female AvePerPerson: $4.47

Female Persons: 81

Other Total: 15

Other AvePrice: $3.35

Other TotPrice: $50.19

Other AvePerPerson: $4.56

Other Persons: 11

Out[5]:

|  | **Gender** | **Purchase Count** | **Average Price Paid** | **Total Purchase Value** | **Average Purchase PerPerson** |
| --- | --- | --- | --- | --- | --- |
| **0** | Male | 652 | 3.017853 | 1967.64 | 4.065372 |
| **1** | Female | 113 | 3.203009 | 361.94 | 4.468395 |
| **2** | Other / Non-Disclosed | 15 | 3.346000 | 50.19 | 4.562727 |

**Age Demographics**

* Establish bins for ages
* Categorize the existing players using the age bins. Hint: use pd.cut()
* Calculate the numbers and percentages by age group
* Create a summary data frame to hold the results
* Optional: round the percentage column to two decimal points
* Display Age Demographics Table

In [6]:

purchase\_data\_pd **=** pd.DataFrame(purchase\_data)

​

*# ----ByPrchr\_Data\_pd is all the data I need including the purchasers with multiple items---*

ByPrchr\_Data\_pd **=** purchase\_data\_pd.loc[:,['SN', 'Age', 'Item ID', 'Item Name', 'Price']]

*#*

ByPrchr\_Price\_df **=** ByPrchr\_Data\_pd.groupby(["SN"]).sum()["Price"]

​

*# ----ByPrchr\_Data\_pd is all the data I need by purchasers without the summed price data---*

ByPrchr\_Uniq\_pd **=** ByPrchr\_Data\_pd.drop\_duplicates(subset**=**'SN')

*# drop unneeded Price Column*

ByPrchr\_Uniq\_pd **=** ByPrchr\_Uniq\_pd.drop(columns**=**'Price')

​

*# ----------------sort the uniq's by SN then append...*

ByPrchr\_UniqSorted\_pd **=** ByPrchr\_Uniq\_pd.sort\_values(by**=**['SN'])

*# --------------------Merge two dataframes using an inner join*

*# merge\_table = pd.merge(\_pd, \_pd, on="")*

ByPrchr\_UniqSorted\_pd **=** pd.merge(ByPrchr\_UniqSorted\_pd, ByPrchr\_Price\_df, on**=**'SN')

In [7]:

*# Create bins and bin labels for the age column*

Aged\_df **=** pd.DataFrame(ByPrchr\_UniqSorted\_pd,columns**=**["Age", "SN", "Price"])

​

Age\_bins **=** [0, 9, 14, 19, 24, 29, 34, 39, 100]

Age\_labels **=** ["<10", "10-14", "15-19", "20-24", "25-29", "30-34", "35-39", "40+"]

In [8]:

*# Bin the Age column*

*# cut() returns a Pandas Series containing each of the binned column's values*

*# translated into their corresponding bins*

*# We can append our bins to Aged\_df*

Aged\_df["Level"] **=** pd.cut(Aged\_df["Age"], Age\_bins, labels**=**Age\_labels)

In [9]:

*# Group the Aged data by the Age Level we've created...*

​

*# group the purchasers by count of purchases*

grouped\_AgedByLvlCnt\_df **=** Aged\_df.groupby(["Level"]).count()['SN']

grouped\_AgedByLvlDol\_df **=** Aged\_df.groupby(["Level"]).sum()['Price']

grouped\_AgedByLvlCPct\_df **=** grouped\_AgedByLvlCnt\_df **/** TotPlayers

​

*# Merge the tables*

ByPrchr\_UniqSortedCompl\_pd **=** pd.merge(grouped\_AgedByLvlCnt\_df, grouped\_AgedByLvlCPct\_df, on**=**'Level')

​

*# rename the columns*

ByPrchrRen\_pd **=** ByPrchr\_UniqSortedCompl\_pd.rename(columns**=**{"SN\_x":"Total Count", "SN\_y":"Percentage of Players"})

ByPrchrRen\_pd.style.format({'Total Count': "{:.0f}",'Percentage of Players': "{:.2%}"})

Out[9]:

|  | **Total Count** | **Percentage of Players** |
| --- | --- | --- |
| **Level** |  |  |
| **<10** | 17 | 2.95% |
| **10-14** | 22 | 3.82% |
| **15-19** | 107 | 18.58% |
| **20-24** | 258 | 44.79% |
| **25-29** | 77 | 13.37% |
| **30-34** | 52 | 9.03% |
| **35-39** | 31 | 5.38% |
| **40+** | 12 | 2.08% |

**Purchasing Analysis (Age)**

* Bin the purchase\_data data frame by age
* Run basic calculations to obtain purchase count, avg. purchase price, avg. purchase total per person etc. in the table below
* Create a summary data frame to hold the results
* Optional: give the displayed data cleaner formatting
* Display the summary data frame

In [10]:

*# ----these values are accumulated by Age Level without regard to purchaser -----*

RawPurch\_pd **=** pd.DataFrame(purchase\_data)

