3-Finding Answers

Importing Necessary Libraries

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.interpolate as interpolate
import _pickle as pickle
```

Function to Sort Dictonary

In [2]:

```
def sort_dict(dct,parameter,order='Ascending'):
    if parameter=='key':
        if(order=='reverse'):
            sorted_tuples = sorted(dct.items(),key=lambda item:item[0],reverse=True)
            return {k:v for k,v in sorted_tuples}
        else:
            sorted_tuples = sorted(dct.items(),key=lambda item:item[0])
            return {k:v for k,v in sorted_tuples}
    else:
        if(order=='reverse'):
            sorted_tuples = sorted(dct.items(),key=lambda item:item[1],reverse=True)
            return {k:v for k,v in sorted_tuples}
        else:
            sorted_tuples = sorted(dct.items(),key=lambda item:item[1])
            return {k:v for k,v in sorted_tuples}
```

Importing Database

In [3]:

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

In [4]:

```
df.head()
```

Out[4]:

	Unnamed: 0	Authors	Title	Year	Cited by	Country	Funding_Details
0	0	Soares, J.V.B., Leandro, J.J.G., Cesar Jr., R....	Retinal vessel segmentation using the 2-D Gabo...	2006	1083.0	Australia	0
1	1	Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuch...	The graph neural network model	2009	1031.0	Australia	0
2	2	Karantonis, D.M., Narayanan, M.R., Mathie, M.,...	Implementation of a real-time human movement c...	2006	908.0	Australia	0
3	3	Mirjalili, S.	Dragonfly algorithm: a new meta-heuristic opti...	2016	865.0	Australia	0
4	4	Naseem, I., Togneri, R., Bennamoun, M.	Linear regression for face recognition	2010	768.0	Australia	0

In [5]:

```
df.shape
```

Out[5]:

```
(67694, 7)
```

Removing Unnamed column and renaming Cited by column for ease of use

In [6]:

```
df.drop('Unnamed: 0',axis='columns',inplace=True)
```

In [7]:

```
df.rename(columns={'Cited by':'Cited_by','Funding Details':'Funding_Details'},inplace
=True)
```

Reading Authors names from previously created file and storing those in python list

In [8]:

```
set_authors = []
with open('Authors_list.txt','r') as filehandle:
    filecontents = filehandle.readlines()

    for line in filecontents:
        # remove linebreak which is the last character of the string
        author = line[:-1]

        # add item to the list
        set_authors.append(author)
filehandle.close()
```

Reading Previously Created Author:Database Dictonary

In [9]:

```
dct_author_database = {}
with open('Author_database_dictonary.txt','rb') as file:
    dct_author_database = pickle.load(file)
file.close()
```

In [10]:

```
index = 6
print(set_authors[index])
dct_author_database[set_authors[index]]
```

```
Jelinek, H.F.
```

Out[10]:

	Authors	Title	Year	Cited_by	Funding Details
0	Soares, J.V.B., Leandro, J.J.G., Cesar Jr., R....	Retinal vessel segmentation using the 2-D Gabo...	2006	1083.0	0
1	Rocha, A., Carvalho, T., Jelinek, H.F., Golden...	Points of interest and visual dictionaries for...	2012	96.0	0
2	Jelinek, H.F., Cree, M.J., Leandro, J.J.G., So...	Automated segmentation of retinal blood vessel...	2007	47.0	0
3	Hassan, M.M., Huda, S., Yearwood, J., Jelinek,...	Multistage fusion approaches based on a genera...	2018	18.0	0
4	Abawajy, J., Kelarev, A., Chowdhury, M., Stran...	Predicting cardiac autonomic neuropathy catego...	2013	15.0	0

a) Highest cited author and his h-index (from the world)

In [11]:

```
author_with_highest_citations = ""
max_citations = 0
for author in set_authors:
    cites = dct_author_database[author]['Cited_by'].sum()
    if max_citations<cites:
        author_with_highest_citations = author
        max_citations = cites
```

In [12]:

