In [2]:
```
import os
from nltk.tokenize import WordPunctTokenizer
import time
import re
from collections import Counter
from collections import OrderedDict
import matplotlib.pyplot as plt
```

1. Create an inverted index for the corpus. The index should contain only the document postings. What is the time taken to create this index?

In [34]:
```
path_to_bbc = "C:\\Users\\91720\\Downloads\\bbc\\"
stopwords = open("C:\\Users\\91720\\Downloads\\stopwords.txt","r",encoding="ut
stopwords = stopwords.split('\n')
stopwords

vocab = []
postings = {}
token_per_file = []

subfolder_names = os.listdir(path_to_bbc)
ind=0
for subfolder in subfolder_names:
    if os.path.isdir(path_to_bbc + subfolder):
        textfiles = os.listdir(path_to_bbc + subfolder+"\\")
        for data in textfiles:
            try:
                filetext = open(path_to_bbc + subfolder+"\\"+data,"r",encoding
                text=re.sub(r'\s+',' ',filetext)
                pattern = re.compile(r"^\w*[-']?\w*$")
                better_token = list(filter(pattern.match, text.split(" ")))
                better_tokens = [tk for tk in better_token if len(tk) > 3 and
                token_per_file.append(better_tokens)
                for tk in better_tokens:
                    vocab.append(tk)
                    if posting.get(tk) == None:
                        posting[tk] = {ind}
                    else:
                        posting[tk].add(ind)
            except:
                print("error")
                ind = ind - 1;


vocab.sort()
vocab = set(vocab)
len(vocab)
posting_list = OrderedDict(sorted(posting.items()))
posting_list
```

```
("'scruffy", {0, 1177, 2214}),
```

In [ ]:

In [ ]:

2. Write a function to query against the index for a single token. On an average what is the query time for a single word query?

In [35]:
```
def query_search(query):
    return posting_list[query]

start_time = time.time()
query = input("Enter Query (in small caps) : ")
if posting_list.get(query):
    result = query_search(query)
    end_time=time.time()
    print("Time Taken : ", (end_time-start_time) , " s\n")
    print(result)

else:
    print("Try with another query")
```

```
Enter Query (in small caps) : 1900
Time Taken :  3.0053417682647705  s

{0, 2213, 2214, 1416, 685, 2192, 1425, 308, -1, 1822}
```

3. Extend the index created in the first question to contain the offset of the terms in the vocabulary in each document.

In [40]:
```
import os
import re
from collections import OrderedDict

path_to_bbc = "C:\\Users\\91720\\Downloads\\bbc\\"
stopwords = open("C:\\Users\\91720\\Downloads\\stopwords.txt", "r", encoding="
stopwords = stopwords.split('\n')

vocab = set()
postings = {}
token_per_file = []

subfolder_names = os.listdir(path_to_bbc)
ind = 0

for subfolder in subfolder_names:
    if os.path.isdir(os.path.join(path_to_bbc, subfolder)):
        textfiles = os.listdir(os.path.join(path_to_bbc, subfolder))
        for data in textfiles:
            try:
                filetext = open(os.path.join(path_to_bbc, subfolder, data), "r
                text = re.sub(r'\s+', ' ', filetext)
                pattern = re.compile(r"^\w*[-']?\w*$")
                better_token = list(filter(pattern.match, text.split(" ")))
                better_tokens = [tk for tk in better_token if len(tk) > 3 and
                token_per_file.append(better_tokens)
                for offset, tk in enumerate(better_tokens):
                    vocab.add(tk)
                    if tk not in postings:
                        postings[tk] = {ind: [offset]}
                    else:
                        if ind in postings[tk]:
                            postings[tk][ind].append(offset)
                        else:
                            postings[tk][ind] = [offset]
            except:
#                 print("error")
                ind = ind - 1
            ind += 1

vocab = sorted(vocab)
posting_list = OrderedDict(sorted(postings.items()))

# Print limited output
print("Vocabulary size:", len(vocab))
posting_list
```

```
Vocabulary size: 31820
```

Out[40]:
```
OrderedDict([("'2004", {689: [2]}),
             ("'After", {548: [53]}),
             ("'All", {1673: [253]}),
             ("'Before", {548: [51]}),
             ("'Best", {1159: [0]}),
             ("'Big", {1956: [2]}),
             ("'Blackburn", {1575: [75]}),
             ("'Budget", {1262: [1]}),
             ("'Can", {1823: [208, 213]}),
```

