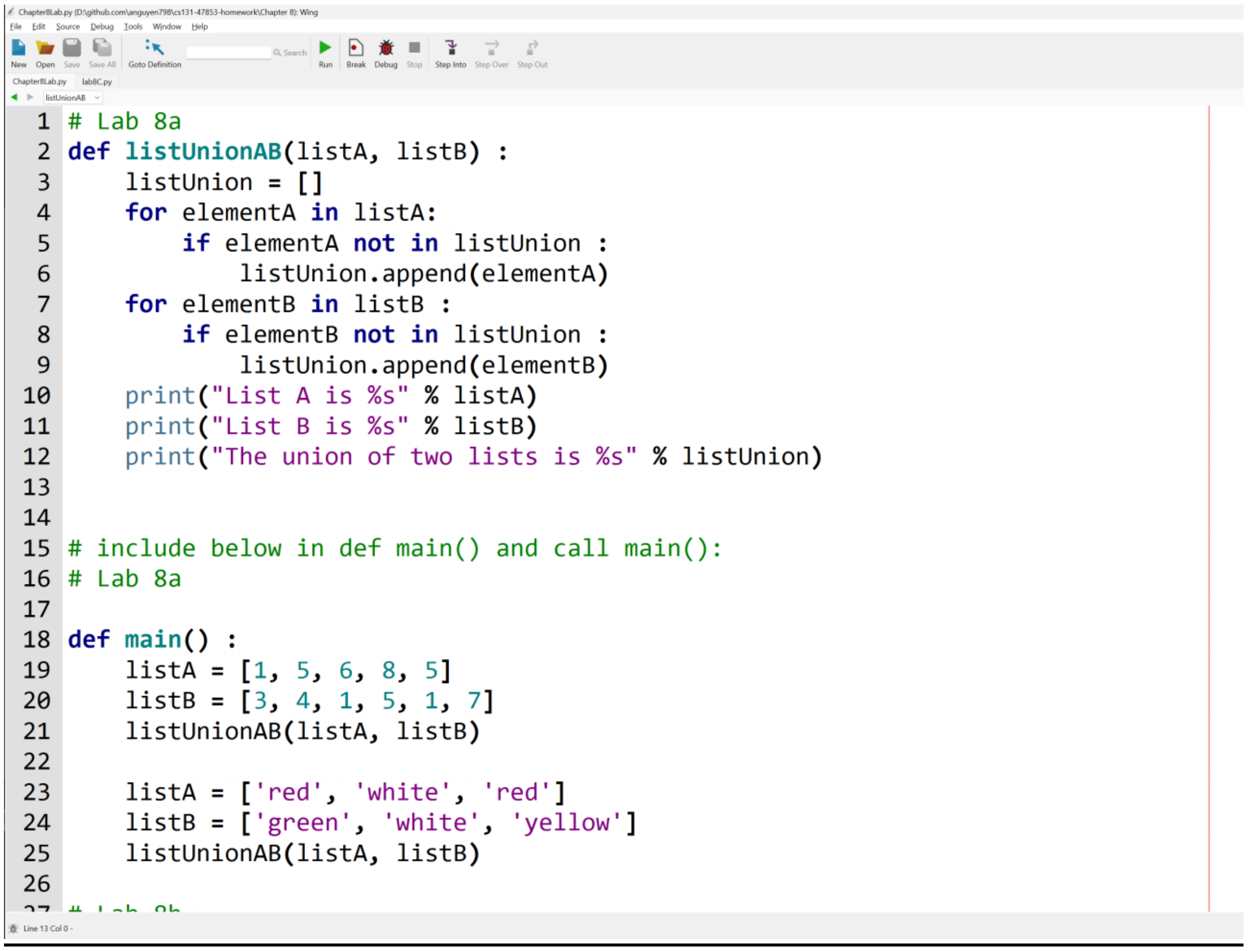
Thomas Nguyen, Albert Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D