```
df_of_highest_cited_author = dct_author_database[author_with_highest_citations]
rows,cols = df_of_highest_cited_author.shape
avg_citations_of_author_with_highest_citations = max_citations/rows
h_index = min(rows,avg_citations_of_author_with_highest_citations)
```

```
In [13]: print(f'Max cited author    = {author_with_highest_citations}')
         print(f'Total cited by      = {max_citations}')
         print(f'His h-index         = {h_index}')
```

```
Max cited author    = Hassabis, D.
Total cited by      = 17466.0
His h-index         = 13
```

b) Highest publication author

```
In [14]: author_with_highest_publication = ""
         max_publication_count = 0
         for author in set_authors:
             rows, columns = dct_author_database[author].shape
             if rows>max_publication_count:
                 max_publication_count=rows
                 author_with_highest_publication = author
                 # print(f'{author}                    \t{rows}')
```

```
In [15]: print(f'Author\t\t= {author_with_highest_publication}\nPublications\t= {max_publicati
         on_count}')
```

```
Author          = Wang, Y.
Publications    = 439
```

c) Highest cited authors avg. citations, and the country name.

```
In [16]: dct_author_database[author_with_highest_citations]
```

Out[16]:

	Authors	Title	Year	Cited_by	Funding Details
0	Vinyals, O., Babuschkin, I., Czarnecki, W.M., ...	Grandmaster level in StarCraft II using multi-...	2019	224.0	0
1	Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A...	Human-level control through deep reinforcement...	2015	7346.0	0
2	Silver, D., Huang, A., Maddison, C.J., Guez, A...	Mastering the game of Go with deep neural netw...	2016	5282.0	0
3	Silver, D., Schrittwieser, J., Simonyan, K., A...	Mastering the game of Go without human knowledge	2017	2391.0	0
4	Kirkpatrick, J., Pascanu, R., Rabinowitz, N., ...	Overcoming catastrophic forgetting in neural n...	2017	647.0	0
5	De Fauw, J., Ledsam, J.R., Romera-Paredes, B.,...	Clinically applicable deep learning for diagno...	2018	602.0	0
6	Silver, D., Hubert, T., Schrittwieser, J., Ant...	A general reinforcement learning algorithm tha...	2018	457.0	0
7	McKinney, S.M., Sieniek, M., Godbole, V., Godw...	International evaluation of an AI system for b...	2020	266.0	NIHR Imperial Biomedical Research Centre\n\nOf...
8	Wang, J.X., Kurth-Nelson, Z., Kumaran, D., Tir...	Prefrontal cortex as a meta-reinforcement lear...	2018	112.0	0
9	Jaderberg, M., Czarnecki, W.M., Dunning, I., M...	Human-level performance in 3D multiplayer game...	2019	71.0	0
10	Dabney, W., Kurth-Nelson, Z., Uchida, N., Star...	A distributional code for value in dopamine-ba...	2020	43.0	0
11	Yim, J., Chopra, R., Spitz, T., Winkens, J., O...	Predicting conversion to wet age-related macul...	2020	23.0	0
12	Schrittwieser, J., Antonoglou, I., Hubert, T.,...	Mastering Atari, Go, chess and shogi by planni...	2020	2.0	0

In [17]:
```
rows, cols = dct_author_database[author_with_highest_citations].shape

avg_citations_of_author_with_highest_citations = max_citations/rows
print(f'Highest Cited Author \t= {author_with_highest_citations}')
print(f'His Average Citations \t= {avg_citations_of_author_with_highest_citations}')
print(f'His Total Publications \t= {rows}')
```

```
Highest Cited Author    = Hassabis, D.
His Average Citations   = 1343.5384615384614
His Total Publications  = 13
```

A google search with above author's name tells us that he's from 'United Kingdom' and doing reasearch on 'Artificial Intelligence'

d) Total number of publications of the highest cited author

In [18]:
```
print(f'Highest Cited Author \t= {author_with_highest_citations}')
print(f'Total Publications \t= {rows}')
dct_author_database[author_with_highest_citations]
```

