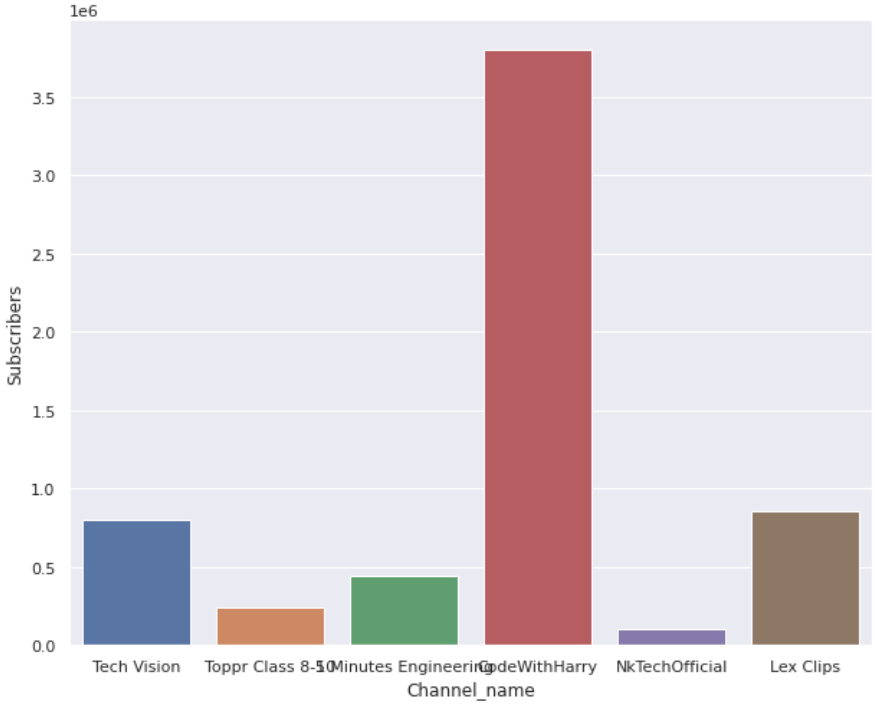
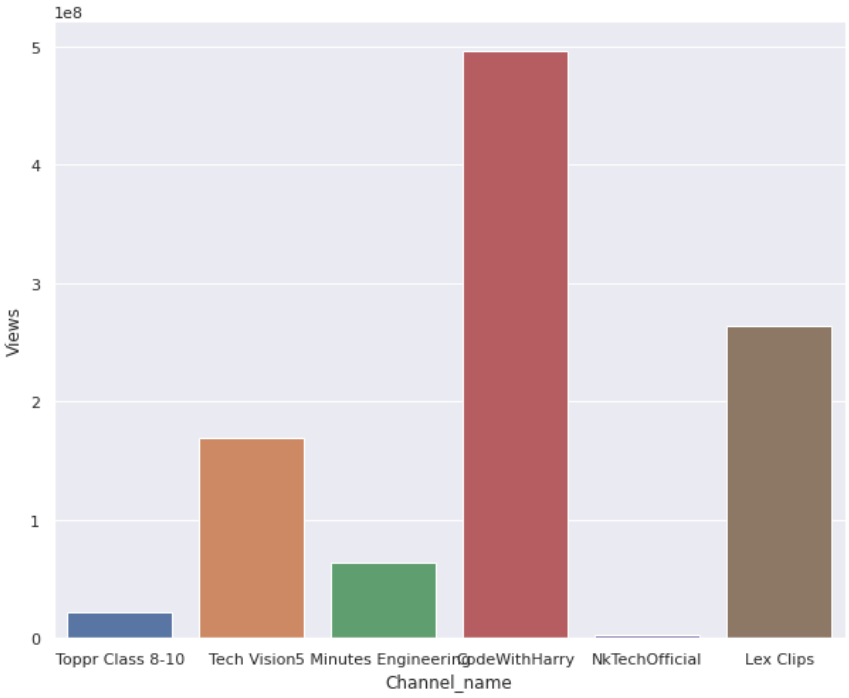
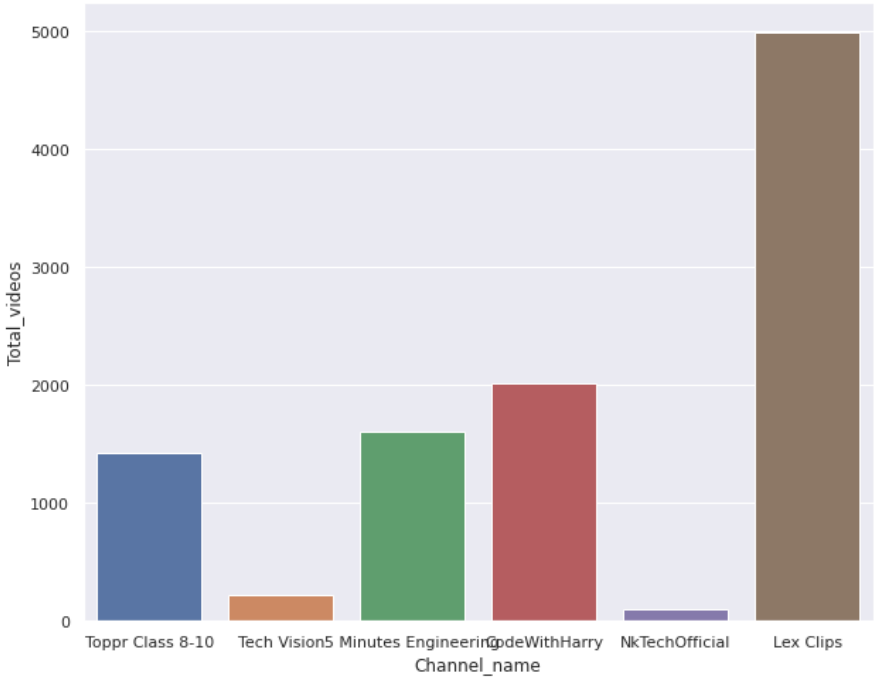
```
channel_data['Subscribers'] = pd.to_numeric(channel_data['Subscribers'])
channel_data['Views'] = pd.to_numeric(channel_data['Views'])
channel_data['Total_videos'] = pd.to_numeric(channel_data['Total_videos'])
channel_data.dtypes
sns.set(rc={'figure.figsize':(10,8)})
ax = sns.barplot(x='Channel_name', y='Subscribers', data=channel_data)
```




```
ax = sns.barplot(x='Channel_name', y='Views', data=channel_data)
```



```
ax = sns.barplot(x='Channel_name', y='Total_videos', data=channel_data)
```




```
playlist_id = channel_data.loc[channel_data['Channel_name']=='5 Minutes Engineering',
'playlist_id'].iloc[0]
def get_video_ids(youtube, playlist_id):
    request = youtube.playlistItems().list(
                part='contentDetails',
                playlistId = playlist_id,
                maxResults = 50)
    response = request.execute()
    video_ids = []
    for i in range(len(response['items'])):
        video_ids.append(response['items'][i]['contentDetails']['videoId'])
    next_page_token = response.get('nextPageToken')
    more_pages = True
    while more_pages:
        if next_page_token is None:
            more_pages = False
        else:
            request = youtube.playlistItems().list(
                        part='contentDetails',
                        playlistId = playlist_id,
                        maxResults = 50,
                        pageToken = next_page_token)
            response = request.execute()
            for i in range(len(response['items'])):
                video_ids.append(response['items'][i]['contentDetails']['videoId'])
            next_page_token = response.get('nextPageToken')
    return video_ids
```

```
video_ids = get_video_ids(youtube, playlist_id)
