Name: Rohit Abhishek
Course: CSEE 5590 Python/ Deep learning
Assignment 1

1 Problem 1

For any web application login, the user password need to be validated against database rules.
For My UMKC web application following are the criteria for valid password:
a) The password length should be in range 6-16 characters
b) Should have atleast one number
c) Should have atleast one special character in [$@!*]
d) Should have atleast one lowercase and atleast one uppercase character
Use loops to write a python program for the above scenario

1.1 Solution

1.1.1 Approach

Since we need to perform multiple operations to check the validity of the password, we can create function for each of them. The following functions needs to be created.

- Function to check if the input string has any integer (num_in_string)
- Function to check if the input string has atleast on lower and uppercase characters(lower_case, upper_case)
- Function to ask the user to input the password(input_user)
- Function to ask the user if he wants to continue (ask_count)
- Main function (main())

The main function will responsible for asking the user for the input password. Once the user inputs the password, the main function would call other function to check for each of the validity criteria for the password. If the password does not satisfy any of the criteria, it would display which all criteria has not been satisfied and would ask the user if he wants to continue. If the user wants to continue, user would be asked to input the password again and repeat the process.

The below function will be used to get the input from the user

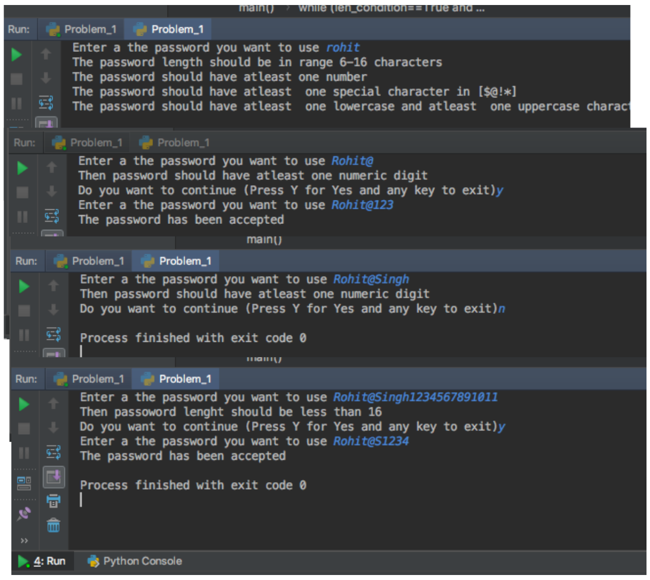
```
def input_user():#function to get input from the user
    password=raw_input("Enter a the password you want to use ")
    return password
```

The below function will be used to check if the password has any integers

```
def num_in_string(password): #function to check for integer in the input password
    return any(i.isdigit() for i in password)
```

The below function will be used to check if the password has any lower case. If lower case is present, it would return True

```
def lower_case(password): #function to check for lower case in the input password
    for word in password:
        if word.islower():
            return "True"
```



```
Enter a the password you want to use rohit
The password length should be in range 6-16 characters
The password should have atleast one number
The password should have atleast  one special character in [$@!*]
The password should have atleast  one lowercase and atleast  one uppercase charact
```

```
Enter a the password you want to use Rohit@
Then password should have atleast one numeric digit
Do you want to continue (Press Y for Yes and any key to exit)y
Enter a the password you want to use Rohit@123
The password has been accepted
```

```
Enter a the password you want to use Rohit@Singh
Then password should have atleast one numeric digit
Do you want to continue (Press Y for Yes and any key to exit)n

Process finished with exit code 0
```

```
Enter a the password you want to use Rohit@Singh1234567891011
Then passoword lenght should be less than 16
Do you want to continue (Press Y for Yes and any key to exit)y
Enter a the password you want to use Rohit@S1234
The password has been accepted

Process finished with exit code 0
```

Figure 1: Problem 1 output

The below function will be used to check if the password has any upper case. If upper case is present, it would return True

```
def upper_case(password): #function to check for lower case in the input password
        for word in password:
                if word.upper():
                        return "True"
```

We will use the following command to check if any special character is present.

```
special_condition=(("$"not in password) and ("@" not in password) and ("!"not in password)
and ("*" not in password))
```

While loop has been used in the code to check for validity for each of the functions.

