Unit 4 Career Preparation: Technical Assessment

Problem 1

Write a script that:

- Reads the file `problem1.txt`.
- Adds each line to a new list.
- Prints the new list.

In [3]:
```
import os
file_name = "/voc/data/problem1.txt"
with open(file_name, 'r') as f:
    for line in f.readlines():
        print(line)
```

```
item1

item2

item3

item4

item5
```

Problem 2

Write a script that:

- Reads the file `problem2.txt`.
- Counts how many times 192.168.1.1 appears in the file.
- Prints the result.

In [8]:
```
import os
file_name = "/voc/public/problem2.txt"
with open(file_name, "r") as f:
    lines = []
    for line in f:
        line = line.strip()
        lines.append(line)
    counter = 0
    for IP in lines:
        if IP == "192.168.1.1":
                counter +=1
print(counter)
```

```
5
```

Problem 3

Write a script using a function (`dedupe`) that:

- Takes a list `l = [1,5,7,2,4,3,5,1,6,2,6]` .
- Returns a new list that contains all of the elements from the first list, excluding duplicates.

In [25]:
```
def dedupe(lst):
    new_list = []
    for i in lst:
        if i not in new_list:
            new_list.append(i)

    return new_list

l = [1,5,7,2,4,3,5,1,6,2,6]

print(dedupe(l))
```

```
[1, 5, 7, 2, 4, 3, 6]
```

Problem 4

Write a program (using a function) that: Asks the user for a long string containing multiple words. Prints back the same string, except with the words in reverse order.

For example, if the user types the string: 'My name is robert', it will print 'robert is name My'.

In [2]:
```
def reverse_word(string):
    return ' '.join(string.split(' ')[::-1])
print(reverse_word(input("Please enter a string with multiple words:")))
```

```
 Nii is name My
```

Problem 5

Write a script that:

- Opens the file `problem5.txt`.
- Counts each port and puts the results in a dictionary.

In [12]:
```
import os

with open("/voc/public/problem5.txt") as f:

    lines = []
    for line in f:
        line = line.strip()
        lines.append(line)

    Dict_1 = {}
    for port in lines:
        if port not in Dict_1:
            Dict_1[port] = 1
        else:
            Dict_1[port] +=1
    print(Dict_1)
```

```
{'80': 7, '443': 3, '22': 5, '21': 2, '25': 3, '389': 1, '3389': 1, '445': 3, '': 1}
```

In [ ]:
```

```

Unit 4: Challenge

Problem 1

1. Open the `exam.log` file.

2. Write a function `ip_result` that:

- Searches for lines with IP
- Counts the number of each IP
- Puts the results in a dictionary
- Sorts the dictionary
- Puts the results into a file

3. Write a function `invalid_user_count` that:

- Searches for invalid user logins
- Counts the invalid logins for each user
- Puts the results in a dictionary
- Sorts the dictionary
- Puts the result into a file

4. Write a function `failed_logins` that:

- Searches for wrong passwords
- Counts the failed logins
- Puts the results in a dictionary
- Sorts the dictionary
- Puts the result in a file

5. Call the functions

In [12]:

```
import re
def ip_result(f):

    dict1={}
    for line in f.readlines():
        list_line=line.split()
        match=re.search(r"\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}[^0-9]",line)
        if match:
            ip=match.group(0)
            ip=ip.strip("\n")
            #print(ip)
            if ip in dict1:
                dict1[ip]+=1
            else:
                dict1[ip]=1
    list1=sorted(dict1.items(), key=lambda x: x[1])

    f.seek(0)
    with open("ip_result.txt","w") as s:
        for line in list1 :
            s.writelines("IP: "+line[0] + ", Count: "+str(line[1])+"\n")


def invalid_user_count(f):
    dict2={}
    for line in f.readlines():


        if "Invalid user" in line:
            #print(line)

            user=line.split()[7]
            if user in dict2:
                dict2[user]+=1
            else:
                dict2[user]=1
    list1=sorted(dict2.items(), key=lambda x: x[1], reverse=True)
    f.seek(0)
    with open("Invaliduser.txt","w") as s:
        for line in list1 :
            s.writelines("USER: "+line[0] + ", Count: "+str(line[1])+"\n")

def failed_logins(f):
```

```
    dict3={}
    for line in f.readlines():
        if "Failed" in line and "invalid" not in line and "message" not in line:

            user=line.split()[8]
            if user in dict3:
                dict3[user]+=1
            else:
                dict3[user]=1
    list1=sorted(dict3.items(), key=lambda x: x[1], reverse=True)
    f.seek(0)
    with open("failed_logins.txt","w") as s:
        for line in list1 :
            s.writelines("User: "+line[0] + ", Count: "+str(line[1])+"\n")


def main():
        with open("/voc/public/exam.log") as f:
            invalid_user_count(f)
            ip_result(f)
            failed_logins(f)

main()
```