print("For channel YouTube 5 Minutes Engineering we have over ",len(video_ids)," video
ids")
For channel YouTube 5 Minutes Engineering we have over  1605  video ids
def get_video_details(youtube, video_ids):
    all_video_stats = []
    for i in range(0, len(video_ids), 50):
        request = youtube.videos().list(
                    part='snippet,statistics',
                    id=','.join(video_ids[i:i+50]))
        response = request.execute()
        for video in response['items']:
            video_stats = dict(Title = video['snippet']['title'],
                               Published_date = video['snippet']['publishedAt'],
                               Views = video['statistics']['viewCount'],
                               Likes = video['statistics']['likeCount'],
                               Comments = video['statistics']['commentCount'],
                               favourites = video['statistics']['favoriteCount']
                               )
            all_video_stats.append(video_stats)
    return all_video_stats
video_details = get_video_details(youtube, video_ids)
video_data = pd.DataFrame(video_details)

video_data['Month'] = pd.to_datetime(video_data['Published_date']).dt.strftime('%b')
videos_per_month = video_data.groupby('Month', as_index=False).size()
sort_order = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun','Jul', 'Aug', 'Sep', 'Oct',
'Nov', 'Dec']
videos_per_month.index = pd.CategoricalIndex(videos_per_month['Month'],
categories=sort_order, ordered=True)
videos_per_month = videos_per_month.sort_index()
```

```
videos_per_month
```

```
ax2 = sns.barplot(x='Month', y='size', data=videos_per_month)
```

Month	Month	size
Jan	Jan	92
Feb	Feb	204
Mar	Mar	194
Apr	Apr	120
May	May	116
Jun	Jun	99
Jul	Jul	140
Aug	Aug	128
Sep	Sep	125
Oct	Oct	125
Nov	Nov	146
Dec	Dec	116