Ricardo McCrary, Hanyue Wang,
Cs 126L 001
Lab 5 phone contacts
10/11/19

1. Problem statement

In this lab we were tasked with creating code that would run as an import to test code. With our code we had to make a system where you could create new contacts with a first name, last name, email, age and phone number. Through this system we had to write code that would be able to interact with dictionaries to help create said contacts on a possible phone app. Some of the important features was able to add people into the contacts list, update their age, email, phone number, and the capability to check if they are in our phone contacts as well as remove them. We were required to create code that would be used as a library of functions that could then be accessed by the test code. beneath is some of the code that was required to make this work:

- Add_contacat(contacts, first, last, email, phone number, age)
- Update contact age(contacts, first, last, age)
- Other update contacts for email and phone number
- A remove contact function
- A make key function
- Assigning the make key function values for the dictionary
- Get contacts age, email, phone number, name
- A has contact function to see if they are in there

2. Planning

In our planning step we first went overview of what the code should do and then looked at each type of code that was wanted and tried to figure out a plan for each type of function. After that we started with the make contact which required no return statement, but it did require us to use the make key function and create that so it could add things into the contacts. For the make key function we wound up using strings and concatenation to create a key for the contacts, we then added in a way for the key to be given a value as well. for has contact function we wound up writing code that checked to see if the name was in the test dictionary called contacts, if it was it would return true else return false. for the get contacts, age email or phone number we used has contact to see if they were in our contacts dictionary, if it was true we then set it up to pull from the dictionary their age, phone number or email as needed by the code. Contacts[name] [index #]. for the update contacts age, email, or phone number, we wrote something like get contacts age, instead though we called back our has contact function, if it returned true then we set the key of name and the index that age was on equal to the new age wanted then ran a check to see if it was true. If it was true it would return true else, it would return false. For the remove contact function, we decided on using the del key word in python to delete the key and value associated with the dictionary. For this function we started with name = make key(). From there we used the has contact function to check if we had the contact, if it came up true we then had it delete the key of name so del contacts[name] and then had it return the first name and last name of the deleted contact. If we didn't have the contact, it would only return none. Through all of our coding we would stop to see if each function would run and pass the test that was given to us, if it did we would continue on and

code more otherwise we would look over our code and see where we went wrong and seek help from the instructors if we couldn't figure it out in about 10 minutes.

3. Implementation and testing

When we first implemented our plan, we were stuck on the add contact function for 30 minutes trying to get it to work, that was when we asked for help from the instructor. After that and struggling with one of each function we were able to get the code to run smoothly and pass all the tests given to us. overall by following our plan and testing after each block of code we were able to finish the project in reasonable time since we could isolate where the error might be in our code. Below is a screen shot of our codes compliance testing and it running the tests:

```
rm842@tatooine:~/cs126/labs/lab5$ ls
__pycache__/  rm842_wh259_rad449_phone_contacts.py  test_contacts.py
rm842@tatooine:~/cs126/labs/lab5$ pycodestyle-3 rm842_wh259_rad449_phone_contact
s.py
rm842@tatooine:~/cs126/labs/lab5$
```

```
rm842@tatooine:~/cs126/labs/lab5$ python test_contacts.py
               add_contact() test 1:   Pass!
               has_contact() test 1:   Pass!
               has_contact() test 2:   Pass!
          get_contact_age() test 1:   Pass!
          get_contact_age() test 2:   Pass!
        get_contact_email() test 1:   Pass!
        get_contact_email() test 2:   Pass!
  get_contact_phone_number() test 1:   Pass!
  get_contact_phone_number() test 2:   Pass!
       update_contact_age() test 1:   Pass!
       update_contact_age() test 2:   Pass!
     update_contact_email() test 1:   Pass!
     update_contact_email() test 2:   Pass!
update_contact_phone_number() test 1:   Pass!
update_contact_phone_number() test 2:   Pass!
rm842@tatooine:~/cs126/labs/lab5$
```

4. Reflection and Refactoring

Looking back on our code we believe that what we created works for what was asked. Our code is able to create new contacts, check to see if a contact is in the contacts dictionary, get certain things from contacts and update them as well. We are also able to remove contacts and give the name of the contact removed. If we could have used while or for loops we would have but since we just went over them and are still going over loops it is hard to implement them properly in our code, but if we were able to implement them properly it could help reduce the amount of checking true or false statements throughout our code. Along with this is there a way to code to run more than one thing at once, is it possible for the code to create a contact and right after check to see if it's part of

the contacts without having to write two different functions. Overall, we believe our code is well optimized for where we are in the class and our coding careers.