4. Use your new index to answer queries like very simple , total failure etc.

In [42]:
```
import re

def search(query):
    query = re.sub(r'\s+', ' ', query)
    query_tokens = [tk for tk in query.split(" ") if len(tk) > 3 and tk not in

    relevant_documents = set()

    for tk in query_tokens:
        if tk in posting_list:
            relevant_documents.update(posting_list[tk])

    return relevant_documents

# Example queries
query1 = "very simple"
query2 = "total failure"

relevant_docs_query1 = search(query1)
relevant_docs_query2 = search(query2)

print("Relevant documents for query '{}':".format(query1), relevant_docs_query
print("\n")
print("Relevant documents for query '{}':".format(query2), relevant_docs_query
```

```
Relevant documents for query 'very simple': {1152, 1793, 1921, 1923, 2177, 21
82, 905, 2185, 1163, 1039, 1168, 2191, 2192, 2194, 1948, 2205, 1822, 1055, 11
87, 682, 1707, 1962, 1963, 2090, 2100, 1847, 1975, 1977, 315, 1212, 1853, 198
1, 959, 1343, 1987, 970, 2129, 1235, 1238, 1239, 1879, 1241, 1246, 2145, 125
0, 2025, 1771, 1772, 621, 1901, 1008, 2161, 2038, 2166, 2045}


Relevant documents for query 'total failure': {2049, 1038, 1554, 2067, 20, 15
57, 23, 535, 2075, 546, 2087, 40, 553, 51, 2100, 565, 2102, 1592, 572, 69, 58
3, 1100, 1101, 78, 2127, 83, 595, 1115, 93, 1118, 95, 608, 2151, 105, 107, 62
0, 619, 622, 112, 121, 1145, 133, 135, 1672, 1673, 2185, 139, 652, 1165, 145,
1685, 150, 1176, 2200, 670, 162, 163, 164, 684, 1199, 691, 693, 183, 1222, 20
2, 725, 1237, 736, 740, 229, 1252, 235, 1260, 1266, 248, 1273, 1275, 252, 25
8, 263, 265, 1801, 1295, 282, 797, 287, 1826, 292, 298, 299, 302, 1839, 306,
309, 821, 823, 313, 1338, 1340, 319, 832, 320, 322, 1346, 838, 842, 332, 186
8, 336, 1877, 346, 1882, 1883, 1885, 1378, 1901, 1911, 1912, 377, 1915, 380,
892, 1918, 893, 386, 899, 389, 1931, 909, 398, 399, 1429, 406, 1942, 410, 41
7, 929, 1953, 418, 1959, 1965, 1966, 431, 946, 947, 1971, 1458, 440, 952, 198
1, 1469, 1472, 962, 452, 1988, 966, 970, 1998, 2000, 466, 978, 2005, 982, 47
8, 998, 488, 1512, 2027, 1007}
```

