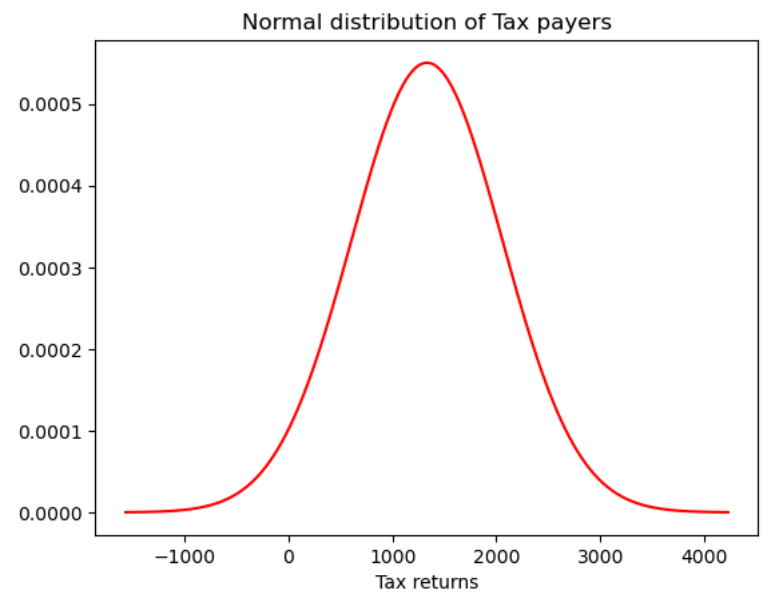
Importing necessary modules

In [49]:
```
from scipy.stats import zscore,norm,kurtosis,skew
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import statistics as st
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Tax payer problems

In [88]:
```
mean=1332
stdev=725
x1=2000
x2=0
x3=100
x4=700
zscore4=(x4-mean)/stdev
zscore3=(x3-mean)/stdev
zscore2=(x2-mean)/stdev
zscore1=(x1-mean)/stdev
p1=norm.cdf(zscore1)
p2=norm.cdf(zscore2)
p3=norm.cdf(zscore3)
p4=norm.cdf(zscore4)

#Task1:
print("Proportion of tax payers who pay above 2000$",round(1-p1,2))

#Task2:
print("Proportion of tax payers who don't recieve returns",round(p2,2))

#Task3:
print("Proportion of tax payers who recieve returns between 100 to 700$ ",round(p4-p3,2))

#Representing in norm graph
lower_n= mean-4*stdev
upper_n= mean+4*stdev

norm_n= np.arange(lower_n,upper_n)
plt.plot(norm_n, norm.pdf(norm_n, mean , stdev), color='red')
plt.title("Normal distribution of Tax payers")
plt.xlabel("Tax returns")
plt.show()
```