```
Highest Cited Author    = Hassabis, D.
Total Publications      = 13
```

Out[18]:

	Authors	Title	Year	Cited_by	Funding Details
0	Vinyals, O., Babuschkin, I., Czarnecki, W.M., ...	Grandmaster level in StarCraft II using multi-...	2019	224.0	0
1	Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A...	Human-level control through deep reinforcement...	2015	7346.0	0
2	Silver, D., Huang, A., Maddison, C.J., Guez, A...	Mastering the game of Go with deep neural netw...	2016	5282.0	0
3	Silver, D., Schrittwieser, J., Simonyan, K., A...	Mastering the game of Go without human knowledge	2017	2391.0	0
4	Kirkpatrick, J., Pascanu, R., Rabinowitz, N., ...	Overcoming catastrophic forgetting in neural n...	2017	647.0	0
5	De Fauw, J., Ledsam, J.R., Romera-Paredes, B.,...	Clinically applicable deep learning for diagno...	2018	602.0	0
6	Silver, D., Hubert, T., Schrittwieser, J., Ant...	A general reinforcement learning algorithm tha...	2018	457.0	0
7	McKinney, S.M., Sieniek, M., Godbole, V., Godw...	International evaluation of an AI system for b...	2020	266.0	NIHR Imperial Biomedical Research Centre\n\nOf...
8	Wang, J.X., Kurth-Nelson, Z., Kumaran, D., Tir...	Prefrontal cortex as a meta-reinforcement lear...	2018	112.0	0
9	Jaderberg, M., Czarnecki, W.M., Dunning, I., M...	Human-level performance in 3D multiplayer game...	2019	71.0	0
10	Dabney, W., Kurth-Nelson, Z., Uchida, N., Star...	A distributional code for value in dopamine-ba...	2020	43.0	0
11	Yim, J., Chopra, R., Spitz, T., Winkens, J., O...	Predicting conversion to wet age-related macul...	2020	23.0	0
12	Schrittwieser, J., Antonoglou, I., Hubert, T.,...	Mastering Atari, Go, chess and shogi by planni...	2020	2.0	0

e) Total publication in year

In [19]:
```
year_lst = sorted(list(df['Year'].unique()))
country_lst = list(df['Country'].unique())
```

```
In [20]: df_without_duplicates = df.drop_duplicates(subset=['Authors','Title'],keep='first')
```

```
In [21]: dct_df_per_year_publications = {}
         for year in year_lst:
             dct_df_per_year_publications[year], cols = df_without_duplicates[df_without_dupli
         cates.Year==year].shape
```

```
In [22]: dct_df_per_year_publications = sort_dict(dct_df_per_year_publications,'Values','rever
         se')
         dct_df_per_year_publications
```

```
Out[22]: {2020: 6222,
          2019: 4688,
          2018: 4122,
          2016: 3341,
          2014: 3159,
          2015: 3117,
          2017: 3049,
          2021: 2688,
          2013: 2277,
          2004: 2191,
          2012: 2010,
          2008: 1933,
          2009: 1565,
          2011: 1559,
          2010: 1542,
          2006: 1464,
          2007: 1401,
          2003: 1357,
          2005: 1346,
          2000: 793,
          1989: 734,
          1999: 730,
          2001: 726,
          1997: 718,
          1996: 713,
          1994: 703,
          2002: 702,
          1998: 697,
          1995: 601,
          1988: 531,
          1993: 485,
          1990: 457,
          1991: 394,
          1992: 357,
          1987: 351,
          1986: 165,
          1985: 117,
          1984: 70,
          1983: 25,
          1982: 23,
          1977: 17,
          1980: 13,
          1973: 12,
          1978: 12,
          1979: 12,
          1981: 11,
          1974: 8,
          1975: 8,
          1976: 8,
          1971: 7,
          1972: 6,
          1970: 3,
          1962: 1,
          1963: 1,
          1964: 1,
          1965: 1,
          1968: 1,
          1969: 1}
```

f) Total citation per year