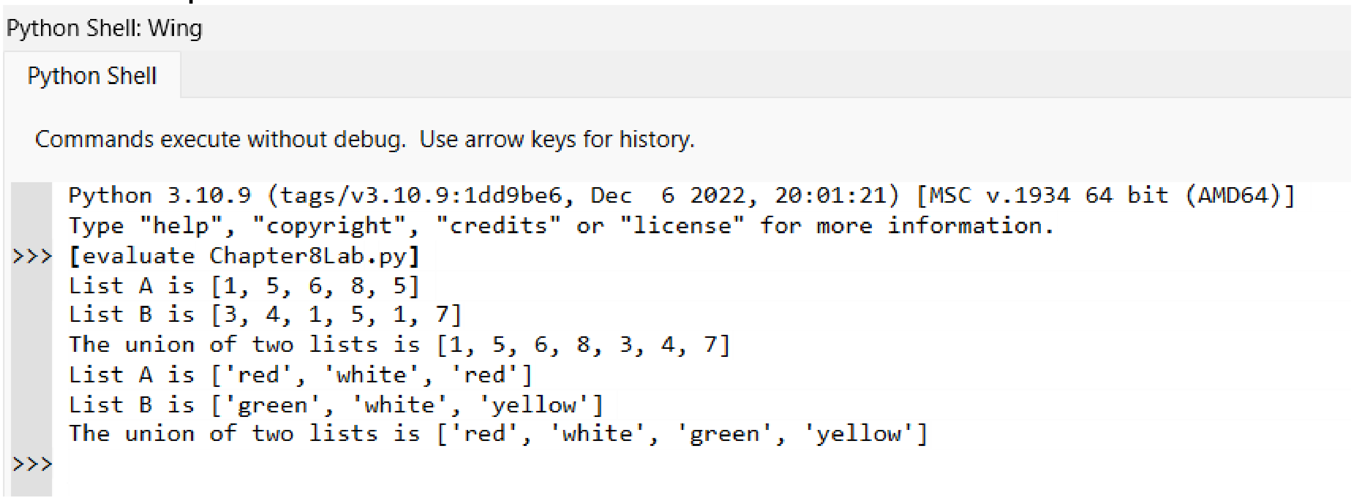
Lab 8A

Lab 8A - Code


Chapter8Lab.py (D:\github.com\anguyen798\cs131-47853-homework\Chapter 8): Wing
File Edit Source Debug Tools Window Help
New Open Save Save All Goto Definition Search Run Break Debug Stop Step Into Step Over Step Out
Chapter8Lab.py lab8C.py
listUnionAB
Line 13 Col 0 -

```python
# Lab 8a
def listUnionAB(listA, listB) :
    listUnion = []
    for elementA in listA:
        if elementA not in listUnion :
            listUnion.append(elementA)
    for elementB in listB :
        if elementB not in listUnion :
            listUnion.append(elementB)
    print("List A is %s" % listA)
    print("List B is %s" % listB)
    print("The union of two lists is %s" % listUnion)


# include below in def main() and call main():
# Lab 8a

def main() :
    listA = [1, 5, 6, 8, 5]
    listB = [3, 4, 1, 5, 1, 7]
    listUnionAB(listA, listB)

    listA = ['red', 'white', 'red']
    listB = ['green', 'white', 'yellow']
    listUnionAB(listA, listB)

```

Lab 8A - Output



```
Commands execute without debug.  Use arrow keys for history.

    Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec  6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
    Type "help", "copyright", "credits" or "license" for more information.
>>> [evaluate Chapter8Lab.py]
    List A is [1, 5, 6, 8, 5]
    List B is [3, 4, 1, 5, 1, 7]
    The union of two lists is [1, 5, 6, 8, 3, 4, 7]
    List A is ['red', 'white', 'red']
    List B is ['green', 'white', 'yellow']
    The union of two lists is ['red', 'white', 'green', 'yellow']
>>>
```