5. Extend your IR System to answer single wild card queries of the form X*Y or *X or X*

In [45]:
```
import re

def wildcard_match(pattern, term):
    pattern = pattern.replace('*', '.*').replace('?', '.')
    return re.match(pattern, term) is not None

def wildcard_search(query):
    query = re.sub(r'\s+', ' ', query)
    query_tokens = [tk for tk in query.split(" ") if len(tk) > 3 and tk not in

    relevant_documents = set()

    for tk in query_tokens:
        if '*' in tk or '?' in tk:
            for vocab_term in vocab:
                if wildcard_match(tk, vocab_term) and vocab_term in posting_li
                    relevant_documents.update(posting_list[vocab_term])
        else:
            if tk in posting_list:
                relevant_documents.update(posting_list[tk])

    return relevant_documents

# Example wildcard queries
wildcard_query1 = "ver*"
wildcard_query2 = "*tion"
wildcard_query3 = "failure"

relevant_docs_wildcard_query1 = wildcard_search(wildcard_query1)
relevant_docs_wildcard_query2 = wildcard_search(wildcard_query2)
# relevant_docs_wildcard_query3 = wildcard_search(wildcard_query3)

print("Relevant documents for wildcard query '{}':".format(wildcard_query1), r
print("\n")
print("Relevant documents for wildcard query '{}':".format(wildcard_query2), r
# print("Relevant documents for wildcard query '{}':".format(wildcard_query3),
```

```
Relevant documents for wildcard query 'ver*': {2048, 2050, 521, 2057, 2060, 2
063, 1040, 529, 2072, 2073, 540, 542, 2082, 2085, 2089, 555, 46, 2094, 2096,
2097, 2099, 2101, 2103, 568, 2106, 573, 2110, 577, 580, 2118, 2121, 2123, 212
4, 590, 2126, 2128, 85, 2133, 2136, 2138, 1115, 605, 2144, 2145, 611, 2151, 6
18, 2155, 1655, 1656, 633, 2171, 125, 638, 1666, 643, 2179, 645, 1670, 2182,
658, 659, 660, 1172, 2202, 1179, 668, 2204, 667, 1693, 670, 2209, 1189, 677,
678, 2214, 1704, 1706, 1705, 690, 181, 696, 703, 1217, 1234, 724, 735, 226, 2
31, 746, 748, 756, 248, 763, 764, 1789, 1281, 777, 277, 1814, 792, 795, 1825,
802, 807, 1320, 808, 1831, 1833, 1840, 1338, 1340, 1853, 1861, 1864, 1866, 18
67, 1869, 850, 340, 853, 1877, 346, 1883, 1884, 862, 1889, 1896, 1385, 1899,
876, 364, 1902, 366, 881, 1907, 884, 885, 1908, 1912, 1918, 1921, 387, 1925,
1926, 1928, 1929, 1930, 1420, 1935, 1936, 1944, 1434, 922, 1436, 1948, 1951,
420, 1957, 1956, 1964, 1456, 1969, 1983, 1992, 1482, 458, 1999, 2003, 2007, 2
009, 2010, 987, 997, 487, 2026, 2030, 2032, 1011, 2035, 2037, 1529, 2042, 50
9, 2047}


Relevant documents for wildcard query '*tion': {0, 1, 2, 3, 4, 5, 6, 7, 9, 1
0, 11, 12, 13, 14, 15, 16, 17, 18, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 3
0, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 49, 5
0, 51, 52, 54, 55, 56, 57, 58, 59, 60, 62, 63, 64, 65, 66, 67, 68, 69, 70, 7
1, 72, 73, 74, 75, 77, 78, 79, 80, 81, 82, 83, 85, 86, 87, 88, 89, 90, 91, 9
2, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 1
09, 110, 111, 112, 113, 114, 115, 116, 117, 119, 120, 121, 122, 123, 124, 12
5, 126, 128, 129, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142,
143, 144, 145, 146, 147, 149, 150, 151, 152, 153, 154, 155, 157, 158, 159, 16
0, 162, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 175, 176, 177, 178,
179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 19
4, 195, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210,
211, 212, 213, 214, 215, 216, 217, 218, 220, 221, 222, 225, 226, 228, 230, 23
1, 232, 233, 234, 235, 236, 238, 239, 240, 241, 242, 243, 244, 245, 247, 248,
249, 250, 251, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 26
5, 266, 267, 268, 269, 272, 274, 275, 276, 277, 279, 280, 281, 282, 283, 284,
285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 30
0, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315,
316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 33
1, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 343, 344, 346, 347, 348,
349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 362, 363, 364, 36
5, 366, 367, 368, 369, 370, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381,
382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 39
7, 398, 399, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413,
414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 43
0, 431, 432, 433, 434, 435, 436, 438, 439, 441, 442, 443, 444, 445, 447, 448,
449, 450, 453, 454, 455, 456, 457, 458, 459, 461, 462, 463, 464, 465, 466, 46
7, 468, 469, 470, 471, 472, 473, 474, 476, 477, 478, 479, 480, 481, 483, 484,
485, 486, 487, 488, 489, 491, 492, 493, 494, 495, 496, 497, 499, 501, 502, 50
3, 504, 505, 506, 507, 508, 509, 510, 511, 513, 514, 515, 516, 517, 518, 520,
521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 532, 533, 534, 535, 537, 53
8, 539, 540, 541, 542, 544, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555,
556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 57
1, 572, 573, 574, 575, 577, 578, 579, 580, 581, 582, 583, 585, 586, 588, 589,
590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 602, 603, 605, 60
7, 608, 610, 611, 614, 615, 616, 619, 620, 621, 622, 623, 625, 626, 627, 630,
631, 632, 633, 634, 635, 637, 638, 639, 641, 642, 643, 645, 646, 647, 648, 64
9, 650, 652, 653, 655, 656, 658, 659, 661, 662, 663, 664, 666, 667, 668, 671,
672, 673, 674, 675, 676, 679, 681, 682, 683, 684, 686, 687, 690, 691, 692, 69
3, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 709, 711,
```