```
In [23]: dct_citations_per_year = {}
         for year in year_lst:
             dct_citations_per_year[year] = df_without_duplicates[df_without_duplicates.Year==
         year]['Cited_by'].sum()

         dct_citations_per_year = sort_dict(dct_citations_per_year,'Values','reverse')
```

```
In [24]: dct_citations_per_year
```

```
Out[24]: {2008: 86548.0,
          2016: 85426.0,
          2015: 81343.0,
          2005: 78325.0,
          2004: 75544.0,
          2007: 73896.0,
          2006: 73652.0,
          2014: 69065.0,
          2009: 68433.0,
          2018: 66871.0,
          2013: 63245.0,
          2017: 61390.0,
          2010: 61118.0,
          2012: 56435.0,
          2011: 55280.0,
          2019: 42281.0,
          2000: 36085.0,
          2003: 32783.0,
          1999: 28128.0,
          1998: 26193.0,
          2002: 25418.0,
          2001: 25176.0,
          1997: 21638.0,
          2020: 21386.0,
          1995: 18774.0,
          1994: 18694.0,
          1996: 16943.0,
          1989: 12072.0,
          1992: 10540.0,
          1990: 9800.0,
          1993: 8807.0,
          1991: 8453.0,
          1988: 8385.0,
          1987: 5439.0,
          1986: 3652.0,
          1980: 2616.0,
          1985: 1930.0,
          1977: 1644.0,
          2021: 1568.0,
          1979: 997.0,
          1984: 812.0,
          1971: 446.0,
          1973: 256.0,
          1978: 252.0,
          1976: 215.0,
          1983: 208.0,
          1982: 202.0,
          1975: 186.0,
          1970: 83.0,
          1981: 76.0,
          1972: 49.0,
          1974: 33.0,
          1963: 17.0,
          1962: 16.0,
          1969: 4.0,
          1964: 2.0,
          1965: 2.0,
          1968: 0.0}
```

g) Author(country) having highest co-authorship with indian authors.

Reading India Authors name from previously created file

In [25]:

```
set_of_indian_authors = []
with open('Indian_authors_list.txt','r') as filehandle:
    filecontents = filehandle.readlines()

    for line in filecontents:
        # remove linebreak which is the last character of the string
        author = line[:-1]

        # add item to the list
        set_of_indian_authors.append(author)
filehandle.close()
```

Reading Foreign Authors name from previously created file

In [26]:

```
set_of_foreign_authors = []
with open('Foreign_authors_list.txt','r') as filehandle:
    filecontents = filehandle.readlines()

    for line in filecontents:
        # remove linebreak which is the last character of the string
        author = line[:-1]

        # add item to the list
        set_of_foreign_authors.append(author)
```

Loading Previously created dictonary from file

In [27]:

```
dct_foreign_author_coauth_count = {}
with open('Foreign_auth_and_their_publication_count_with_india_authors_dct.txt','rb')
as file:
    dct_foreign_author_coauth_count = pickle.load(file)
file.close()
```

Finding Foreign author with highest co-authorship with India authors

In [28]:

```
mx_pubs_with_indian_authors = 0
foreign_auth_corr_to_mx_pubs_with_indian_authors = ""
for author in set_of_foreign_authors:
    pubs = dct_foreign_author_coauth_count[author]
    if mx_pubs_with_indian_authors < pubs:
        mx_pubs_with_indian_authors = pubs
        foreign_auth_corr_to_mx_pubs_with_indian_authors = author
```

In [29]:

```
print(f'Foreign author with Highest Co-authorship with Indian Authors =  {foreign_aut
h_corr_to_mx_pubs_with_indian_authors}')
print(mx_pubs_with_indian_authors)
```

```
Foreign author with Highest Co-authorship with Indian Authors =  Nicolaides, A.
127
```

A google search with above authors name tells us that he is from "Vascular Screening and Diagnostic Centre, University of Nicosia, Nicosia, Cyprus"

h) Highest cited author from India and the university

In [31]:

```
max_cites_of_indian_author = 0
highest_cited_indian_author = ""

for author in set_of_indian_authors:
    cites = dct_author_database[author]['Cited_by'].sum()
    if max_cites_of_indian_author < cites:
        max_cites_of_indian_author = cites
        highest_cited_indian_author = author
```

In [32]:

```
print(f'Highest Cited Author from India = {highest_cited_indian_author}')
print(f'His Total Citations = {max_cites_of_indian_author}')
```

```
Highest Cited Author from India = Raghava, G.P.
His Total Citations = 3132.0
```

i) Comparative year wise article publication analysis of India, China and USA

In [33]:

```
df_india = df[df.Country=='India'].copy().reset_index(drop=True)
df_china = df[df.Country=='China'].copy().reset_index(drop=True)
df_usa = df[df.Country=='United States'].copy().reset_index(drop=True)
```

In [34]:

```
dct_india_year_publications = {}
dct_china_year_publications = {}
dct_usa_year_publications = {}
for year in year_lst:
    dct_india_year_publications[year], cols1 = df_india[df_india.Year==year].shape
    dct_china_year_publications[year], cols2 = df_china[df_china.Year==year].shape
    dct_usa_year_publications[year], cols3 = df_usa[df_usa.Year==year].shape
```

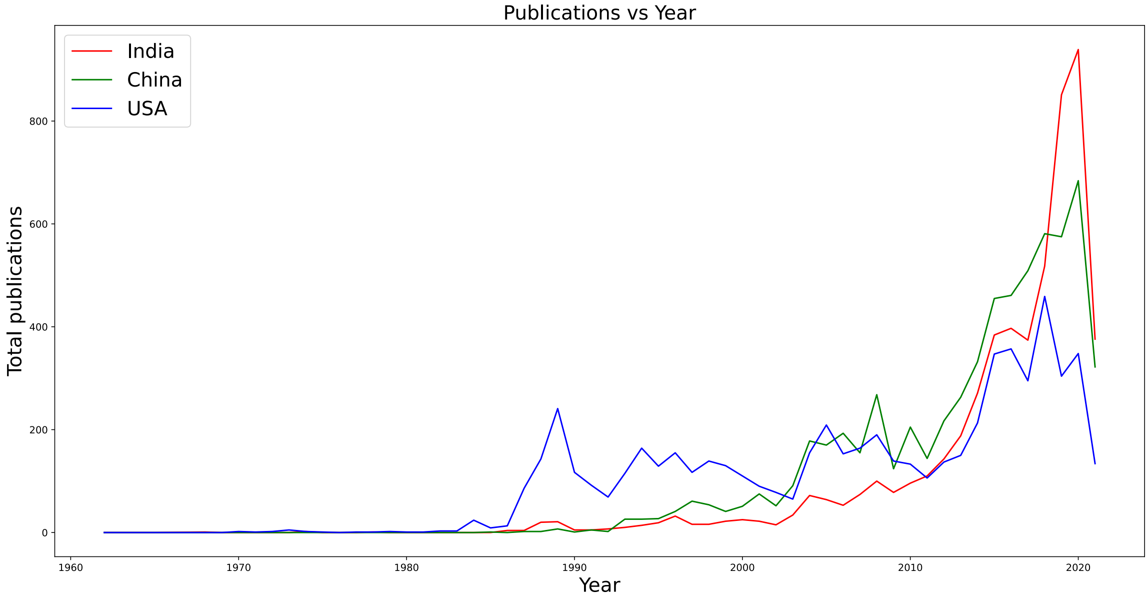
Comparative analysis using graph

In [35]:

```
x_data = [list(dct_india_year_publications.keys()),
          list(dct_china_year_publications.keys()),
          list(dct_usa_year_publications.keys())]

y_data = [list(dct_india_year_publications.values()),
          list(dct_china_year_publications.values()),
          list(dct_usa_year_publications.values())]
```

In [36]:

```
fig = plt.figure(figsize=[20,10])

plt.plot(x_data[0], y_data[0],    label='India',  color='r')
plt.plot(x_data[1], y_data[1],    label='China',  color='g')
plt.plot(x_data[2], y_data[2],    label='USA',    color='b')

plt.xlabel('Year',fontsize=20)
plt.ylabel('Total publications',fontsize=20)
plt.title('Publications vs Year',fontsize=20)
plt.legend(loc='upper left',fontsize=20)
plt.show()
```

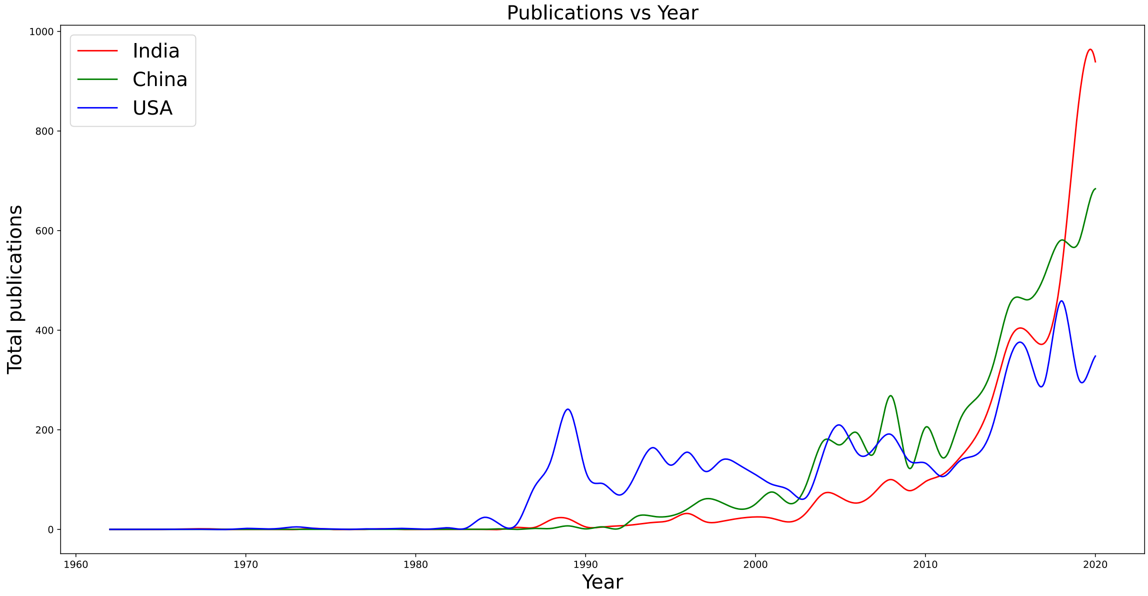


Generating a smoother curve using scipy.interpolate library

In [37]:

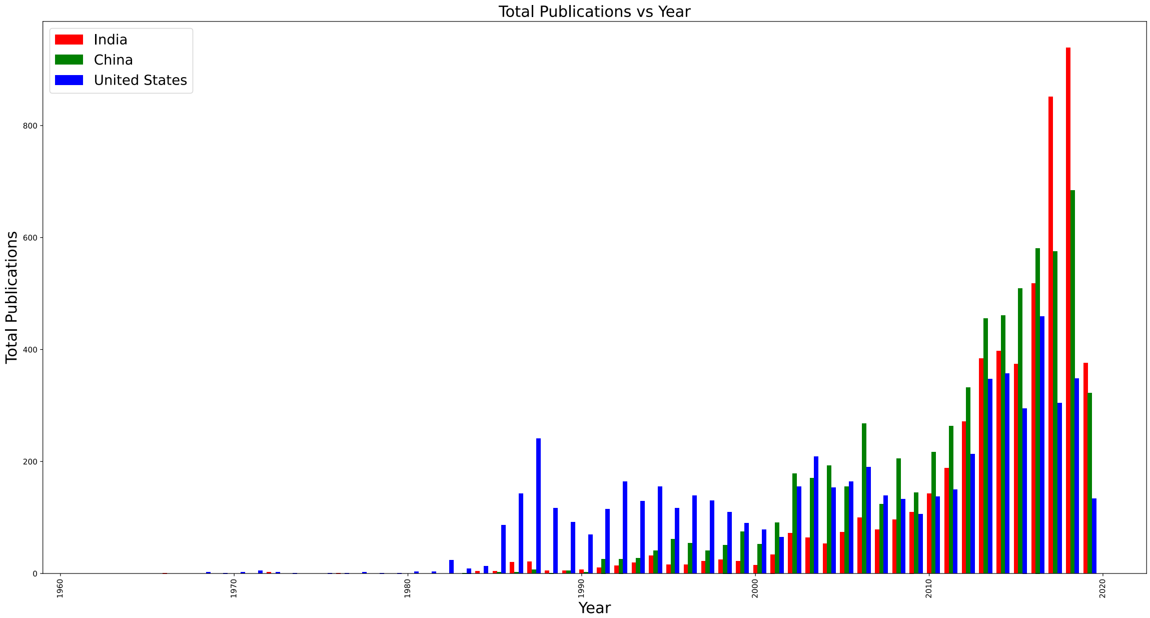
```
y_new = []
x_new = []
for i in range(3):
    x_new_tmp = np.linspace(year_lst[0],year_lst[0]+len(x_data[0]),1000)
    x_new.append(x_new_tmp)

