Imports

In [46]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.ticker as tcr
%matplotlib inline
pd.set_option('display.max_rows', None)
```

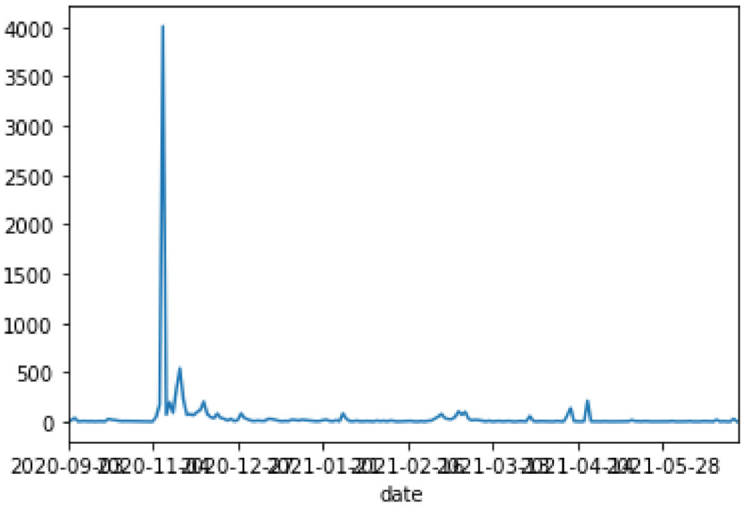
In [47]:
```
best_bet = pd.read_csv('/home/slackroo/JDS/data_practice/bestbet_all/all_data.csv',deli
```

Games Played / Day

In [48]:
```
x=best_bet.groupby(['date','game_id']).count()
daily_rounds = x.groupby(['date']).count()['round_id']
```

In [49]:
```
daily_rounds.plot()
```

Out[49]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3f58b08dc0>
```



In [50]:
```
daily_mean =daily_rounds.mean()
daily_median= daily_rounds.median()
```

In [51]:
```
print( "%.2f"%daily_mean, daily_median)
```

```
45.01 5.0
```

Games that ended to 0

In [52]:
```
zero_stack = best_bet[best_bet.new_stack == 0]
```

```
check = zero_stack.groupby(['date','game_id']).count()['new_stack']
zero_counts_perday = check.groupby(['date']).count().reset_index()
show=zero_counts_perday.new_stack.sum()
print("total games that ended with zero",show )
```

```
total games that ended with zero 1273
```

Distribution of scores

In [53]:

```
scores_df = best_bet.query('round_id == 50 and new_stack > 0')
stacks = scores_df[['date','new_stack']].reset_index(drop=True)
stacks['new_stack']= stacks['new_stack']
stacks_1 = scores_df[['new_stack']].reset_index(drop=True)
```

In [54]:

```
bins = [0,10,100,1000,10000,100000,1000000,10000000,100000000,1000000000,10000000000]
stacks['binned'] = pd.cut(stacks['new_stack'], bins)
bin_count = stacks.groupby(pd.cut(stacks['new_stack'], bins=bins)).size()
bin_count_2 = pd.cut(stacks['new_stack'], bins=bins).value_counts()
print (bin_count)
```

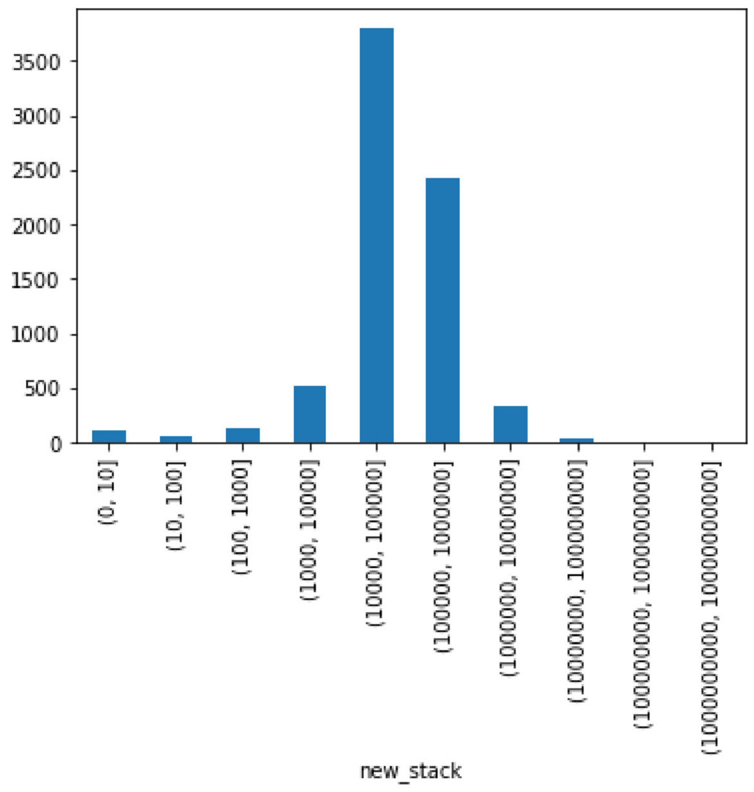
```
new_stack
(0, 10]                        112
(10, 100]                       54
(100, 1000]                    127
(1000, 10000]                  518
(10000, 100000]               3800
(100000, 1000000]             2422
(1000000, 10000000]            333
(10000000, 100000000]           41
(100000000, 1000000000]          8
(1000000000, 10000000000]        3
dtype: int64
```