1.1.2 Code

This section presents the code for the along with comments to understand each block of the code.

```
import re

def main(): #Main function

    password=input_user() #calling the functipn to get input from the user and storing it in
         variable password
    length_password= len(password) #finding the length of the password the user has input

    #print length_password
    len_condition=(len(password)<6) or (len(password)>16) #to check whether length condition
         is true or not
    #print(len_condition)
```

```
num_condition=(num_in_string(password)!=True)#to check whether numeric condition is true
     or not
#print(num_condition)
special_condition=(("$"not in password) and ("@" not in password) and ("!"not in
    password) and ("*" not in password))#to check whether special character condition is
     true or not
#print(special_condition)
lowercase_cond=((lower_case(password)!="True") or ( upper_case(password)!="True")) #to
    check whether lower case condition is true or not
#print(lowercase_cond)
#print(lower_case(password))
#print(upper_case(password))
#print lowercase_cond

#loop to check which all condition is not satisfied and accordingly asking the user to
    input password
while (len_condition==True and num_condition== True and special_condition==True and
    lowercase_cond==True):
    print("The password length should be in range 6-16 characters\n"
          "The password should have atleast one number\n"
          "The password should have atleast one special character in [$@!*]\n"
          "The password should have atleast one lowercase and atleast one uppercase
              character\n")
    password=ask_cont()

while (num_condition== True and lowercase_cond==True):
    print(
          "The password should have atleast one number\n"
          "The password should have atleast one lowercase and atleast one uppercase
              character\n")
    password=ask_cont()

while (len_condition==True and num_condition== True and special_condition==True ):
    print("The password length should be in range 6-16 characters\n"
          "The password should have atleast one number\n"
          "The password should have atleast one special character in [$@!*]\n")
    password= ask_cont()



while (len_condition==True and num_condition== True) :
    print("The password length should be in range 6-16 characters\n"
          "The password should have atleast one number\n")
    password=ask_cont()

while (num_condition== True and special_condition==True and lowercase_cond==True):
    print(
          "The password should have atleast one number\n"
          "The password should have atleast one special character in [$@!*]\n"
          "The password should have atleast one lowercase and atleast one uppercase
              character\n")
    password=ask_cont()

while (special_condition==True and lowercase_cond==True):
    print(
            "The password should have atleast one special character in [$@!*]\n"
            "The password should have atleast one lowercase and atleast one uppercase
                character\n")
        password=ask_cont()

    while (num_condition== True and special_condition==True ):
        print(
            "The password should have atleast one number\n"
            "The password should have atleast one special character in [$@!*]\n"
            )
        password=ask_cont()

    while (len_condition==True and lowercase_cond==True):
        print("The password length should be in range 6-16 characters\n"
            "The password should have atleast one lowercase and atleast one uppercase
                character\n")
        password=ask_cont()



    #checking the password range is between 6-16 and then printing out whether the password
        is greater or less.
    # In either case it will ask the user if he wants to continue and input the password
        again
    while ((len(password)<6) or (len(password)>16)) :
        if (len(password)<6):
            print "Then passoword lenght should be greater than 6"
        elif (len(password)>16):
            print "Then passoword lenght should be less than 16"
        password=ask_cont()
    #calling the fucntion to check if the input password has integer.
    # If not it will ask the user if he wants to continue and input the password again
    while (num_in_string(password)!=True) :
        print "Then password should have atleast one numeric digit"
        password=ask_cont()
    #calling the fucntion to check if the input password has atleast one special characters.
    # If not it will ask the user if he wants to continue and input the password again
    while (("$"not in password) and ("@" not in password) and ("!"not in password) and ("*"
        not in password)) :
        print "Then password should have atleast one special characters ($,@,!,*)"
        password=ask_cont()
    #calling the fucntion to check if the input password has atleast one upper case and one
        lower case.
    # If not if will ask the user if he wants to continue and input the password again
    while lower_case(password)!="True" or upper_case(password)!="True":
        if lower_case(password)!="True":
            print "The password should have atleast on lower case"
        elif upper_case(password)!="True":
            print "The password should have atleast on upper case"
        password=ask_cont()
    print "The password has been accepted"

def num_in_string(password): #function to check for integer in the input password
    return any(i.isdigit() for i in password)
```

```
def input_user():#function to get input from the user
    password=raw_input("Enter a the password you want to use ")
    return password

def lower_case(password): #function to check for lower case in the input password
    for word in password:
        if word.islower():
            return "True"

def upper_case(password):#function to check for upper case in the input password
    for word in password:
        if word.isupper():
            return "True"
def ask_cont(): #function to ask the user if he wants to continue after entering the wrong
    password
    asktocontinue=raw_input("Do you want to continue (Press Y for Yes and any key to exit)")
    if (asktocontinue=="Y" ) or (asktocontinue=="y" ):
        return input_user()
    else:
        exit()

main() #calling the main function
```