Albert Nguyen, Thomas Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D

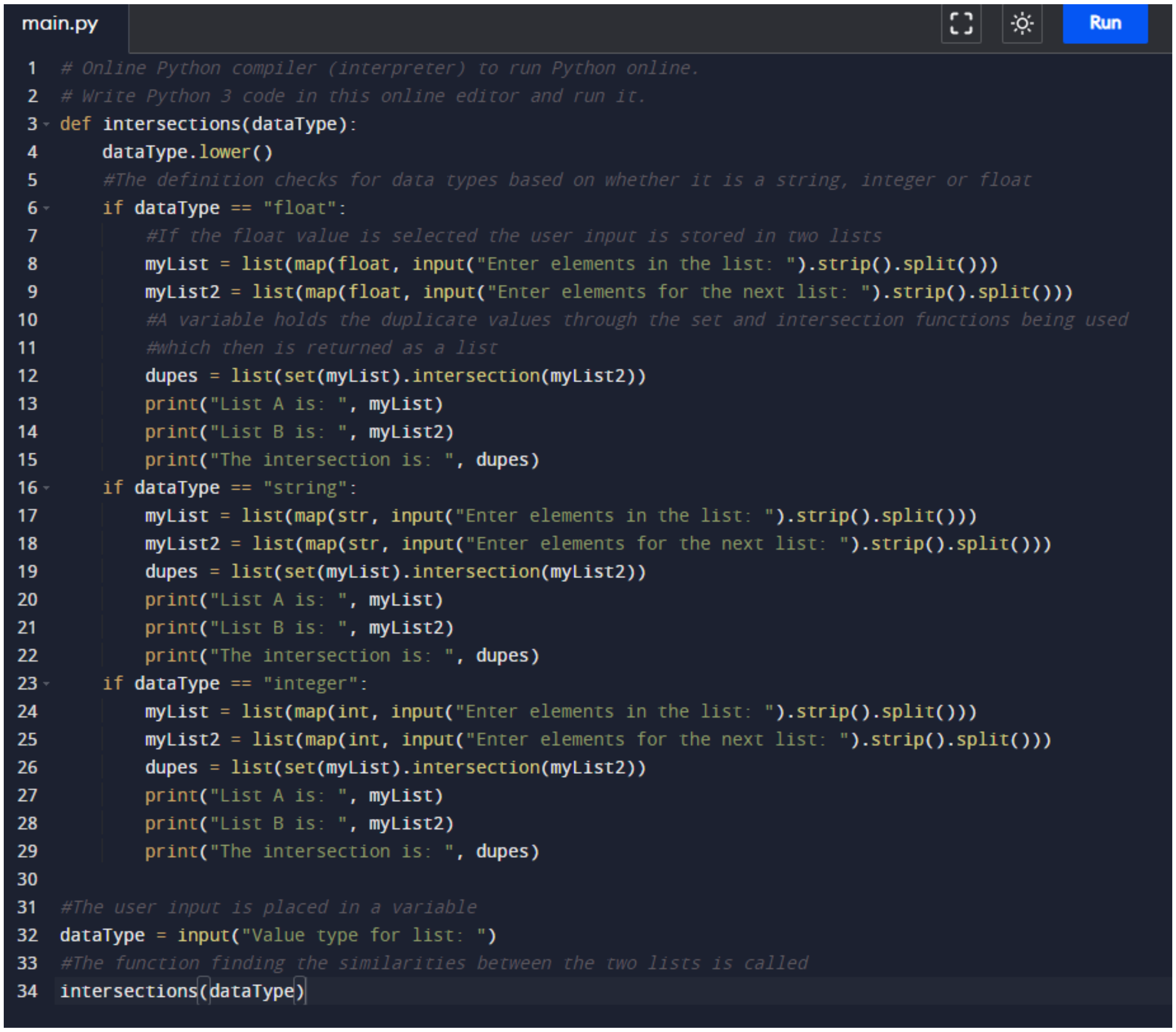
Lab 8A – Written Code

```
def listUnionAB(listA, listB) :
    listUnion = []
    for elementA in listA:
        if elementA not in listUnion :
            listUnion.append(elementA)
    for elementB in listB :
        if elementB not in listUnion :
            listUnion.append(elementB)
    print("List A is %s" % listA)
    print("List B is %s" % listB)
    print("The union of two lists is %s" % listUnion)


def main() :
    listA = [1, 5, 6, 8, 5]
    listB = [3, 4, 1, 5, 1, 7]
    listUnionAB(listA, listB)

    listA = ['red', 'white', 'red']
    listB = ['green', 'white', 'yellow']
    listUnionAB(listA, listB)


main()
```

