Michael Turner

Task Sheet

Firstly, I viewed the original data and there is a grouping that can be made with the first three characters in a 6 character postcode. So I striped away the last 3 characters from all postcodes and then grouped all the similar postcodes together. This is a similar approach that the US uses in the example for §2 of the task sheet. Now for this new grouped dataset, denoted `df`, shown below where `(1108,5)` is the dimension of the dataframe

```
Grouped Postcode Data

  Postcode  Total  Males  Females  Occupied_Households
0      AL1  71682  36101    35581                27506
1      AL2  23335  11452    11883                 9380
2      AL3  28025  13622    14403                11197
3      AL4  27221  13373    13848                10792
4      AL5  30486  14888    15598                11672
(1108, 5)
```

There are in fact some postcode groups that have fewer than 20,000 residents. I found this by trying to extract all the postcodes that have less than 20,000 residents and creating a second dataset, denoted `dfSmall`, which produced the second result

```
Grouped Data where Total < 20000

  Postcode  Total  Males  Females  Occupied_Households
0      AL6  11181   5478     5703                 4502
1      AL8  13366   6298     7068                 5648
2      AL9  10641   5089     5552                 4342
3      B1    8514   4834     3680                 4526
4      B15  17872   8976     8896                 6220
(302, 5)

The percentage of postcodes less than 20000 is: 27.3%
```

The percentage is calculated by

$$\text{Percentage} = \frac{\text{No. of Data} < 20000}{\text{All Data}} = \frac{302}{1108} = 27.25\%$$

So my approach was to order the dataframe `dfSmall` so the column `Total` goes from lowest to highest and create a new column, denoted `Total Addition`, that takes the sum of the column `Total` shown below

```
Ordered Grouped Data
   index Postcode  Total  Males  Females  Occupied_Households  Total Addition
0    181      N1C      7      3        4                   3               7
1     12      B40      9      6        3                   2              16
2    244      TD5     10      6        4                   5              26
3    270      TW6     16     11        5                   7              42
4    165      M17     17      8        9                   7              59
```

Once the column `Total Addition` reaches a value greater than 20,000 I would record the highest index and then group all of the rows together to then be placed under a new code `000`. After that, I would then remove these rows from the dataframe `dfSmall`, reset the index and reset the column `Total Addition` and repeat this process until the entire dataframe has been covered.

So, to visualise this process better, here is the first iteration of the loop where the first 26 rows are found to have a `Total Addition` less than 20,000

	index	Postcode	Total	Males	Females	Occupied_Households	Total Addition
0	181	N1C	7	3	4	3	7
1	12	B40	9	6	3	2	16
2	244	TD5	10	6	4	5	26
3	270	TW6	16	11	5	7	42
4	165	M17	17	8	9	7	59
5	273	W1C	48	30	18	23	107
6	73	DG1	65	30	35	26	172
7	166	M2	88	57	31	54	260
8	245	TD9	105	45	60	49	365
9	280	W1S	299	200	99	208	664
10	60	CR9	339	185	154	25	1003
11	90	EC3	421	264	157	250	1424
12	134	L29	435	206	229	168	1859
13	272	W1B	549	282	267	300	2408
14	6	B2	655	374	281	473	3063
15	150	LD4	817	409	408	373	3880
16	128	L2	935	580	355	622	4815
17	151	LD5	1105	565	540	486	5920
18	91	EC4	1190	723	467	802	7110
19	174	M60	1212	976	236	13	8322
20	274	W1D	1242	801	441	681	9564
21	278	W1J	1396	791	605	820	10960
22	190	PE5	1453	701	752	618	12413
23	275	W1F	1584	956	628	954	13997
24	259	TQ8	2059	986	1073	1030	16056
25	52	CA9	2138	1042	1096	1016	18194

So I group all of these rows together to create one single row and store that in a new dataframe and then remove them all from the original dataframe `dfSmall`. Next I reset the index and the column `Total Addition` to start from 0 again. Here is a figure of the second iteration

	index	Postcode	Total	Males	Females	Occupied_Households	Total Addition
0	8	B3	2226	1283	943	1406	2226
1	252	TN7	2275	1129	1146	966	4501
2	58	CO8	2282	1132	1150	976	6783
3	139	L38	2575	1253	1322	1101	9358
4	97	GL9	2587	1282	1305	1050	11945
5	236	SY9	2821	1378	1443	1233	14766
6	152	LD6	2857	1383	1474	1307	17623

Here we see that the number of rows has decreased to 7. As the values of `Total` become larger, `Total Addition` sums up to a value less than 20,000 in fewer rows. This works fine until we reach postcode where the value of `Total` is greater than 10,000 since the addition of two integers greater than 10,000 will always be greater than 20,000.