2 Problem 2

Write a Python function that accepts a sentence of words from user and display the following:
a) Middle word
b) Longest word in the sentence
c) Reverse all the words in sentence

2.1 Solution

2.1.1 Approach

For this problem we need to create 4 different functions.

- Function to get the middle word (middle_word)
- Function to get the longest word (longest_word)
- Function to reverse the word (reverse_word)
- Main function (main)

The function middle_word will print the middle word of the sentence. First it would check the length of the string, if it is 1 or 2 then it would print all of the string else it would check if the len of the string is odd or even. If the length is even there will be 2 middle words. whereas if the length is odd there would be one middle word. Below is the code for the function

```
def middle_word(string_split,len_string):
        if len_string==2: #if the string has lenght 2 then both will be considered as middle words
                print ("The middle word is : %s %s" %(string_split[0],string_split[1]))

        elif len_string % 2 ==0: #if the len is divisible by 2, there should be 2 middle words
                print ("The middle words are :%s%s"%(string_split[(len_string/2)-
1],string_split[(len_string/2)]))
        else:
                print ("The middle word is : %s" %string_split[(len_string/2)])
```

The function longest_word would print the check length of each of the sub-string and store the length and the sub-string inside an array and then sort it. So the last item would give us the longest sub-string. The second loop in the function would check if we have more than one sub-string. If so it would print out all the them. Below is the code for the same.

```
def longest_word(string_split,len_string):
#iterating through the splitted string to find the length of each word and store the len and the words
in another 2 dimensional array long_word_lst
        long_word_lst=[]
        for i in range(len_string):
                long_word_lst.append([len(string_split[i]),string_split[i]])

#sort the long_word_lst based on the len of the words. the last item should be the longest item
long_word_lst=sorted(long_word_lst)
#print(long_word_lst)
#long_word_lst_tmp=[]

        print ("The longest word is: ",end="")
#checking if there is any other item in the string of the same length by comparing it to the last item
since it is the longest
        for i in range(len_string):
                if long_word_lst[len_string-1][0]==long_word_lst[i][0]:
#long_word_lst_tmp.append(long_word_lst[i][1])
                        print(long_word_lst[i][1]," ",end=" ")
        print(" ")
```

