Python Abilities:

- Write a complete Python program.
- Test and verify a Python program.
- Analyze a basic set of algorithms.
- Identify the object oriented programming using Python.
- Analyze and design basic algorithms.
- Apply the phases of software development to write Python programs that solve a scientific problem. Apply the process of testing and debugging a Python program.
- Structure a program using decomposition and abstraction, including specification and parameter. Identify object-oriented programming paradigm in Python.
- Explore and identify some of the classic optimization problems.
- Develop computational methods for data modeling and analysis using Python Packages such as Numpy, Matplotlib, and IPython.
- Skillful in getting the right output

(1) VECTOR CLASS

```
 #lab 09 Dianna Ulm CS61002, make a vector class
#constructor, takes 3 args:self, x and y
#string, for printing a string
#provide the operators, add (vector+vector), sub(vector-vector),
#multiply(vector*vector), and magnitude (Pythagorean theorem)
import math
class Vector:
    def __init__(self, x=0, y=0):
        self.x=float(x)
        self.y=float(y)
    def __str__(self):
        return "self.x","self.y"
    def __add__(self, other):
        a=self.x+other.x
        b=self.y+other.y
        return Vector(a,b)
    def __sub__(self, other):
        a=self.x-other.x
        b=self.y-other.y
        return Vector(a,b)
    def __mul__(self, other):
        a=self.x*other.x
        b=self.y*other.y
        return Vector(a,b)
    def magnitude(self,other):
        return (self.x**2+other.y**2)**0.5
a=Vector(1,2)
b=Vector(3,4)
c=a+b
d=a-b
e=c*a
```

(2) CREATE A USER PROMPT, READ,WRITE & RETRIEVE, DICTIONARY

```
 >>> #import random & glob for Project One Dianna Ulm CS61002
import random
import glob
#set a variable to zero so that later you can set the global variable
n=0
#create a file name for future distinction in glob
##create an assignment to allow glob to retrieve all text files
testone=glob.glob('*.txt')
print testone
#error check with assert function
assert len(testone)!=0,"no file found"
#create a for loop
for files in testone:
    print files
#create a function to prompt the user to choose a glob file
#note, currently there is only one choice, the verbs text file
#for test taking, raw input to prompt the user to enter the file
filename=raw_input(["Enter the File Name"])
print filename
#create a function to open and read the file so that glob can open
##for the user, strip and split to display all then english then
##spanish (close later)
###Creating a list of data inside the text file, convert it into a dictionary
def readuserfile(filename):
    mydictionary=dict()
    with open(filename,'r') as file:
        for line in file.readlines():
            verbenglish,verbspanish = line.strip().split(':',1)
            mydictionary[verbenglish]=verbspanish
            verbslist = line.strip().split(':')
    return mydictionary
#create the function for the test, declare the variable global and
##allow the iteration for the dictionary using iteritems so verbs can be
###retrieved one by one, test user and write the incorrect answers into a file
def Test(spanishverbtest):
        keyfile=[]
## ## ## ##
valuefile=[]
numbertotest=int(input("Enter the number you want to answer?"))
assert numbertotest<=len(spanishverbtest)
for questions,answers in spanishverbtest.iteritems():
        global n
        print "What is the Spanish word for this verb?:" + questions
        userresponse1=raw_input("Enter your answer:")
        if userresponse1==answers:
                print "That's Correct"
n=n+1 else:
                print "That's not right"
                keyfile.append(questions)
                valuefile.append(answers)
                n=n+1
                if n>= numbertotest:
                        print "Test is finished"
                        writtenfiles = dict(zip(keyfile,valuefile))
                        if len(writtenfiles)>= 1:
                                writewrongones=open('notgood.txt','w')
      for questions,answers in writtenfiles.iteritems():
          wrong=questions+":"+answers+'/n'
          writewrongones.write(wrong)
          writewrongones.close()
readuserfile(filename)
z=readuserfile(filename)
print(z)
```