In [55]:

```
bin_count.plot(kind='bar')
```

Out[55]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3f589c9d90>
```



Average mean score

```
In [ ]:
stacks
```

```
In [57]:
mean_score = stacks['new_stack'].mean()
mean_score
```

```
Out[57]: 65214483.830031
```

```
In [58]:
count_winnings = stacks.new_stack.count()
count_winnings
```

```
Out[58]: 7419
```

```
In [ ]:
above_avg =stacks[stacks.new_stack < mean_score ].count()
above_avg
```

```
In [ ]:
below_avg = stacks[stacks.new_stack > mean_score ].count()
below_avg
```

Average stack per round on won games

```
In [61]:
only_wins=best_bet.loc[(best_bet["round_id"]== 50) & (best_bet["new_stack"] > 0) ]
only_wins = only_wins.reset_index(drop = True)
```

```
only_wins.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7419 entries, 0 to 7418
Data columns (total 10 columns):
date                 7419 non-null object
game_id              7419 non-null object
round_id             7419 non-null int64
risked_money         7419 non-null int64
expected_value       7419 non-null float64
multiplier           7419 non-null float64
chance_of_winning    7419 non-null int64
did_they _win        7419 non-null bool
how_much             7419 non-null int64
new_stack            7419 non-null int64
dtypes: bool(1), float64(2), int64(5), object(2)
memory usage: 529.0+ KB
```

In [62]:
```
all_games_won = best_bet.loc[best_bet['game_id'].isin(only_wins['game_id'])]
```

In [63]:
```
pd.options.display.float_format = '{:.2f}'.format
all_games_won.groupby(['round_id']). mean().reset_index()[['round_id','risked_money','n
```

Out[63]:

	round_id	risked_money	new_stack
0	1	2190.78	13289.34
1	2	2392.67	15429.79
2	3	2686.32	17416.81
3	4	3169.98	19863.12
4	5	3558.36	21769.11
5	6	3313.60	24123.57
6	7	3763.44	26732.16
7	8	4331.54	28979.87
8	9	5059.56	31882.18
9	10	5233.28	33567.75
10	11	5266.50	36892.98
11	12	6113.46	40872.82
12	13	7459.05	44462.46
13	14	6939.66	47465.46
14	15	7774.73	52973.94
15	16	8705.96	56215.01
16	17	7354.25	59104.35
17	18	9182.18	63521.30
18	19	9287.09	66854.98

	round_id	risked_money	new_stack
19	20	8752.20	70853.79
20	21	11000.01	77980.85
21	22	14778.73	85968.04
22	23	14730.85	90625.14
23	24	13691.54	97094.00
24	25	14518.92	105883.29
25	26	20587.30	123877.53
26	27	16517.95	129493.87
27	28	19118.40	139794.78
28	29	32352.77	149808.16
29	30	36958.09	192994.87
30	31	20958.42	203929.44
31	32	22887.29	210658.30
32	33	31673.95	232562.33
33	34	28852.71	267035.26
34	35	90240.77	334383.57
35	36	120136.09	486949.52
36	37	50590.55	492175.68
37	38	276656.52	1335914.38
38	39	63675.90	1361719.47
39	40	39589.04	1402429.74
40	41	54223.60	1480053.62
41	42	1127067.13	3327745.45
42	43	74032.27	3367872.74
43	44	115330.27	3430924.33
44	45	2943052.14	8039770.59
45	46	7560426.27	18493598.60
46	47	17920583.69	18567921.62
47	48	17888700.76	34518506.28
48	49	33994761.96	65074264.62
49	50	289395.27	65214483.83

In [64]:

```
won_stats = all_games_won.groupby(['round_id']).agg({'new_stack':['mean','std'],'risked
```

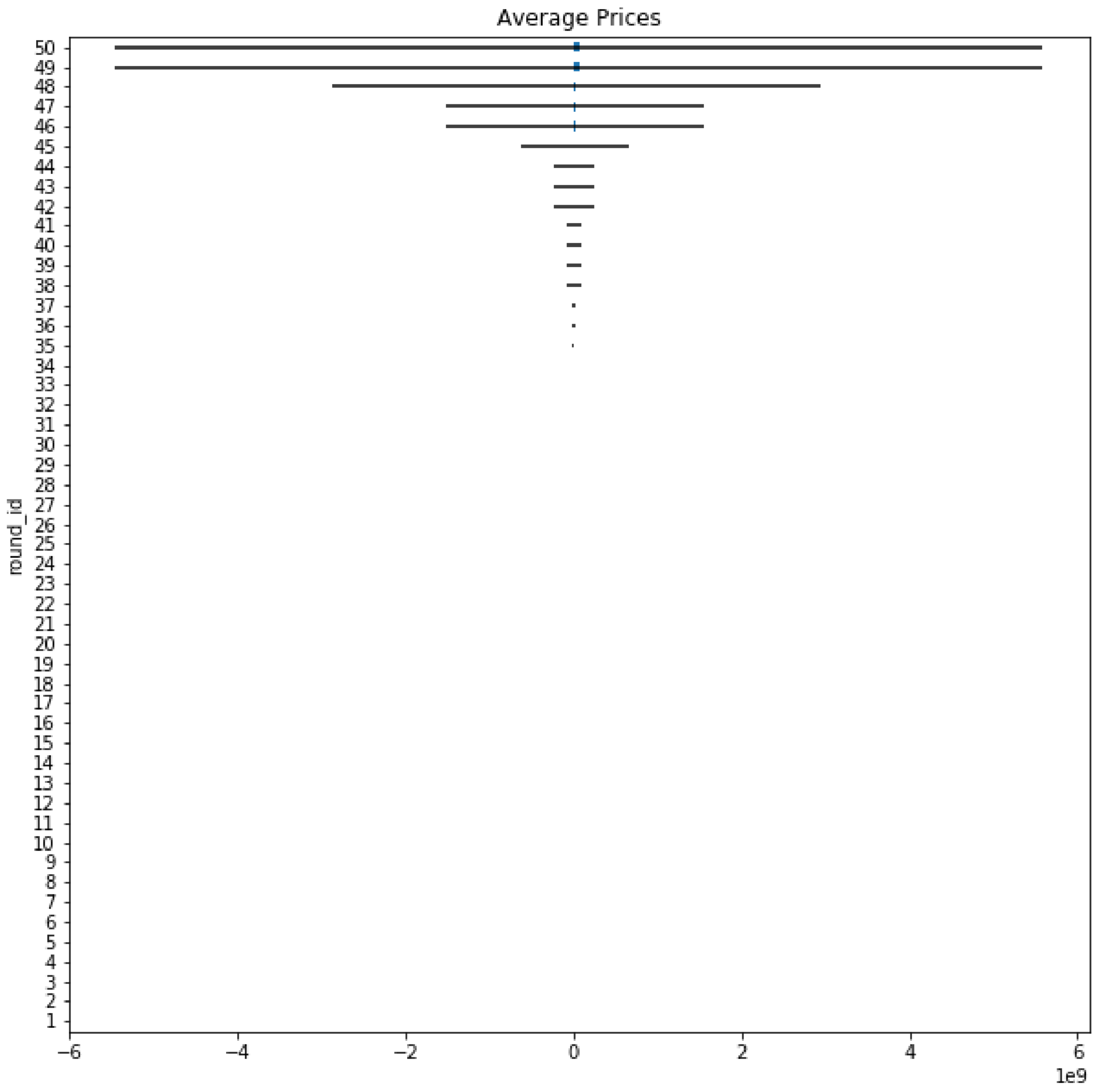
In [65]:
```
new_stack_stats = all_games_won.groupby(['round_id']).agg(['mean','std'])['new_stack']
```

In [ ]:
```
new_stack_stats
```

In [ ]:
```
new_stack_stats.describe()
```

In [66]:
```
new_stack_stats.plot(kind = "barh",figsize = (10,10) ,y = "mean", legend = False,
            title = "Average Prices", xerr = "std")
```

Out[66]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3f588d90a0>
```



In [ ]:
```

```