712, 713, 714, 715, 716, 718, 719, 720, 721, 722, 723, 726, 727, 728, 730, 73
1, 733, 734, 735, 736, 738, 739, 741, 743, 744, 745, 747, 748, 749, 750, 751,
753, 754, 755, 756, 757, 758, 759, 760, 762, 763, 764, 765, 766, 768, 769, 77
0, 771, 772, 773, 775, 776, 777, 778, 779, 780, 781, 782, 784, 785, 787, 788,
789, 790, 791, 793, 794, 795, 796, 797, 798, 799, 800, 801, 802, 803, 805, 80
6, 807, 808, 809, 812, 813, 814, 815, 816, 819, 820, 821, 823, 824, 825, 826,
827, 828, 829, 830, 831, 832, 834, 835, 836, 837, 838, 840, 841, 844, 845, 84
7, 848, 849, 850, 851, 852, 853, 854, 855, 857, 858, 859, 860, 861, 862, 863,
864, 865, 866, 867, 868, 869, 870, 871, 872, 873, 874, 875, 876, 877, 878, 88
0, 881, 882, 883, 884, 885, 886, 888, 889, 890, 891, 892, 893, 894, 895, 896,
897, 898, 899, 900, 901, 902, 903, 904, 905, 906, 907, 908, 909, 910, 911, 91
2, 913, 914, 915, 916, 917, 918, 919, 920, 921, 922, 923, 924, 925, 926, 927,
928, 929, 930, 931, 932, 933, 934, 935, 936, 937, 938, 939, 941, 942, 943, 94
4, 945, 946, 947, 948, 949, 950, 951, 952, 953, 954, 955, 956, 957, 958, 959,
960, 961, 962, 963, 964, 965, 966, 968, 969, 970, 971, 972, 973, 974, 975, 97
6, 977, 978, 979, 980, 981, 982, 983, 984, 985, 986, 987, 988, 989, 990, 991,
992, 993, 994, 995, 996, 997, 998, 999, 1000, 1001, 1002, 1003, 1004, 1005, 1
006, 1007, 1008, 1009, 1010, 1011, 1012, 1013, 1014, 1015, 1016, 1017, 1018,
1019, 1020, 1021, 1022, 1023, 1024, 1025, 1026, 1027, 1028, 1029, 1030, 1031,
1032, 1033, 1034, 1035, 1036, 1037, 1038, 1039, 1040, 1041, 1042, 1043, 1044,
1045, 1046, 1047, 1048, 1049, 1050, 1051, 1052, 1053, 1054, 1055, 1056, 1057,
1058, 1059, 1060, 1061, 1062, 1063, 1064, 1065, 1066, 1067, 1068, 1069, 1070,
1071, 1072, 1073, 1074, 1075, 1076, 1077, 1078, 1079, 1080, 1081, 1082, 1083,
1084, 1085, 1086, 1087, 1088, 1089, 1090, 1091, 1092, 1093, 1094, 1095, 1096,
1097, 1098, 1099, 1100, 1101, 1102, 1103, 1104, 1105, 1106, 1107, 1108, 1109,
1110, 1111, 1112, 1113, 1115, 1116, 1117, 1118, 1119, 1120, 1121, 1123, 1124,
1125, 1126, 1127, 1128, 1129, 1130, 1131, 1132, 1133, 1134, 1135, 1136, 1137,
1138, 1139, 1140, 1141, 1142, 1143, 1144, 1145, 1146, 1147, 1148, 1149, 1150,
1151, 1152, 1153, 1154, 1155, 1156, 1157, 1158, 1159, 1160, 1161, 1162, 1163,
1164, 1165, 1166, 1167, 1168, 1169, 1170, 1171, 1172, 1173, 1174, 1175, 1176,
1177, 1178, 1179, 1180, 1181, 1182, 1183, 1184, 1186, 1187, 1188, 1189, 1191,
1192, 1193, 1195, 1196, 1197, 1198, 1199, 1200, 1201, 1202, 1203, 1204, 1205,
1206, 1207, 1208, 1209, 1210, 1211, 1212, 1213, 1214, 1215, 1216, 1217, 1218,