Thomas Nguyen, Albert Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D

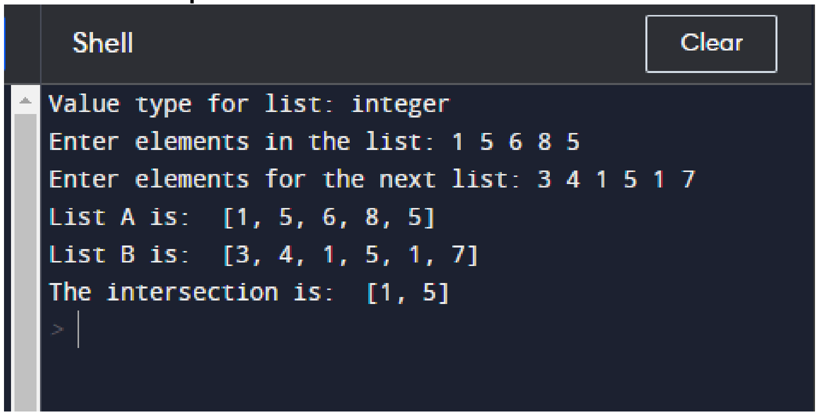
Lab 8B

Lab 8B - Code



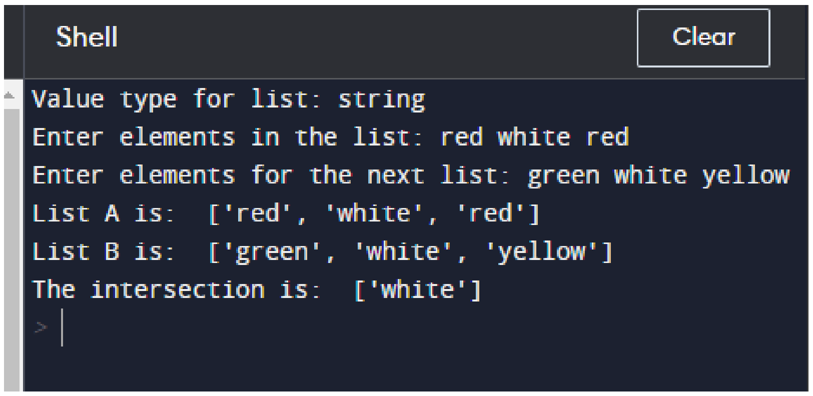
```python
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
def intersections(dataType):
    dataType.lower()
    #The definition checks for data types based on whether it is a string, integer or float
    if dataType == "float":
        #If the float value is selected the user input is stored in two lists
        myList = list(map(float, input("Enter elements in the list: ").strip().split()))
        myList2 = list(map(float, input("Enter elements for the next list: ").strip().split()))
        #A variable holds the duplicate values through the set and intersection functions being used
        #which then is returned as a list
        dupes = list(set(myList).intersection(myList2))
        print("List A is: ", myList)
        print("List B is: ", myList2)
        print("The intersection is: ", dupes)
    if dataType == "string":
        myList = list(map(str, input("Enter elements in the list: ").strip().split()))
        myList2 = list(map(str, input("Enter elements for the next list: ").strip().split()))
        dupes = list(set(myList).intersection(myList2))
        print("List A is: ", myList)
        print("List B is: ", myList2)
        print("The intersection is: ", dupes)
    if dataType == "integer":
        myList = list(map(int, input("Enter elements in the list: ").strip().split()))
        myList2 = list(map(int, input("Enter elements for the next list: ").strip().split()))
        dupes = list(set(myList).intersection(myList2))
        print("List A is: ", myList)
        print("List B is: ", myList2)
        print("The intersection is: ", dupes)

#The user input is placed in a variable
dataType = input("Value type for list: ")
#The function finding the similarities between the two lists is called
intersections(dataType)
```

Lab 8B – Output



```
Value type for list: integer
Enter elements in the list: 1 5 6 8 5
Enter elements for the next list: 3 4 1 5 1 7
List A is:  [1, 5, 6, 8, 5]
List B is:  [3, 4, 1, 5, 1, 7]
The intersection is:  [1, 5]
>
```

Albert Nguyen, Thomas Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D



```
Value type for list: string
Enter elements in the list: red white red
Enter elements for the next list: green white yellow
List A is:  ['red', 'white', 'red']
List B is:  ['green', 'white', 'yellow']
The intersection is:  ['white']
>
```