```
Proportion of tax payers who pay above 2000$ 0.18
Proportion of tax payers who don't recieve returns 0.03
Proportion of tax payers who recieve returns between 100 to 700$  0.15
```



High end video games

In [33]:
```
games=pd.read_csv(r"K:\Desktop\NIIT\tables\DS1_C5_S5_Computers_Data_Challenge.csv")
games1=games
games1
```

Out[33]:

	index	price	speed	hd	ram	screen	cd	multi	premium	ads	trend
0	1	1499	25	80	4	14	no	no	yes	94	1
1	2	1795	33	85	2	14	no	no	yes	94	1
2	3	1595	25	170	4	15	no	no	yes	94	1
3	4	1849	25	170	8	14	no	no	no	94	1
4	5	3295	33	340	16	14	no	no	yes	94	1
...	...	...	...	...	...	...	...	...	...	...	...
6254	6255	1690	100	528	8	15	no	no	yes	39	35
6255	6256	2223	66	850	16	15	yes	yes	yes	39	35
6256	6257	2654	100	1200	24	15	yes	no	yes	39	35
6257	6258	2195	100	850	16	15	yes	no	yes	39	35
6258	6259	2490	100	850	16	17	yes	no	yes	39	35

6259 rows × 11 columns

Task1: To sample data with device screen size with price less than 4000$

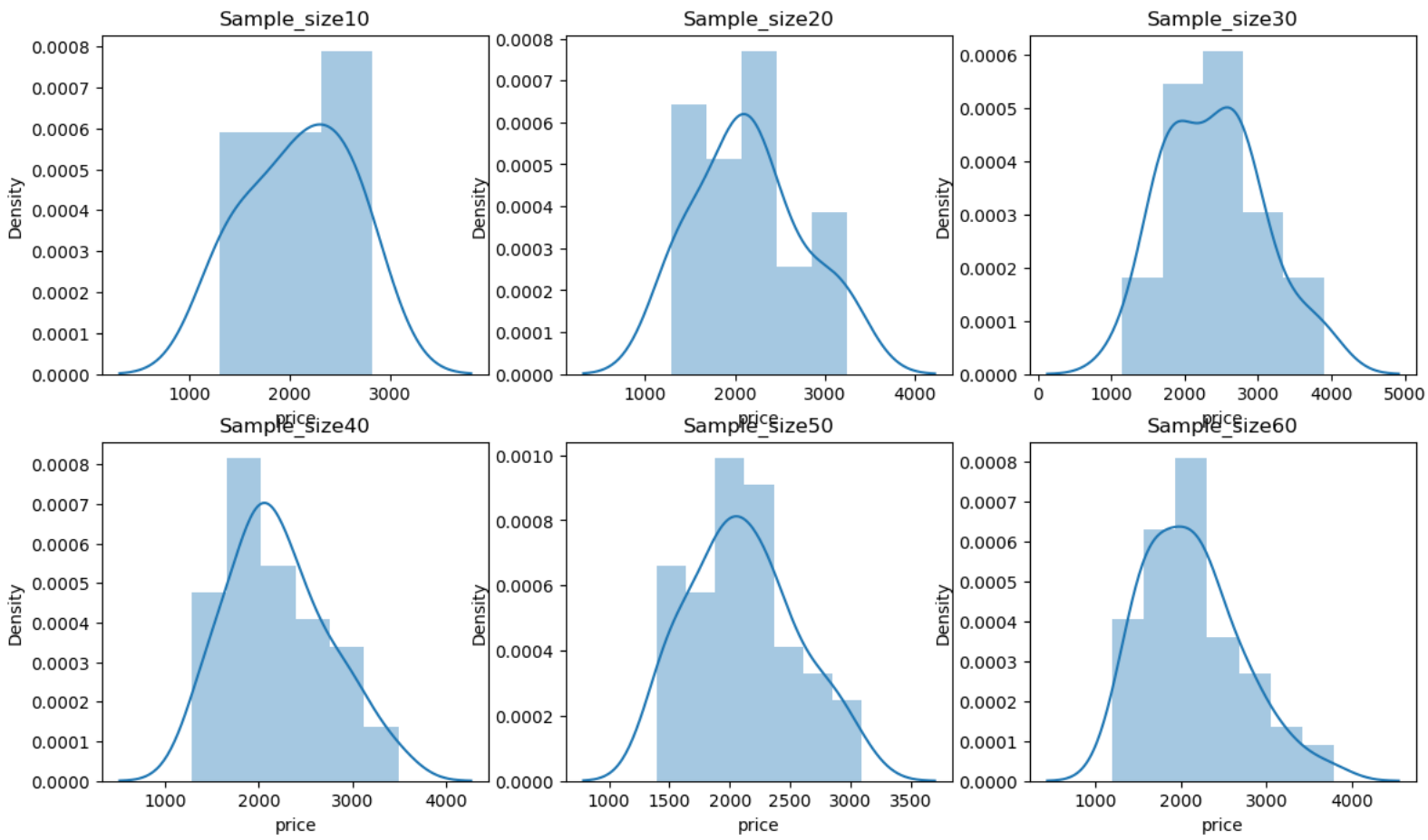
In [56]:
```
game_samp=games.price[games.price<4000].sample(50,ignore_index=True)
games=games[games.price<4000]
measures_game=pd.DataFrame([[game_samp.mean(),games.price.mean()],[game_samp.median(),games.price.median()],[st.mode(game_samp),st.mode(games.price)],[game_samp.std(),games.price.std()
                            [kurtosis(game_samp),kurtosis(games.price)],[skew(game_samp),skew(games.price)]],index=["Mean","Median","Mode","Standard dev","Kurtosis","Skewness"],columns=
```

Task 2 : To calculate measures of central tendency and show central limit thoerem with population data

Rule 1

In [71]:
```
print("As per central limit theorem's first rules we need to plot 6 different sample sizes to compare normal distribution")
fig,ax=plt.subplots(2,3,figsize=(14,8))
samps=[10,20,30,40,50,60]
j=0
i=0
for samp in samps:
    sns.distplot(games.price.sample(samp,ignore_index=True,replace=True),ax=ax[j,i])
    ax[j,i].set_title("Sample_size"+str(samp))
    i+=1
    if(i==3):
        i=0
        j=1


```