1219, 1220, 1221, 1222, 1223, 1224, 1225, 1226, 1227, 1228, 1230, 1231, 1232,
1233, 1234, 1235, 1236, 1237, 1238, 1239, 1240, 1241, 1242, 1243, 1244, 1245,
1246, 1247, 1248, 1249, 1250, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 1259,
1260, 1261, 1263, 1264, 1265, 1266, 1268, 1269, 1270, 1271, 1272, 1273, 1274,
1275, 1276, 1277, 1278, 1279, 1280, 1281, 1282, 1283, 1284, 1285, 1286, 1287,
1288, 1289, 1290, 1291, 1292, 1293, 1294, 1295, 1296, 1297, 1298, 1299, 1300,
1301, 1302, 1303, 1304, 1305, 1306, 1307, 1310, 1311, 1312, 1313, 1314, 1315,
1316, 1317, 1318, 1319, 1320, 1321, 1322, 1323, 1324, 1325, 1326, 1327, 1328,
1329, 1330, 1332, 1333, 1334, 1335, 1336, 1337, 1338, 1339, 1340, 1341, 1342,
1343, 1344, 1345, 1346, 1348, 1349, 1350, 1351, 1352, 1353, 1354, 1355, 1356,
1357, 1359, 1360, 1361, 1362, 1364, 1366, 1367, 1369, 1370, 1371, 1372, 1373,
1374, 1375, 1376, 1377, 1378, 1379, 1380, 1382, 1383, 1384, 1385, 1386, 1387,
1389, 1391, 1392, 1393, 1394, 1395, 1396, 1397, 1398, 1399, 1400, 1401, 1402,
1403, 1404, 1405, 1407, 1408, 1409, 1410, 1411, 1413, 1414, 1415, 1417, 1418,
1419, 1420, 1421, 1422, 1424, 1427, 1428, 1429, 1430, 1431, 1432, 1434, 1435,
1436, 1438, 1439, 1440, 1441, 1442, 1444, 1445, 1449, 1450, 1452, 1455, 1456,
1457, 1458, 1459, 1461, 1462, 1463, 1464, 1465, 1466, 1467, 1468, 1469, 1470,
1471, 1472, 1473, 1474, 1475, 1477, 1479, 1480, 1481, 1482, 1483, 1484, 1485,
1487, 1488, 1489, 1490, 1491, 1492, 1494, 1495, 1496, 1497, 1498, 1499, 1500,
1501, 1502, 1503, 1505, 1506, 1508, 1509, 1510, 1511, 1512, 1513, 1514, 1515,
1517, 1518, 1519, 1520, 1521, 1524, 1528, 1529, 1530, 1531, 1532, 1533, 1534,
1535, 1536, 1537, 1538, 1539, 1543, 1544, 1545, 1547, 1548, 1550, 1551, 1552,
1553, 1555, 1556, 1558, 1559, 1560, 1562, 1563, 1564, 1565, 1566, 1567, 1568,
1570, 1572, 1573, 1574, 1575, 1576, 1577, 1578, 1579, 1580, 1581, 1582, 1585,

```
1586, 1587, 1588, 1589, 1590, 1591, 1592, 1593, 1594, 1595, 1596, 1597, 1598,
1599, 1600, 1601, 1602, 1605, 1606, 1607, 1608, 1609, 1610, 1611, 1612, 1613,
1614, 1615, 1617, 1618, 1620, 1621, 1622, 1623, 1624, 1625, 1627, 1628, 1629,
1632, 1633, 1634, 1635, 1636, 1637, 1638, 1639, 1640, 1641, 1642, 1644, 1645,
1646, 1647, 1648, 1649, 1650, 1651, 1653, 1654, 1655, 1656, 1657, 1658, 1659,
1660, 1661, 1662, 1663, 1664, 1665, 1666, 1667, 1668, 1669, 1670, 1671, 1672,
1673, 1674, 1675, 1676, 1677, 1678, 1679, 1680, 1681, 1682, 1684, 1685, 1686,
1687, 1688, 1689, 1690, 1691, 1692, 1693, 1694, 1695, 1696, 1697, 1698, 1699,
1700, 1701, 1702, 1703, 1704, 1705, 1706, 1707, 1708, 1709, 1710, 1711, 1712,