Thomas Nguyen, Albert Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D

Lab 8B – Written Code

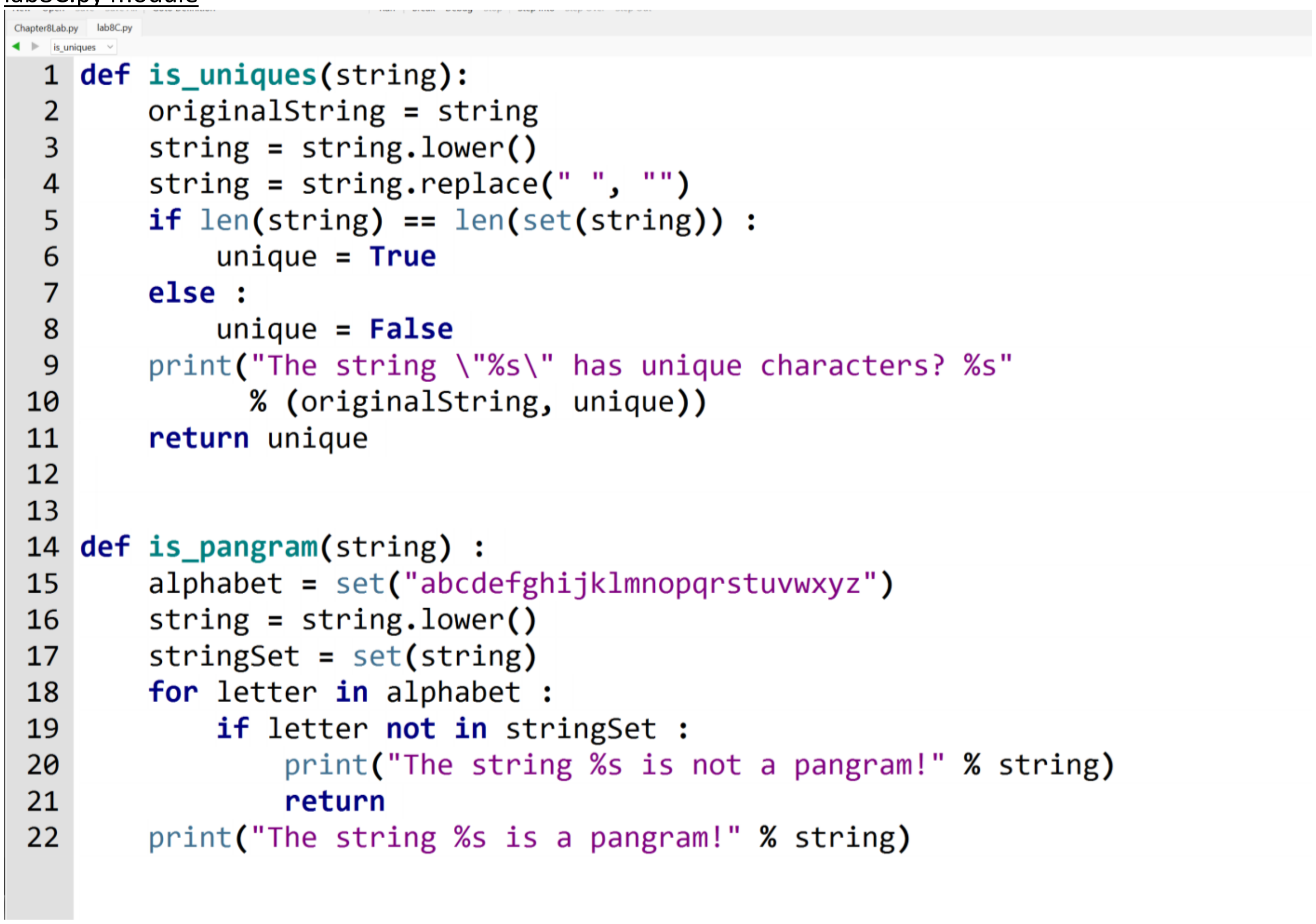
```
def intersections(dataType) :
    dataType.lower()
    #The definition checks for data types based on whether it is a string, integer or float
    if dataType == "float" :
        # If the float value is selected the user input is stored in two lists
        myList = list(map(float, input("Enter elements in the list: ").strip().split()))
        myList2 = list(map(float, input("Enter elements for the next list: ").strip().split()))
        # A variable holds the duplicate values through the set and intersection functions being used
        # Which then is returned as a list
        dupes = list(set(myList).intersection(myList2))
        print("List A is: ", myList)
        print("List B is: ", myList2)
        print("The intersection is ", dupes)
    if dataType == "string" :
        myList = list(map(str, input("Enter elements in the list: ").strip().split()))
        myList2 = list(map(str, input("Enter elements for the next list: ").strip().split()))
        dupes = list(set(myList).intersection(myList2))
        print("List A is: ", myList)
        print("List B is: ", myList2)
        print("The intersection is ", dupes)
    if dataType == "integer" :
        myList = list(map(int, input("Enter elements in the list: ").strip().split()))
        myList2 = list(map(int, input("Enter elements for the next list: ").strip().split()))
        dupes = list(set(myList).intersection(myList2))
        print("List A is: ", myList)
        print("List B is: ", myList2)
        print("The intersection is ", dupes)

# The user input is placed in a variable
dataType = input("Value type for list: ")

# The function finding the similarities between the two lists is called
intersections(dataType)
```