```
As per central limit theorem's first rules we need to plot 6 different sample sizes to compare normal distribution
```



Rule 2

In [57]:
```
print("As per central limit theorem 2 we can see the sample and population data have very close mean")
measures_game
```

```
As per central limit theorem 2 we can see the sample and population data have very close mean
```

Out[57]:

	Sample	Population
Mean	2276.320000	2208.855515
Median	2067.000000	2144.000000
Mode	1499.000000	1999.000000
Standard dev	710.293772	560.501267
Kurtosis	0.036968	-0.126395
Skewness	0.973392	0.518092

Rule 3

In [74]:
```
means=[]
for i  in range(1,21):
    means.append(games.price.sample(50,ignore_index=True,replace=True).mean())
samp_means_std=np.std(means)
pop_std=np.std(games.price)

print("The sample means std = {0} and population std by N sqrt = {1}".format(round(samp_means_std,2),round(pop_std/(50)**0.5),2))
```

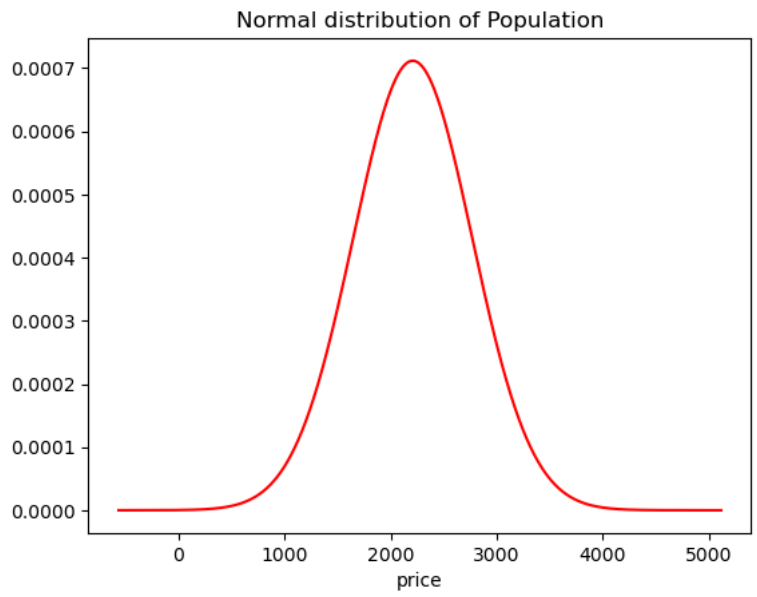
```
The sample means std = 77.59 and population std by N sqrt = 79
```

In [75]:
```
"""When comparing the sample means standard deviation with population's standard deviatio by sqrt of sample size we can see they are very close to each other
Thus the dataset follows central limit theorem"""
```

Out[75]:
```
"When comparing the sample means standard deviation with population's standard deviatio by sqrt of sample size we can see they are very close to each other\nThus the dataset follows centr
al limit theorem"
```

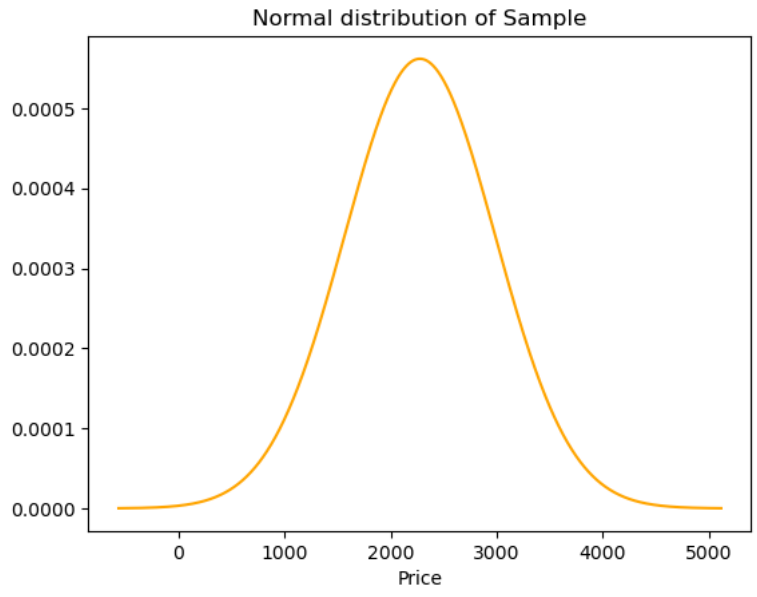
Task3 : To draw normal distribution plot for population

In [87]:
```
lower_p= games.price.mean()-4*games.price.std()
upper_p=  games.price.mean()+4*games.price.std()