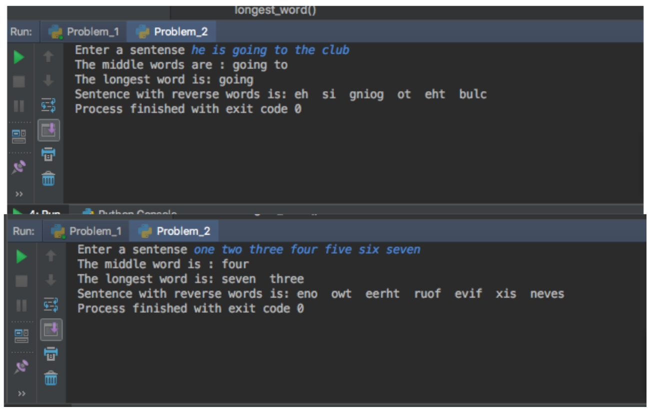
Below is the function to reverse the word. The loop in the function selects each of the sub-string and reverses it.

```
def reverse_word(string_split,len_string):
        print ("Sentence with reverse words is: ",end='')
        for i in range(len_string):
                print (string_split[i][::-1]," ",end='')
```

2.1.2 Code

This section presents the code for the along with comments to understand each block of the code.

```
from __future__ import print_function

#function to abstract the middle word
def middle_word(string_split,len_string):
    if len_string==2: #if the string has lenght 2 then both will be considered as middle
        words
        print ("The middle word is : %s %s" %(string_split[0],string_split[1]))

    elif len_string % 2 ==0: #check if the len is divisible by 2, if yes then there should
        be 2 middle words
        print ("The middle words are : %s %s"%(string_split[(len_string/2)-1],string_split[(
            len_string/2)]))
    elif len_string % 2 ==1:
        print ("The middle word is : %s" %string_split[(len_string/2)])

#fucntion to get the longest word
def longest_word(string_split,len_string):
    #iterating through the splitted string to find the length of each word and store the len
         and the words in another 2 dimensinal array long_word_lst
```

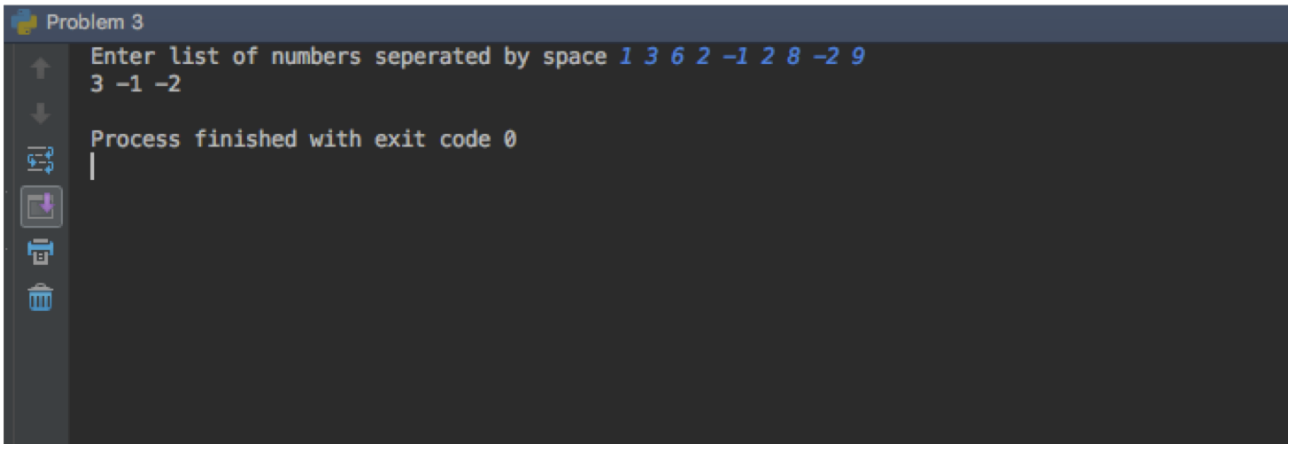


```
Enter a sentense he is going to the club
The middle words are : going to
The longest word is: going
Sentence with reverse words is: eh  si  gniog  ot  eht  bulc
Process finished with exit code 0
```

```
Enter a sentense one two three four five six seven
The middle word is : four
The longest word is: seven  three
Sentence with reverse words is: eno  owt  eerht  ruof  evif  xis  neves
Process finished with exit code 0
```