(3) DNA SEQUENCE SEARCH

```
 #Dianna Ulm CS61002 Lab 08 Matching DNA
#Prompt for two strings, Biological consist of DNA specific
DNAstring1=raw_input("Please enter a DNAstring1 using ('A','C','G','T') :")
DNAstring2=raw_input("Please enter a DNAstring2 using ('A','C','G','T') :")
#prompt user the four commands a,d,s,and q using while one statement
#length of the strings, also index the strings
#If statement for the choices from the user, set range for length of strings
#Adding and deleting an indel, then sorting an indel
while(1):
    print("a. is for adding an indel")
    print("d is for deleting an indel")
    print("s is for score")
    print("q is for quit")
    userstr1=len(DNAstring1)
    userstr2=len(DNAstring2)
    UserIndel=raw_input("Enter either 'a','d','s', or 'q' :")
    if(UserIndel=='a'):
        masterstring="";
        print("1. Adding an indel to DNAstring1 :")
        print("2. Adding an indel to DNAstring2 :")
        StringSelection4indel=raw_input("Are you adding indel to 1 or 2? :")
        IndelPlace=raw_input("Enter the index where you want to add the indel? :")
        index=int(IndelPlace);
        select=int(StringSelection4indel);
        if(select==1):
            if(index>userstr1-1):
                print("ERROR");
else:
wantone=1
for wantone in range(0,userstr1):
    masterstring+=DNAstring1[wantone];
    if (wantone==index):
        masterstring=masterstring[:wantone]+'-'+DNAstring1[wantone];
DNAstring1=DNAstring1[:0]+masterstring;
        elif(select==2):
             if(index>userstr2-1):
                print("ERROR");
             else:
                wanttwo=2
                for wanttwo in range(0,userstr2):
                    masterstring+=DNAstring2[wanttwo];
                    masterstring=masterstring[:wanttwo]+'-'+DNAstring2[wanttwo];
                DNAstring2=DNAstring2[:0]+masterstring;
        print ("Printing the strings after adding the indel")
        print (masterstring)
```

```
##Deleting an indel
    if(UserIndel=='d'):
        masterstring="";
        print("1. Deleting an indel to DNAstring1 :")
        print("2. Deleting an indel to DNAstring2 :")
        StringSelection4indel=raw_input("Are you deleting indel on 1 or 2? :")
        IndelPlace=raw_input("Enter the index where you want to delete the indel? :")
        index=int(IndelPlace);
        select=int(StringSelection4indel);
        if(select==1):
                for delone in range(0,userstr1):
                    if (DNAstring1[delone]!='-'):
                        masterstring+=DNAstring1[delone];
                DNAstring1=DNAstring1[:0]+masterstring;
        elif(select==2):
             for deltwo in range(0,userstr2):
                if (DNAstring2[deltwo]!='-'):
                    masterstring+=DNAstring2[deltwo];
             DNAstring2=DNAstring2[:0]+masterstring;
        print ("Printing the strings after deleting the indel")
        print (masterstring)
##Sorting an indel, create mismatch, fill shorter string with indels
##also create empty strings necessary
##after scoring, print both strings, matching is printed in lower case
##mismatch to be printed in upper case, using built in python function
##for lower and upper
    elif(UserIndel=='s'):
        shortstring="";
        notshort="";
        matchmatch="";
        notmatch="";
        if(userstr1>userstr2):
           notshort=notshort[:0]+DNAstring1;
           shortstring=shortstring[:0]+DNAstring2;
        else:
            notshort=notshort[:0]+DNAstring2;
            shortstring=shortstring[:0]+DNAstring1;
        for step5 in range(0,len(notshort)):
           shortstring=shortstring[:len(shortstring)]+"-";
        print("shortstring")
        print("notshort")
        for step5 in range(0,len(notshort)):
            if((notshort[step5]==shortstring[step5]) &(notshort[step5]!='-')):
                matchmatch=matchmatch+notshort[step5].lower();
            elif(notshort[step5]!='-'):
                notmatch=notmatch+notshort[step5].upper()
        print ("The characters that are equal are")
        print ("matchmatch")
        print ("The characters that are not equal are")
        print ("notmatch")
##Quit
    if(UserIndel=='q'):
exit();
```