In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:

```
df = pd.read_csv('Ads_Optimisation.csv')
df
```

Out[2]:

	Ad 1	Ad 2	Ad 3	Ad 4	Ad 5	Ad 6	Ad 7	Ad 8	Ad 9	Ad 10
0	1	0	0	0	1	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
9995	0	0	1	0	0	0	0	1	0	0
9996	0	0	0	0	0	0	0	0	0	0
9997	0	0	0	0	0	0	0	0	0	0
9998	1	0	0	0	0	0	0	1	0	0
9999	0	1	0	0	0	0	0	0	0	0

10000 rows × 10 columns

In [11]:

```
import random
N = 10000
d = 10
ads_selected = []
total_reward= 0
for n in range(0, N):
    ad = random.randrange(d)
    ads_selected.append(ad)
    reward = df.values[n, ad]
    total_reward = total_reward + reward
print(total_reward)
```

```
1256
```

In [15]:

```
pd.Series(ads_selected).tail(1000).value_counts(normalize=True)
```

Out[15]:

```
7    0.110
2    0.106
1    0.105
5    0.104
9    0.103
6    0.099
0    0.098
8    0.095
3    0.091
4    0.089
dtype: float64
```

In [22]:

```
#Implementing UCB
import math
N = 10000
d = 10
ads_selected = []
number_of_selections = [0]*d
sum_of_reward = [0]*d
total_reward = 0
```

In [25]:

```
for n in range(10000):
    ad = 0
    max_upper_bound = 0
    for i in range(0, d):
        if(number_of_selections[i] > 0):
            average_reward = sum_of_reward[i] / number_of_selections[i]
            delta_i = math.sqrt(2*math.log(n+1) / number_of_selections[i])
            upper_bound = average_reward + delta_i
        else:
            upper_bound = 1e400
        if upper_bound > max_upper_bound:
            max_upper_bound = upper_bound
            ad = i
    ads_selected.append(ad)
    number_of_selections[ad] += 1
    reward = df.values[n, ad]
    sum_of_reward[ad] += reward
    total_reward += reward
print(total_reward)
```

```
2387
```

In [26]:

```
pd.Series(ads_selected).tail(1000).value_counts(normalize=True)
```

Out[26]:

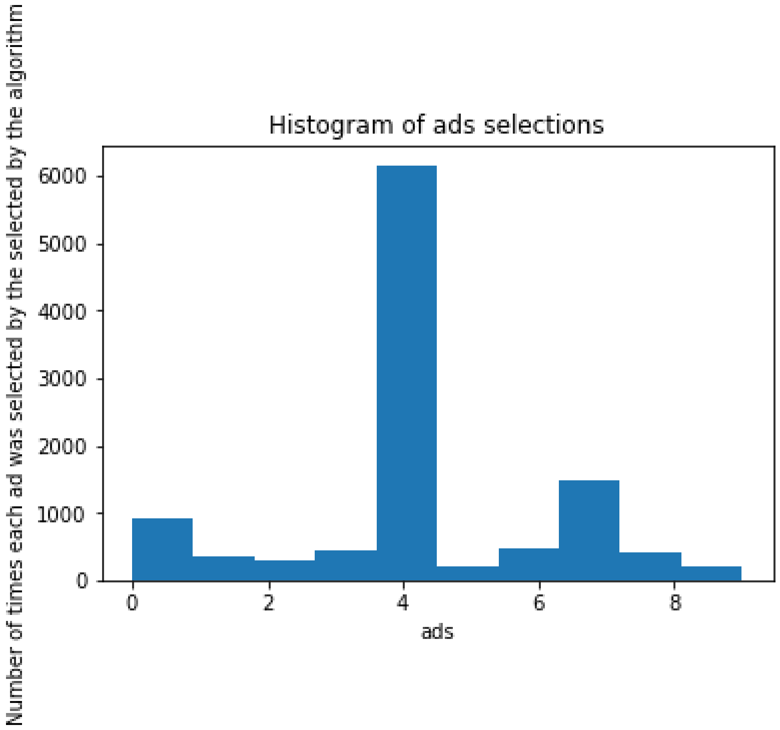
```
4    0.791
7    0.073
1    0.040
3    0.026
2    0.022
0    0.019
8    0.009
6    0.009
5    0.006
9    0.005
dtype: float64
```

In [27]:

```
plt.hist(ads_selected)
plt.title('Histogram of ads selections')
plt.xlabel('ads')
plt.ylabel('Number of times each ad was selected by the selected by the algorithm')
```

Out[27]:

```
Text(0, 0.5, 'Number of times each ad was selected by the selected by the algorithm')
```


Histogram of ads selections
Number of times each ad was selected by the selected by the algorithm
ads
0
1000
2000
3000
4000
5000
6000
0
2
4
6
8

```
In [ ]:
```