Albert Nguyen, Thomas Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D

Lab 8C

Lab 8C - Code

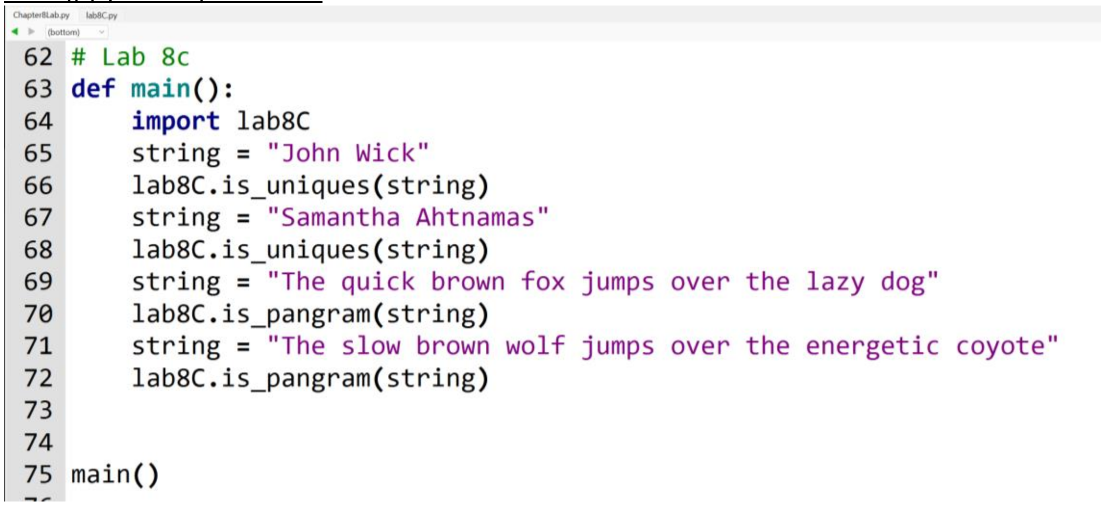
lab8C.py module



Chapter8Lab.py lab8C.py
is_uniques

```python
def is_uniques(string):
    originalString = string
    string = string.lower()
    string = string.replace(" ", "")
    if len(string) == len(set(string)) :
        unique = True
    else :
        unique = False
    print("The string \"%s\" has unique characters? %s"
          % (originalString, unique))
    return unique


def is_pangram(string) :
    alphabet = set("abcdefghijklmnopqrstuvwxyz")
    string = string.lower()
    stringSet = set(string)
    for letter in alphabet :
        if letter not in stringSet :
            print("The string %s is not a pangram!" % string)
            return
    print("The string %s is a pangram!" % string)
```

main().py file import lab8C

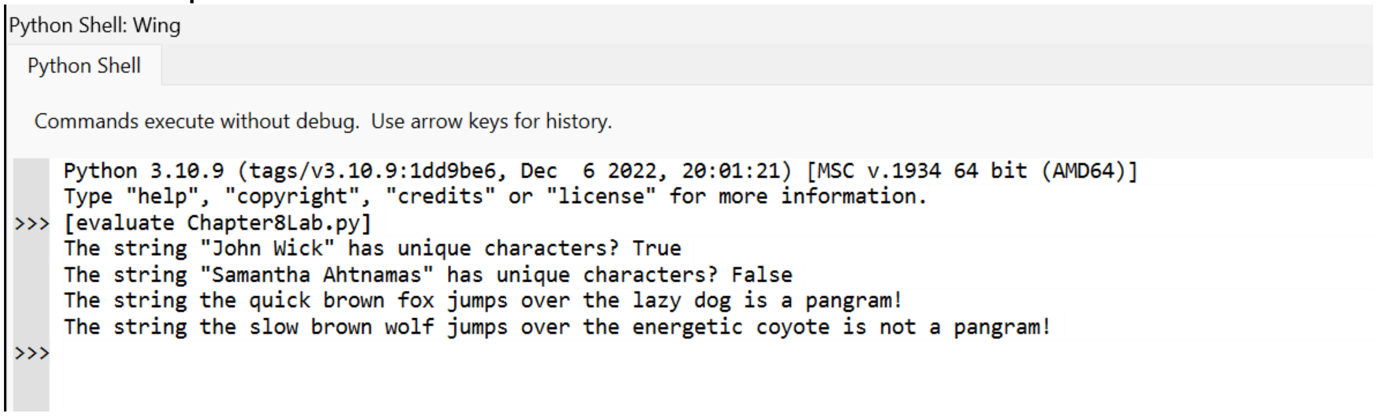


Chapter8Lab.py lab8C.py
(bottom)

```python
# Lab 8c
def main():
    import lab8C
    string = "John Wick"
    lab8C.is_uniques(string)
    string = "Samantha Ahtnamas"
    lab8C.is_uniques(string)
    string = "The quick brown fox jumps over the lazy dog"
    lab8C.is_pangram(string)
    string = "The slow brown wolf jumps over the energetic coyote"
    lab8C.is_pangram(string)


main()
```

Thomas Nguyen, Albert Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D

Lab 8C - Output


Python Shell: Wing

Python Shell

Commands execute without debug. Use arrow keys for history.

```
    Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec  6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
    Type "help", "copyright", "credits" or "license" for more information.
>>> [evaluate Chapter8Lab.py]
    The string "John Wick" has unique characters? True
    The string "Samantha Ahtnamas" has unique characters? False
    The string the quick brown fox jumps over the lazy dog is a pangram!
    The string the slow brown wolf jumps over the energetic coyote is not a pangram!
>>>
```

Lab 8C – Written Code

# from lab8C.py module

```
def is_uniques(string):
    originalString = string
    string = string.lower()
    string = string.replace(" ", "")
    if len(string) == len(set(string)) :
        unique = True
    else :
        unique = False
    print("The string \"%s\" has unique characters? %s" % (originalString,
unique))
    return unique



def is_pangram(string) :
    alphabet = set("abcdefghijklmnopqrstuvwxyz")
    string = string.lower()
    stringSet = set(string)
    for letter in alphabet :
        if letter not in stringSet :
            print("The string %s is not a pangram!" % string)
            return
    print("The string %s is a pangram!" % string)
```