    spline = interpolate.make_interp_spline(x_data[i], y_data[i])
    y_new.append(spline(x_new_tmp))
```

```
In [38]: fig = plt.figure(figsize=[20,10])

         plt.plot(x_new[0], y_new[0],    label='India',  color='r')
         plt.plot(x_new[1], y_new[1],    label='China',  color='g')
         plt.plot(x_new[2], y_new[2],    label='USA',    color='b')

         plt.xlabel('Year',fontsize=20)
         plt.ylabel('Total publications',fontsize=20)
         plt.title('Publications vs Year',fontsize=20)
         plt.legend(loc='upper left',fontsize=20)
         plt.show()
```



In [39]:

```
# plt.rcParams['figure.figsize'] = [20,10]
fig = plt.figure(figsize=[20,10])

X = np.arange(len(year_lst))
X = X + year_lst[0]

fig = fig.add_axes([0,0,1,1])

fig.bar(X + 0.00, list(dct_india_year_publications.values()),  label='India',
color='r',  width =0.25)
fig.bar(X + 0.25, list(dct_china_year_publications.values()),  label='China',
color='g',  width =0.25)
fig.bar(X + 0.50, list(dct_usa_year_publications.values()),    label='United States',
color='b',  width =0.25)

fig.legend(loc='upper left',fontsize=18)

plt.xticks(rotation = 'vertical')

plt.title('Total Publications vs Year',fontsize=20)
plt.xlabel('Year',fontsize=20)
plt.ylabel('Total Publications',fontsize=20)
plt.show()
```



j) Total number of grants given to the field

In [40]:

```
grants, cols = df_without_duplicates[df_without_duplicates['Funding_Details']!='0'].s
hape
```

In [41]:

```
print(f'Grants given to field = {grants}')
```

```
Grants given to field = 4771
```

k) Country wise total number of publication

In [42]:

```
dct_country_publications = {}
for country in country_lst:
    rows, columns = df[df.Country==country].shape
    dct_country_publications[country] = rows
```

In [43]:

```
dct_country_publications = sort_dict(dct_country_publications,'Value','reverse')
```

In [44]:

```
dct_country_publications
```

Out[44]:

```
{'United Kingdom': 8994,
 'China': 6401,
 'United States': 6104,
 'India': 5383,
 'Germany': 5186,
 'Spain': 4759,
 'Canada': 4486,
 'Japan': 4324,
 'Italy': 4214,
 'France': 4133,
 'Australia': 3361,
 'South Korea': 3026,
 'Iran': 2720,
 'Taiwan': 2430,
 'Netherlands': 2173}
```