Question 1

In [58]:
```
p = 0
while p<5:
    try:
        x = int(input("Enter First Number: "))
        y = int(input("Enter Second Number: "))
    except:
        print("Enter numbers!")
        continue
    print("Enter which operation do you want to perform:\n 1. Addition\n 2. Subtraction\
    p = int(input("Enter your choice: "))
    if p == 1:
        print("Result: ",x+y)
    elif p == 2:
        print("Result: ",x-y)
    elif p == 3:
        print("Result: ",x*y)
    elif p == 4:
        try:
            print("Result: ",x/y)
        except:
            print("Can't divide by zero")
        else:
            break
```

```
Enter First Number: 6
Enter Second Number: 0
Enter which operation do you want to perform:
 1. Addition
 2. Subtraction
 3. Multiplication
 4. Division
 5. Exit
Enter your choice: 4
Can't divide by zero
Enter First Number: 5
Enter Second Number: 2
Enter which operation do you want to perform:
 1. Addition
 2. Subtraction
 3. Multiplication
 4. Division
 5. Exit
Enter your choice: 1
Result:  7
Enter First Number: 8
Enter Second Number: 4
Enter which operation do you want to perform:
 1. Addition
 2. Subtraction
 3. Multiplication
 4. Division
 5. Exit
Enter your choice: 4
Result:  2.0
```

Question 2

In [59]:
```
string = input("Enter any string: ")
words = string.split()
words = list(reversed(words))
print(" ".join(words))
```

```
Enter any string: I love to code
code to love I
```

Question 3

In [60]:
```
file = open('para_file.txt','r')
count = {}
para = file.read() .replace("\n","") .replace(" ","") .replace(",","") .replace(".","")
for a in para:
    count[a] = count.get(a,0)+1
print(sorted(count.items()))
```

```
[('A', 2), ('P', 1), ('a', 26), ('b', 1), ('c', 7), ('d', 10), ('e', 28), ('f', 3),
('g', 6), ('h', 5), ('i', 11), ('l', 6), ('m', 2), ('n', 13), ('o', 12), ('p', 10),
('r', 17), ('s', 12), ('t', 15), ('u', 5), ('v', 1), ('y', 1)]
```

Question 4

In [61]:
```
import numpy as np
rand_num = np.random.randint(0,100,15)
rand_num = np.reshape(rand_num,(3,5))
print("Result before adding 2")
print(rand_num)
print("Result after adding 2")
rand_num = rand_num + 2
print(rand_num)
```

```
Result before adding 2
[[60 16 27 97 45]
 [13 72 48 22 67]
 [11 16 20 20  1]]
Result after adding 2
[[62 18 29 99 47]
 [15 74 50 24 69]
 [13 18 22 22  3]]
```

Question 5

In [64]:
```
# a)
```

In [65]:
```
import pandas as pd
df = pd.read_csv("KCLT.csv")
df['newColumn'] = df['actual_mean_temp']
df.head()
```

Out[65]:

	date	actual_mean_temp	actual_min_temp	actual_max_temp	average_min_temp	average_max_temp	rec
0	2014-7-1	81	70	91	67	89	
1	2014-7-2	85	74	95	68	89	
2	2014-7-3	82	71	93	68	89	
3	2014-7-4	75	64	86	68	89	
4	2014-7-5	72	60	84	68	89	

In [66]:
```
# b)
```

In [67]:
```
df.drop('newColumn', inplace = True, axis = 1)
df.head()
```

Out[67]:

	date	actual_mean_temp	actual_min_temp	actual_max_temp	average_min_temp	average_max_temp	rec
0	2014-7-1	81	70	91	67	89	
1	2014-7-2	85	74	95	68	89	
2	2014-7-3	82	71	93	68	89	
3	2014-7-4	75	64	86	68	89	
4	2014-7-5	72	60	84	68	89	

In [68]:
```
# c)
```

In [69]:
```
df.set_index("date",inplace = True)
row = df.loc["2014-7-3"]
df.drop("2014-7-3", axis = 0, inplace = True)
df.head(10)
```

Out[69]:

date	actual_mean_temp	actual_min_temp	actual_max_temp	average_min_temp	average_max_temp	record
2014-7-1	81	70	91	67	89	
2014-7-2	85	74	95	68	89	
2014-7-4	75	64	86	68	89	
2014-7-5	72	60	84	68	89	
2014-7-6	74	61	87	68	89	
2014-7-7	79	67	91	68	89	
2014-7-8	83	72	94	68	89	
2014-7-9	80	71	89	68	89	
2014-7-10	78	71	85	68	89	
2014-7-11	78	68	87	68	89	

In [70]:
```
# d)
```

In [71]:
```
df = df.append(row)
df
```

Out[71]:

date	actual_mean_temp	actual_min_temp	actual_max_temp	average_min_temp	average_max_temp	record
2014-7-1	81.0	70.0	91.0	67.0	89.0	
2014-7-2	85.0	74.0	95.0	68.0	89.0	
2014-7-4	75.0	64.0	86.0	68.0	89.0	
2014-7-5	72.0	60.0	84.0	68.0	89.0	
2014-7-6	74.0	61.0	87.0	68.0	89.0	
...	...	...	...	...	...	...
2015-6-27	82.0	71.0	92.0	67.0	88.0	
2015-6-28	76.0	66.0	85.0	67.0	88.0	
2015-6-29	73.0	59.0	87.0	67.0	88.0	
2015-6-30	83.0	71.0	94.0	67.0	89.0	
2014-7-3	82.0	71.0	93.0	68.0	89.0	

365 rows × 12 columns

In [72]:
```
df.head(10)
```

Out[72]:

date	actual_mean_temp	actual_min_temp	actual_max_temp	average_min_temp	average_max_temp	record
2014-7-1	81.0	70.0	91.0	67.0	89.0	
2014-7-2	85.0	74.0	95.0	68.0	89.0	
2014-7-4	75.0	64.0	86.0	68.0	89.0	
2014-7-5	72.0	60.0	84.0	68.0	89.0	
2014-7-6	74.0	61.0	87.0	68.0	89.0	
2014-7-7	79.0	67.0	91.0	68.0	89.0	
2014-7-8	83.0	72.0	94.0	68.0	89.0	
2014-7-9	80.0	71.0	89.0	68.0	89.0	
2014-7-10	78.0	71.0	85.0	68.0	89.0	
2014-7-11	78.0	68.0	87.0	68.0	89.0	

In [74]:
```
# e)
```

In [75]:
```
index = df.index
condition = df["date"] == "2014-7-3"
indices = index[condition]
indices_list = indices.tolist()
print(indices_list)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, metho
d, tolerance)
   3079             try:
-> 3080                 return self._engine.get_loc(casted_key)
   3081             except KeyError as err:

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.ge
t_item()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.ge
t_item()

KeyError: 'date'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
<ipython-input-75-d9f8515578d1> in <module>
      1 index = df.index
----> 2 condition = df["date"] == "2014-7-3"
      3 indices = index[condition]
      4 indices_list = indices.tolist()
      5 print(indices_list)

~\anaconda3\lib\site-packages\pandas\core\frame.py in __getitem__(self, key)
   3022             if self.columns.nlevels > 1:
   3023                 return self._getitem_multilevel(key)
-> 3024             indexer = self.columns.get_loc(key)
   3025             if is_integer(indexer):
   3026                 indexer = [indexer]

~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, metho
d, tolerance)
   3080                 return self._engine.get_loc(casted_key)
   3081             except KeyError as err:
-> 3082                 raise KeyError(key) from err
   3083
   3084         if tolerance is not None:

KeyError: 'date'
```