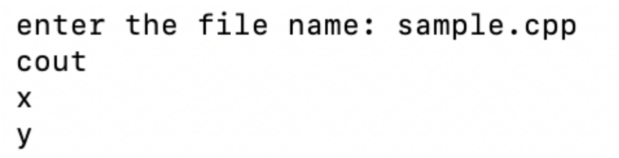
Question 1.

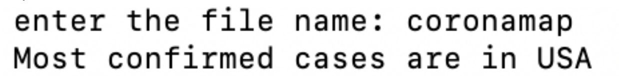


```
enter the file name: sample.cpp
cout
x
y
```

In this program, I write a function called isChar, this function takes in a string and determines if the char inside of this string is an alphabet or not.

Inside of the while loop, first find the element before comment, not in the include part at the beginning of the program, and not in the string constants. After that, find the reserve word in the line, and then find the word after the reserved word, if that word is not inside of reserved word, then it can be added into idents.

Question 2.

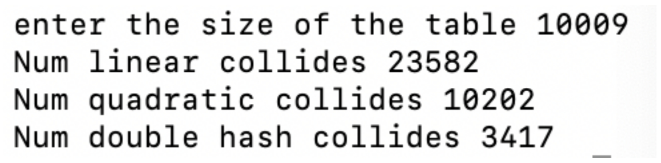


```
enter the file name: coronamap
Most confirmed cases are in USA
```

First read the data from the file, separate the country name and population and make them into pairs and saved in map.

Then, we just need to find the max population in the map and print the country name of max population.

Question 3.



```
enter the size of the table 10009
Num linear collides 23582
Num quadratic collides 10202
Num double hash collides 3417
```

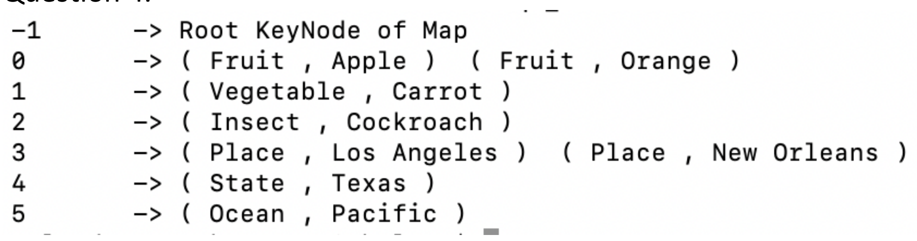
InsertLinear: find the first place to store x by x % linear.size(). If this place is not empty, then position increases by 1 and test if next position is empty. Keep find a position until there is an empty space.

InsertQuad: find the first place to store x by x %quad.size(). Create an integer called i and initialize it to 0. If the place is not empty, increase i by 1 and add i+i*i to position in order to find the next position. If i+i*i is larger than the size of quad, we can return -1 to show that there is no place for this element.

InsertDuble: find the first place to store x by x % duble.size(). If the place is not empty, add dubHash and mod position by the size of duble to find the next position to insert.

Also, always check if the number is same with the number inside of the hash table, if they are same, just break from the loop.

Question 4.



```
-1      -> Root KeyNode of Map
0       -> ( Fruit , Apple )  ( Fruit , Orange )
1       -> ( Vegetable , Carrot )
2       -> ( Insect , Cockroach )
3       -> ( Place , Los Angeles )  ( Place , New Orleans )
4       -> ( State , Texas )
5       -> ( Ocean , Pacific )
```

The hashFunction is used to give hash key to each key. I just give hash keys as the order keys.

If the key already had a hash key, then just return the hash key, else, it will give a new hash key for the key. If we have more keys than the max size, return -1 to show the error.

The insert function is used to insert the value node after key node. There are 4 different situations. The first is there is no nodes after the root node, we have to create a new key node and put it under the root node. The second is hash key is less than the next node, we have to create a new node and put it between current node and the next node. The third is hash key equal to the next node, we need to create a new value node and put all values inside of it, and then put it in the end of the lists (if there is already some value nodes after the key node, just keep going right until the pointer points to NULL). And the last is hash key larger than next nodes, then just goes to the next node.