Figure 2: Problem 2 output

```
    long_word_lst=[]
    for i in range(len_string):
        long_word_lst.append([len(string_split[i]),string_split[i]])

    #sort the long_word_lst based on the len of the words. the last item should be the
        longest item
    long_word_lst=sorted(long_word_lst)
    #print(long_word_lst)
    #long_word_lst_tmp=[]

    print ("The longest word is: ",end="")
    #checking if there is any other item in the string of the same lenghth by comparing it
        to the last item since it is the longest
    for i in range(len_string):
        if long_word_lst[len_string-1][0]==long_word_lst[i][0]:
            #long_word_lst_tmp.append(long_word_lst[i][1])
            print(long_word_lst[i][1]," ",end='')
    print(" ")

#fuction to reverse a word
def reverse_word(string_split,len_string):
    print ("Sentence with reverse words is: ",end='')
    for i in range(len_string):
        print (string_split[i][::-1]," ",end='')

#main function
def main():
    user_input=raw_input("Enter a sentense ")
    string_split=user_input.split(" ") #to split the string
    len_string=len(string_split)#calculating the len of the string
    middle_word(string_split,len_string) #calling the fucntion to get the middle_word
    longest_word(string_split,len_string) #calling the function to get the longest word
    reverse_word(string_split,len_string) #calling the function to reverse the word

main() #calling the main function
```



```
Enter list of numbers seperated by space 1 3 6 2 -1 2 8 -2 9
3 -1 -2

Process finished with exit code 0
```

Figure 3: Problem 3 output

3 Problem 3

Given a list of n number, write a Python program to find triplets in the list which gives the sum of zero.

3.1 Solution

3.1.1 Approach

Since we need to find triplets, so we will need to use three loops to iterate through the list of the numbers. At the same time we need to make sure that the combination is not repeated. For this problem a function triplets has been defined which would run the loops to find the combination of triplets. The range for each of the loops vary as we don't want the combinations to be repeated

3.1.2 Code

This section presents the code for the along with comments to understand each block of the code.

```
def triplets(user_input):#fucntion to find the triplet
    string_split=user_input.split(" ") #splitting the input string
    string_list=[]
    #we will iterate through the string and check if he sum of triplets will be zero. Since
        we have triplets so we will be using 3 lops.
    #the loop varies in all three loops as we dont want to repeat combination
    for i in range(0,len(string_split)-2):
        for j in range(i+1,len(string_split)-1):
            for k in range(j+1,len(string_split)):
                if int(string_split[i]) +int(string_split[j])+int(string_split[k])==0:
                    print string_split[i] , string_split[j] , string_split[k]

user_input= raw_input("Enter list of numbers seperated by space ") #Assuming the input will
    be given by user
triplets(user_input)#calling the main function
```

4 Problem 4

Consider the following scenario. You have a list of students who are attending class "Python" and another list of students who are attending class "Web Application".
Find the list of students who are attending both the classes. Also find the list of students who are not common in both the classes. Print it.

4.1 Solution

4.1.1 Approach

For this approach we need to create 3 function:

- To calculate the common students in both the classes (common_students)
- To calculate the uncommon students in both the classes (not_common_students)
- Main (main)

The function to calculate the common students is shown below. We create two loops, one to iterate through list of students in python class and other for web application class. We take each item from python class and check if it is present in web application class. If present we store it in an array.

```
def common_students(python_student_list,web_application_list):
#function to calculate the common students
        common_student=[] #list of common students

        total_student=[] #list of total students

#loop to find the common students
        for names_python in python_student_list:
                for names_web in web_application_list:
                        if names_python== names_web:
                                common_student.append(names_web)
        return common_student
```

The function to calculate the uncommon students is shown below. We create two loops, one to iterate through list of students in python class and other for web application class. We take each item from python class and check if it is present in web application class. If not present we store it in an array. The same iteration is done for web application class to check if the item in web application class is present in python class. If not than store it the same array where the items from python class which were not present in web application class were stored