Albert Nguyen, Thomas Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D

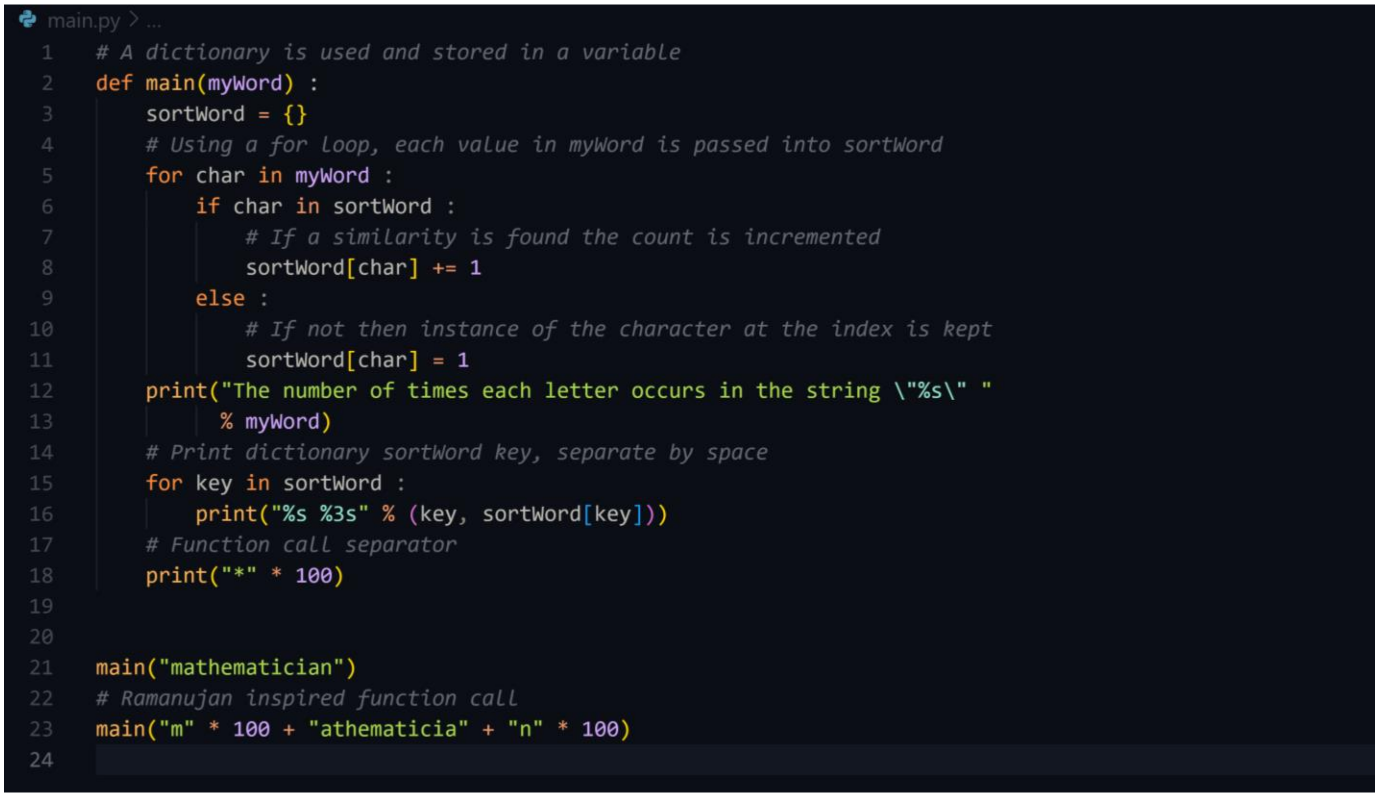
# main().py file import lab8C

```
def main() :
    import lab8C
    string = "John Wick"
    lab8C.is_uniques(string)
    string = "Samantha Ahtnamas"
    lab8C.is_uniques(string)
    string = "The quick brown fox jumps over the lazy dog"
    lab8C.is_pangram(string)
    string = "The slow brown wolf jumps over the energetic coyote"
    lab8C.is_pangram(string)

main()
```