RawAged\_data\_pd **=** RawPurch\_pd

RawAged\_data\_pd["Level"] **=** pd.cut(RawPurch\_pd["Age"], Age\_bins, labels**=**Age\_labels)

*# -------------------------------------------Purchase Count*

grouped\_RawAgedCnt\_pd **=** pd.DataFrame(RawAged\_data\_pd.groupby(["Level"]).count()['Item ID'])

grouped\_RawAgedCnt **=** RawAged\_data\_pd.groupby(["Level"]).count()['Item ID']

​

*# group the purchasers by age level ----------Total Purchase Value*

grouped\_RawAgedDol\_pd **=** pd.DataFrame(RawAged\_data\_pd.groupby(["Level"]).sum()['Price'])

grouped\_RawAgedDol **=** RawAged\_data\_pd.groupby(["Level"]).sum()['Price']

​

*# ---------------------------------------------Average Purchase Price*

grouped\_RawAgedAve\_pd **=** pd.DataFrame(grouped\_RawAgedDol **/** grouped\_RawAgedCnt)

​

*# ---------------------------------------------Avg Total Purchase per Person*

grouped\_AgedByLvlPrctg\_pd **=** pd.DataFrame(grouped\_AgedByLvlDol\_df **/** grouped\_AgedByLvlCnt\_df)

​

*# ---------------------------------Merge the values*

grouped\_RawCompl\_pd **=** pd.DataFrame() *#=========creates a new dataframe that's empty*

grouped\_RawCompl\_pd **=** pd.merge(grouped\_RawAgedCnt\_pd, grouped\_RawAgedAve\_pd, on**=**'Level')

grouped\_RawCompl\_pd **=** pd.merge(grouped\_RawCompl\_pd, grouped\_RawAgedDol\_pd, on**=**'Level')

grouped\_RawCompl\_pd **=** pd.merge(grouped\_RawCompl\_pd, grouped\_AgedByLvlPrctg\_pd, on**=**'Level')

*# ---------------------------- rename apply labels -------------------------*

grouped\_RawCompl\_pd **=** grouped\_RawCompl\_pd.rename(columns**=**{"Item ID":"Purchase Count"})

grouped\_RawCompl\_pd **=** grouped\_RawCompl\_pd.rename(columns**=**{"0\_x":"Average Purchase Price"})

grouped\_RawCompl\_pd **=** grouped\_RawCompl\_pd.rename(columns**=**{"Price":"Total Purchase Value"})

grouped\_RawCompl\_pd **=** grouped\_RawCompl\_pd.rename(columns**=**{"0\_y":"Avg Total Purchase per Person"})

*# ------------------------display in formated style for readability -----*

grouped\_RawCompl\_pd.style.format({'Average Purchase Price': "${:.2f}",

'Total Purchase Value': "${:.2f}",

'Avg Total Purchase per Person': "${:.2f}"

})

Out[10]:

|  | **Purchase Count** | **Average Purchase Price** | **Total Purchase Value** | **Avg Total Purchase per Person** |
| --- | --- | --- | --- | --- |
| **Level** |  |  |  |  |
| **<10** | 23 | $3.35 | $77.13 | $4.54 |
| **10-14** | 28 | $2.96 | $82.78 | $3.76 |
| **15-19** | 136 | $3.04 | $412.89 | $3.86 |
| **20-24** | 365 | $3.05 | $1114.06 | $4.32 |
| **25-29** | 101 | $2.90 | $293.00 | $3.81 |
| **30-34** | 73 | $2.93 | $214.00 | $4.12 |
| **35-39** | 41 | $3.60 | $147.67 | $4.76 |
| **40+** | 13 | $2.94 | $38.24 | $3.19 |

**Top Spenders**

* Run basic calculations to obtain the results in the table below
* Create a summary data frame to hold the results
* Sort the total purchase value column in descending order
* Optional: give the displayed data cleaner formatting
* Display a preview of the summary data frame

In [14]:

*# purchase\_data\_pd contains the correct info*

​

*# ---------- Use the group by to rollup the appropriate aggregations ----*

TopPurchCnt **=** purchase\_data\_pd.groupby(['SN']).count()['Price']

TopPurchAve **=** purchase\_data\_pd.groupby(['SN']).mean()['Price']

TopPurchTot **=** purchase\_data\_pd.groupby(['SN']).sum()['Price']

*# ---------- Pack the calculated value series into a dataframe and give them the correct labels----*

TopSpender\_pd **=** pd.DataFrame({

'Purchase Count': TopPurchCnt,

'Average Purchase Price': TopPurchAve,

'Total Purchase Value': TopPurchTot

})

*# ----get the (5) max by sorting in descending order and then use head() to chop only the first 5 ---*

MaxSpender\_pd **=** TopSpender\_pd.sort\_values(by**=**['Total Purchase Value'], ascending**=False**).head(5)

*# ------------------------display in formated style for readability -----*

MaxSpender\_pd.style.format({'Average Purchase Price': "${:.2f}",

'Total Purchase Value': "${:.2f}"

})