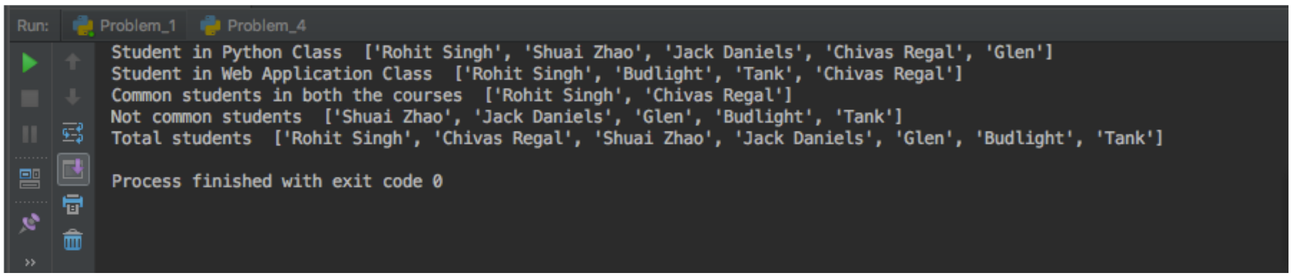
```
def not_common_students(python_student_list,web_application_list):
#function to calculate the uncommon students
    notcommon_student=[] #list of uncommon students
#loop to find the students python students who are not attending web application class
    for i in python_student_list:
        if i not in web_application_list:
            notcommon_student.append(i)

#loop to find the students web application students who are not attending python class
    for j in web_application_list:
        if j not in python_student_list:
            notcommon_student.append(j)
    return notcommon_student
```

The total students list attending both the classes is calculated by adding the common students and uncommon students.

4.1.2 Code

This section presents the code for the along with comments to understand each block of the code.

```
def common_students(python_student_list,web_application_list): #function to calculate the
    common students
   common_student=[] #list of common students

   total_student=[] #list of total students

   #loop to find the common students
   for names_python in python_student_list:
       for names_web in web_application_list:
           if names_python== names_web:
               common_student.append(names_web)
   return common_student
def not_common_students(python_student_list,web_application_list):#function to calculate the
     uncommon students
   notcommon_student=[] #list of uncommon students
   #loop to find the students python students who are not attending web application class
   for i in python_student_list:
       if i not in web_application_list:
           notcommon_student.append(i)

   #loop to find the students web application students who are not attending python class
   for j in web_application_list:
       if j not in python_student_list:
           notcommon_student.append(j)
   return notcommon_student

def main(python_student_list,web_application_list): #main function
   print "Common students in both the courses ", common_students(python_student_list,
       web_application_list) #calling the common fucntion
   print "Not common students ", not_common_students(python_student_list,
       web_application_list)#calling the uncommon fucntion
```



```
Student in Python Class  ['Rohit Singh', 'Shuai Zhao', 'Jack Daniels', 'Chivas Regal', 'Glen']
Student in Web Application Class  ['Rohit Singh', 'Budlight', 'Tank', 'Chivas Regal']
Common students in both the courses  ['Rohit Singh', 'Chivas Regal']
Not common students  ['Shuai Zhao', 'Jack Daniels', 'Glen', 'Budlight', 'Tank']
Total students  ['Rohit Singh', 'Chivas Regal', 'Shuai Zhao', 'Jack Daniels', 'Glen', 'Budlight', 'Tank']

Process finished with exit code 0
```

Figure 4: Problem 4 output

```
    print "Total students ", common_students(python_student_list,web_application_list)+
        not_common_students(python_student_list,web_application_list) #to caculate the total
         students


python_student_list=['Rohit Singh', 'Shuai Zhao', 'Jack Daniels','Chivas Regal','Glen'] #
    list of student in python class
web_application_list=['Rohit Singh','Budlight','Tank','Chivas Regal'] #list of students in
    web application class
print "Student in Python Class ", python_student_list
print "Student in Web Application Class ",web_application_list

main(python_student_list,web_application_list)#calling the main function
```