Thomas Nguyen, Albert Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D

Lab 8D

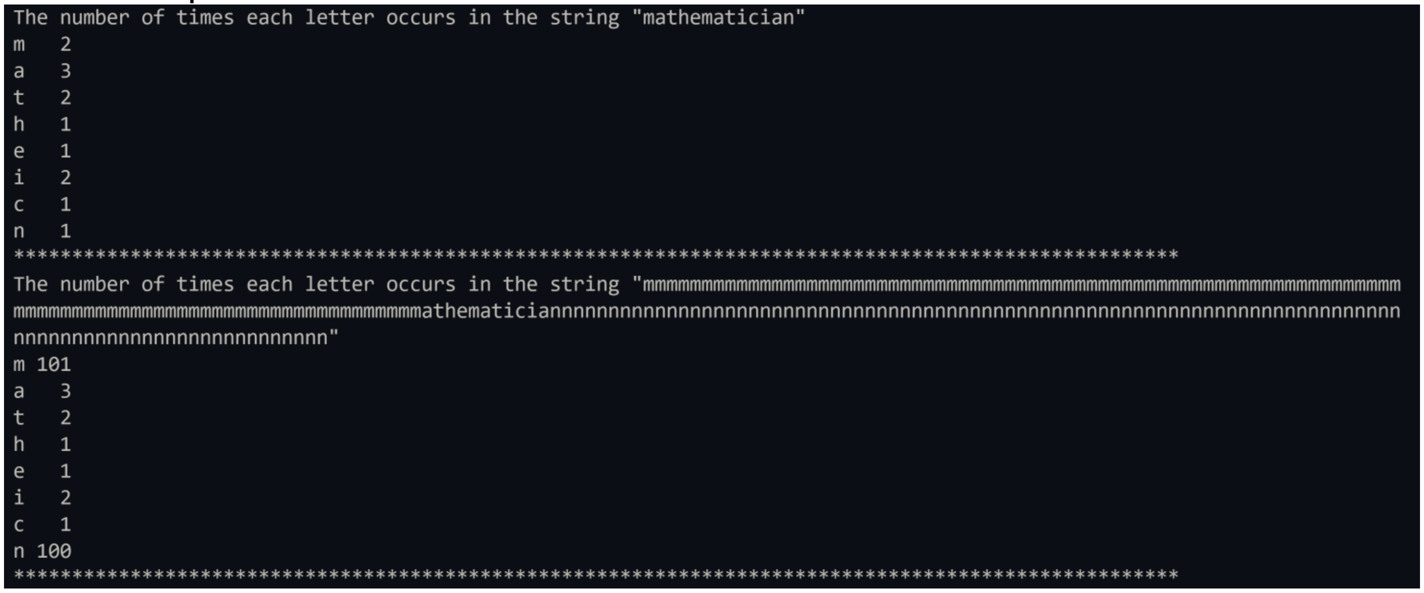
Lab 8D - Code



main.py > ...

```python
# A dictionary is used and stored in a variable
def main(myWord) :
    sortWord = {}
    # Using a for loop, each value in myWord is passed into sortWord
    for char in myWord :
        if char in sortWord :
            # If a similarity is found the count is incremented
            sortWord[char] += 1
        else :
            # If not then instance of the character at the index is kept
            sortWord[char] = 1
    print("The number of times each letter occurs in the string \"%s\" "
          % myWord)
    # Print dictionary sortWord key, separate by space
    for key in sortWord :
        print("%s %3s" % (key, sortWord[key]))
    # Function call separator
    print("*" * 100)


main("mathematician")
# Ramanujan inspired function call
main("m" * 100 + "athematicia" + "n" * 100)
```

Lab 8D - Output



```
The number of times each letter occurs in the string "mathematician"
m   2
a   3
t   2
h   1
e   1
i   2
c   1
n   1
****************************************************************************************************
The number of times each letter occurs in the string "mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm
mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmathematiciannnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn
nnnnnnnnnnnnnnnnnnnnnnnnnn"
m 101
a   3
t   2
h   1
e   1
i   2
c   1
n 100
****************************************************************************************************
```

Albert Nguyen, Thomas Nguyen
CS131-47853 MW 06:00 PM – 08:30 PM
Lab 8A-8D

Lab 8D – Written Code

```
# A dictionary is used and stored in a variable
def main(myWord) :
    sortWord = {}
    # Using a for loop, each value in myWord is passed into sortWord
    for char in myWord :
        if char in sortWord :
            # If a similarity is found the count is incremented
            sortWord[char] += 1
        else :
            # If not then instance of the character at the index is kept
            sortWord[char] = 1
    print("The number of times each letter occurs in the string \"%s\" "
          % myWord)
    # Print dictionary sortWord key, separate by space
    for key in sortWord :
        print("%s %3s" % (key, sortWord[key]))
    # Function call separator
    print("*" * 100)


main("mathematician")
# Ramanujan inspired function call
main("m" * 100 + "athematicia" + "n" * 100)
```