norm_p= np.arange(lower_s,upper_s)
plt.plot(norm_p, norm.pdf(norm_p,games.price.mean(),games.price.std()), color='red')
plt.title("Normal distribution of Population")
plt.xlabel("price")
plt.show()
```



Task4: To draw normal distribution plot for sample

In [84]:
```
lower_s= game_samp.mean()-4*game_samp.std()
upper_s=  game_samp.mean()+4*game_samp.std()

norm_s= np.arange(lower_s,upper_s)
plt.plot(norm_s, norm.pdf(norm_s,game_samp.mean(),game_samp.std()), color='orange')
plt.title("Normal distribution of Sample")
plt.xlabel("Price")
plt.show()
```



Task5: To calculate zscores

In [107]:
```
price_z=zscore(game_samp)
probs=[]
for i in price_z:
    probs.append(norm.cdf(i))

price_score=pd.DataFrame()
price_score["Price"]=game_samp
price_score["Zscore"]=price_z
price_score["Probability"]=probs
price_score
```

Out[107]:

	Price	Zscore	Probability
0	3090	1.157184	0.876401
1	2685	0.581209	0.719450
2	1854	-0.600607	0.274051
3	1879	-0.565053	0.286019
4	1468	-1.149562	0.125162
5	1708	-0.808243	0.209475
6	3334	1.504192	0.933734
7	1839	-0.621940	0.266991
8	2425	0.211447	0.583731
9	1499	-1.105475	0.134477
10	3795	2.159808	0.984606
11	1679	-0.849486	0.197806
12	2575	0.424771	0.664498
13	2445	0.239890	0.594792
14	3799	2.165497	0.984825
15	2495	0.310998	0.622099
16	1799	-0.678826	0.248624
17	1644	-0.899261	0.184257
18	3595	1.875376	0.969629
19	2395	0.168782	0.567016
20	2444	0.238468	0.594241
21	2190	-0.122761	0.451148
22	1740	-0.762734	0.222811
23	2404	0.181582	0.572044
24	1395	-1.253379	0.105034
25	1873	-0.573586	0.283124
26	2299	0.032255	0.512866
27	1595	-0.968947	0.166286
28	1395	-1.253379	0.105034
29	1989	-0.408615	0.341411
30	2145	-0.186758	0.425925
31	2495	0.310998	0.622099
32	3995	2.444241	0.992742
33	1790	-0.691626	0.244586
34	2644	0.522900	0.699478
35	2925	0.922528	0.821873
36	3904	2.314824	0.989689
37	1899	-0.536610	0.295769
38	2390	0.161671	0.564218
39	1894	-0.543721	0.293317
40	1595	-0.968947	0.166286
41	1799	-0.678826	0.248624
42	1499	-1.105475	0.134477
43	2540	0.374996	0.646168
44	1699	-0.821042	0.205811
45	2399	0.174471	0.569252
46	1740	-0.762734	0.222811
47	3440	1.654941	0.951032
48	1728	-0.779800	0.217754
49	1970	-0.435637	0.331550

Task9: To find the probability of getting less than 1301

In [113]:
```
X=(1301-game_samp.mean())/game_samp.std()

print("The probability of getting less than 1301 is ",1-norm.cdf(X))
```

```
The probability of getting less than 1301 is  0.9151427875687974
```

Task10: To find the probability of occurences 2000 and 2900

In [114]:
```
X1=(2000-game_samp.mean())/game_samp.std()
X2=(2900-game_samp.mean())/game_samp.std()
print("The probability of getting between 2000 and 2900 is ",norm.cdf(X2)-norm.cdf(X1))
```

```
The probability of getting between 2000 and 2900 is  0.46141432485830275
```