Problem 2

Analyze the following code that reads the `apache_logs.txt` file. Determine what it does. Write your response as code comments.

In [ ]:

```
import sys  # Importing the system module, allows us to work directly with our sys modules
import os   #Importing the operating system module, allows us to work directly with our os modules

def readfile(f): # Defining a function named readfile
    openfile = open(f,"r") # Creating a variable named, "openfile" to open up the file, f, in read mode.

    unique_outfile = open("uniqueIP.txt","w") # Creating a variable named, "unique_outfile" to open the "uniq
    all_outfile = open("allIP.txt","w")       # Creating a variable named, "all_outfile" that opens the file,
    ipAndUrl_outfile = open("ipAndUrl.txt","w") # Creating a variable named, "ipAndUrl_outfile" that opens th

    lines = []      # Creating a list named lines that is empty.
    ipAndUrl = {}   # Creating an empty dictionary named ipAndUrl.
```

```
    ip_list = set() # Creating a set of objects named ip_list.

    for line in openfile: # We iterate through each line in our openfile, line is our variable for what we ar
        lines.append(line.strip('\n')) # Removing any new line characters from the begining and the end of ea

    for line in lines: # We iterate through each line in our list, called lines, line is our variable for wha
        ip = line.split(" ")[0] # Splitting each line from the the list of lines by spaces to form a string a

        if ip in ipAndUrl:  # If we find the ip address in the dictionary we created called ipAndUrl
            ipAndUrl[ip].append(line.split(" ")[6]) # If we do find the ip address as the key in the dictiona
        else:               # If we do not find the ip address as the key in the dictionary
            ipAndUrl[ip] = [line.split(" ")[6]] # Create a new dictionary entry with the ip address as the ke

        all_outfile.write(ip) # We are actually writting (or appending) the ip address to the "allIP.txt" fil
        all_outfile.write("\n") # Writing a newline character to the "allIP.text" file.

        ip_list.add(ip)  # Adding ip addresses to the set named ip_list using the add method

    for ip in ip_list:   # We iterate through each ip address in the set of ip_list, ip address is our variab
        unique_outfile.write(ip) # We are actually writting (or appending) the ip address to the "uniqueIP.tx
        unique_outfile.write("\n") # Writing a newline character to the "uniqueIP.text" file.

    for key, value in ipAndUrl.items(): # We iterate through each key,value in the dictionary
        ipAndUrl_outfile.write('%s  %s\n' % (key, value)) # Writing a dictionary to the ipAndUrl.txt file as

    unique_outfile.close() # Closing the uniqueIP.txt file
    ipAndUrl_outfile.close() # Closing the ipAndUrl.txt file
    all_outfile.close()      # Closing the allIP.txt file
    openfile.close()         # Closing the "f" file

def Main():    # Defininig a function named main which requires no additional information
    file=input("please enter a file name: ") # Asking the user for a file name, and saving it to "file"

    result = readfile(file) # We are assigning the variable name result to our defined function, readfile.

if __name__ == '__main__': # This is to help execute and call the main function as a script from the command

    Main()   # calling our main function to kick start our script
```