to the `int64` type. This can be accomplished via the `astype` method. You can verify that your columns have been updated using the `info()` method. It should show something like the following:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 103746 entries, 0 to 103745
Data columns (total 11 columns):
year            103746 non-null int64
country         103746 non-null object
origin          103746 non-null object
refugees        103746 non-null int64
asylum          103746 non-null int64
ret_refugees    103746 non-null int64
idps            103746 non-null int64
ret_idps        103746 non-null int64
stateless       103746 non-null int64
others          103746 non-null int64
total           103746 non-null int64
dtypes: int64(9), object(2)
memory usage: 8.7+ MB
```

Hints:

- The `int64` type is defined in the numpy module (`numpy.int64` or `np.int64` depending on how you imported the module).
- You can access multiple columns in a DataFrame via `[[...]]` syntax (passing a list of columns to the lookup). This can be used for both changing the type and storing the resulting columns.
- Remember to update the DataFrame with the new columns

4. Destinations (10 pts)

Write a *function* named `get_destinations` that, given a parameter (a country name), returns all countries that refugees have come from at any time (any year). For example, `s.get_destinations("Turks and Caicos Islands")` would return `['Canada', 'Ukraine', 'United States of America']`. Each country should be reported at most **once** (i.e. there should be no duplicates).

Hints:

- Remember that you select all rows match some criteria using boolean indexing
- It may be useful to think of this as a three-step process: (1) identify all data relating the specified country; (2) obtain only the origins from that data; and (3) make sure the origins are unique.

5. Difference (10 pts)

As discussed in Part 3, the dataset uses an asterisk to indicate a variable number of persons between 1 and 4. We have replaced this unknown value with 4, but this may lead to a mismatch between the sum of the individual fields and the reported total. Add a **new column** named `total_diff` to your DataFrame that encodes the difference between the `total` field and the sum of all the individual refugee categories. The list of all the individual categories is:

```
['asylum', 'idps', 'others','refugees', 'ret_idps','ret_refugees','stateless']
```

Then, the difference is computed as (*total* - *sum*). For example, the value of `df[(df["country"] == "Zimbabwe") & (df["origin"] == 'South Africa') & (df["year"] == 2016)]["total_diff"]` should be -3.

Hints:

- You can extract a subset of columns as in Part 3 using a list to index.
- The `sum` method is very useful here, but remember to set the `axis` parameter appropriately.
- Subtracting one series/column from another works as expected.
- Creating a new column in a DataFrame looks very similar to adding a new key-value pair to a dictionary.

Extra Credit (+5 pts)

Write another *instance function* named `max_year` that, given a country, returns the year that country had the maximum numer of people of concern in their country over all years in the dataset. You must sum multiple rows as there is one row per origin per year. For example, `max_year("Canada")` should return 2010.

Hints:

- `groupby` is very useful here, and you can then sum everything in the same group.
- If a series `s` is labeled by the year, `s.argmax` will be useful.