Out[14]:

|  | **Purchase Count** | **Average Purchase Price** | **Total Purchase Value** |
| --- | --- | --- | --- |
| **SN** |  |  |  |
| **Lisosia93** | 5 | $3.79 | $18.96 |
| **Idastidru52** | 4 | $3.86 | $15.45 |
| **Chamjask73** | 3 | $4.61 | $13.83 |
| **Iral74** | 4 | $3.40 | $13.62 |
| **Iskadarya95** | 3 | $4.37 | $13.10 |

In [ ]:

​

**Most Popular Items**

* Retrieve the Item ID, Item Name, and Item Price columns
* Group by Item ID and Item Name. Perform calculations to obtain purchase count, item price, and total purchase value
* Create a summary data frame to hold the results
* Sort the purchase count column in descending order
* Optional: give the displayed data cleaner formatting
* Display a preview of the summary data frame

In [18]:

*# purchase\_data\_pd contains the correct info*

*# ----PopData\_pd is all the data I need including the purchasers with multiple items---*

PopData\_pd **=** purchase\_data\_pd.loc[:,['Item ID', 'Item Name', 'Price']]

​

*# ---------- Use the group by to rollup the appropriate aggregations ----*

GrpPopDataCnt **=** PopData\_pd.groupby(['Item ID', 'Item Name']).count()['Price']

GrpPopDataTot **=** PopData\_pd.groupby(['Item ID', 'Item Name']).sum()['Price']

​

*# ----PopData\_pd is all the data I need by Item/Name ---*

GrpPopDataPrc **=** PopData\_pd.drop\_duplicates(subset**=**['Item ID', 'Item Name'])

PopDataPrc **=** GrpPopDataPrc.set\_index(['Item ID', 'Item Name'])

*# ----chg DataFrame to Series*

GrpPopDataP **=** pd.Series(PopDataPrc['Price'], index**=**PopDataPrc.index)

​

*# ---------- Pack the calculated value series into a dataframe and give them the correct labels----*

PopPurch\_pd **=** pd.DataFrame({

'Purchase Count': GrpPopDataCnt,

'Item Price': GrpPopDataP,

'Total Purchase Value': GrpPopDataTot

})

SrtPopPurch\_pd **=** PopPurch\_pd.sort\_values(by**=**['Purchase Count'], ascending**=False**).head(10)

*# ------------------------display in formated style for readability -----*

SrtPopPurch\_pd.style.format({'Item Price': "${:.2f}",

'Total Purchase Value': "${:.2f}"

})

Out[18]:

|  |  | **Purchase Count** | **Item Price** | **Total Purchase Value** |
| --- | --- | --- | --- | --- |
| **Item ID** | **Item Name** |  |  |  |
| **178** | **Oathbreaker, Last Hope of the Breaking Storm** | 12 | $4.23 | $50.76 |
| **145** | **Fiery Glass Crusader** | 9 | $4.58 | $41.22 |
| **108** | **Extraction, Quickblade Of Trembling Hands** | 9 | $3.53 | $31.77 |
| **82** | **Nirvana** | 9 | $4.90 | $44.10 |
| **19** | **Pursuit, Cudgel of Necromancy** | 8 | $1.02 | $8.16 |
| **103** | **Singed Scalpel** | 8 | $4.35 | $34.80 |
| **75** | **Brutality Ivory Warmace** | 8 | $2.42 | $19.36 |
| **72** | **Winter's Bite** | 8 | $3.77 | $30.16 |
| **60** | **Wolf** | 8 | $3.54 | $28.32 |
| **59** | **Lightning, Etcher of the King** | 8 | $4.23 | $33.84 |

**Most Profitable Items**

* Sort the above table by total purchase value in descending order
* Optional: give the displayed data cleaner formatting
* Display a preview of the data frame

In [20]:

*# ----get the (5) max by sorting in descending order and then use head() to chop only the first 5 ---*

Profitable\_pd **=** PopPurch\_pd.sort\_values(by**=**['Total Purchase Value'], ascending**=False**).head(10)

​

*# ------------------------display in formated style for readability -----*

Profitable\_pd.style.format({'Item Price': "${:.2f}",

'Total Purchase Value': "${:.2f}"

})

Out[20]:

|  |  | **Purchase Count** | **Item Price** | **Total Purchase Value** |
| --- | --- | --- | --- | --- |
| **Item ID** | **Item Name** |  |  |  |
| **178** | **Oathbreaker, Last Hope of the Breaking Storm** | 12 | $4.23 | $50.76 |
| **82** | **Nirvana** | 9 | $4.90 | $44.10 |
| **145** | **Fiery Glass Crusader** | 9 | $4.58 | $41.22 |
| **92** | **Final Critic** | 8 | $4.88 | $39.04 |
| **103** | **Singed Scalpel** | 8 | $4.35 | $34.80 |
| **59** | **Lightning, Etcher of the King** | 8 | $4.23 | $33.84 |
| **108** | **Extraction, Quickblade Of Trembling Hands** | 9 | $3.53 | $31.77 |
| **78** | **Glimmer, Ender of the Moon** | 7 | $4.40 | $30.80 |
| **72** | **Winter's Bite** | 8 | $3.77 | $30.16 |
| **60** | **Wolf** | 8 | $3.54 | $28.32 |

In [ ]:

​