This means that we end up creating groups consisting of only 1 postcode where the value of `Total` ranges between $10000 <$ `Total` < 20000. This issue first arises after 39 iterations with the postcodes `CM4` and `HX5` which I will show in the following figure

	Postcode	Total
30	LS4CW3	18168
31	CA4DA4	18475
32	CW4S14	18730
33	IG4SG9	18965
34	B47BH4	19125
35	B72LN9	19347
36	S1 CR6	19738
37	B35DL4	19944
38	CM4	10032
39	HX5	10037

One method that can be used to avoid this is by allowing the sums to be greater than 20,000 so that postcodes with a value of `Total` greater than 10,000 can now be grouped.

So if we increased the value of `Total Addition` to say 50,000 then we can still group together all the postcode with less than 20,000 residents and eliminate the issue so that the lowest `Total` value is greater than 20,000 as shown below

	Postcode	Total	Males	Females	Occupied_Households	Original Postcodes
46	046	33355	16821	16534	13375	B64B18
47	047	33552	16611	16941	14347	IP7TA8
48	048	33622	16432	17190	13925	DA3OL7
49	049	33965	16172	17793	12637	IG5L33
50	050	34098	16689	17409	14225	WS4TS4

This new dataframe consisting of all the grouped postcodes is denoted `frames`. I then create a list of the codes $(000, 001, 002, \ldots)$, store all the groups of original postcodes and then replace the postcode column with the new code list as shown below

	Postcode	Total	Males	Females	Occupied_Households	\
0	000	48035	25674	22361	23631	
1	001	49022	24735	24287	20683	
2	002	45173	22917	22256	20313	
3	003	49005	24029	24976	20856	
4	004	47827	23476	24351	21765	

	Original Postcodes
0	N1CB40TD5TW6M17W1CDG1M2 TD9W1SCR9EC3L29W1BB2 L...
1	MK9TR6BA7TR5W1WBA8RM4W1TRH3B4 LD8LD7
2	EC2HR5LS3B96S33HU2LA7TF6W1U
3	SR1NP8WR7TF5DT7RH9B50YO5PL8
4	LD2B48S32DT8B95LN7W1HL28

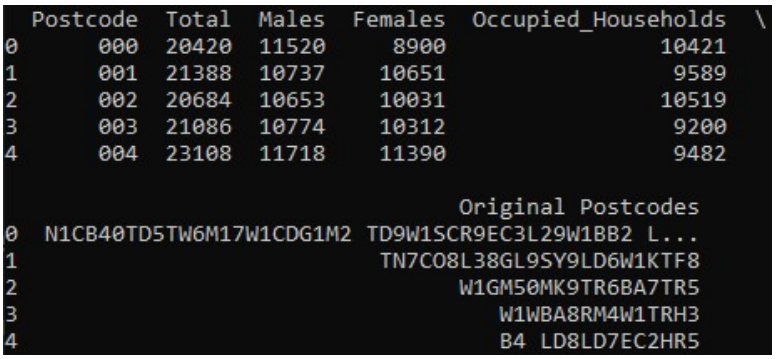
The last step is to then combine this newly coded dataframe with the original dataframe containing all the postcodes with more than 20,000 residents, denoted `combine`.

Final Thoughts

This task really made me think about how best to tackle this problem. I tried lots of different approaches and this seemed to be the best (working) solution to the problem. I did find a way to slightly modify the code so that it works from the other direction, looping until it finds a value of `Total Addition` greater than 20,000 rather than less

than. I will include a figure of how this result looks as well but I shall submit the code for the above results.

This was a very stimulating project and I really enjoyed thinking about the different possibilities to try. It has also improved my ability of working with lists in Python. I now know how manipulate them in much more detail. I hope that this approach is along the right lines and I look forward to possibly discussing even more efficient ways of performing this task.



	Postcode	Total	Males	Females	Occupied_Households	\
0	000	20420	11520	8900	10421	
1	001	21388	10737	10651	9589	
2	002	20684	10653	10031	10519	
3	003	21086	10774	10312	9200	
4	004	23108	11718	11390	9482	

	Original Postcodes
0	N1CB40TD5TW6M17W1CDG1M2 TD9W1SCR9EC3L29W1BB2 L...
1	TN7CO8L38GL9SY9LD6W1KTF8
2	W1GM50MK9TR6BA7TR5
3	W1WBA8RM4W1TRH3
4	B4 LD8LD7EC2HR5