1713, 1714, 1715, 1716, 1717, 1718, 1719, 1720, 1721, 1722, 1734, 1737, 1740,
1745, 1747, 1750, 1751, 1752, 1753, 1757, 1758, 1759, 1762, 1763, 1764, 1766,
1767, 1768, 1769, 1770, 1771, 1772, 1773, 1774, 1775, 1779, 1780, 1781, 1783,
1784, 1785, 1786, 1787, 1788, 1789, 1793, 1794, 1795, 1796, 1797, 1799, 1800,
1801, 1802, 1804, 1809, 1812, 1814, 1815, 1816, 1817, 1818, 1819, 1821, 1822,
1823, 1824, 1825, 1826, 1827, 1829, 1830, 1831, 1832, 1833, 1834, 1835, 1836,
1837, 1838, 1839, 1840, 1841, 1843, 1844, 1845, 1846, 1847, 1848, 1849, 1850,
1851, 1852, 1853, 1854, 1855, 1856, 1857, 1858, 1859, 1861, 1862, 1863, 1864,
1865, 1866, 1867, 1868, 1869, 1870, 1871, 1872, 1874, 1875, 1876, 1877, 1878,
1879, 1880, 1881, 1882, 1883, 1884, 1885, 1886, 1887, 1888, 1889, 1890, 1891,
1892, 1893, 1894, 1895, 1896, 1897, 1898, 1899, 1900, 1901, 1902, 1903, 1904,
1905, 1906, 1907, 1908, 1909, 1910, 1912, 1913, 1914, 1915, 1916, 1917, 1918,
1919, 1920, 1921, 1922, 1923, 1924, 1925, 1926, 1928, 1931, 1932, 1933, 1934,
1935, 1936, 1937, 1938, 1939, 1940, 1941, 1942, 1943, 1944, 1945, 1946, 1947,
1948, 1949, 1950, 1951, 1952, 1953, 1954, 1955, 1956, 1957, 1958, 1959, 1960,
1961, 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970, 1971, 1972, 1973, 1974,
1976, 1977, 1978, 1979, 1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988,
1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001,
2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014,
2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027,
2028, 2029, 2030, 2031, 2032, 2033, 2034, 2035, 2036, 2037, 2038, 2039, 2040,
2041, 2042, 2043, 2044, 2045, 2046, 2047, 2048, 2049, 2050, 2051, 2052, 2053,
2054, 2055, 2056, 2057, 2058, 2059, 2061, 2062, 2063, 2064, 2065, 2066, 2067,
2068, 2069, 2070, 2071, 2072, 2073, 2074, 2075, 2076, 2077, 2078, 2079, 2080,
2081, 2082, 2083, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2092, 2093, 2094,
2095, 2096, 2098, 2099, 2100, 2101, 2104, 2105, 2106, 2107, 2108, 2109, 2110,
2111, 2112, 2113, 2114, 2115, 2116, 2117, 2118, 2119, 2120, 2121, 2122, 2123,
2124, 2125, 2126, 2127, 2128, 2129, 2130, 2131, 2132, 2133, 2134, 2135, 2136,
2137, 2138, 2139, 2141, 2142, 2143, 2144, 2145, 2146, 2147, 2148, 2149, 2150,
2151, 2152, 2153, 2154, 2155, 2156, 2157, 2159, 2160, 2161, 2162, 2163, 2164,
2165, 2166, 2167, 2168, 2169, 2170, 2171, 2172, 2173, 2174, 2175, 2176, 2177,
2178, 2179, 2180, 2181, 2182, 2183, 2184, 2185, 2186, 2187, 2188, 2189, 2190,
2191, 2192, 2193, 2194, 2195, 2196, 2197, 2198, 2199, 2200, 2201, 2202, 2203,
2204, 2205, 2206, 2207, 2208, 2209, 2210, 2211, 2212, 2213, 2214}
```

In [ ]: