*#Need to find the file in via ideally a relative location rather than absolute location***from** os **import** listdir  
*#this is an absolute pathway*path =**"C:/Users/Amanda/Documents/UCL/Computing for Psychologists/Assessment 1/results/"**files =listdir(path)  
  
print(files)  
  
**for** filenames **in** files:  
 **if "csv" in** filenames: *#all the files have csv so it applies to all of them* allfiles =open(path+filenames, **'r'**) *#open all the files in the read format* allLines =allfiles.readlines() *#pull all the lines from all the files* print(allLines)  
  
 *#ncontent is the modified allLines* ncontent =**""** *#ncontent is a string* **for** character **in** allLines: *#pull out the important parts from the* **if** character !=**" " and** character !=**","**: *#taking out the spaces and the commas, only taking what matters* ncontent =ncontent+character  
 print(ncontent)  
  
 nlist = ncontent.replace(**" "**,**""**)  
 nlist = list(nlist)  
 print(nlist)  
  
 **for** parts **in** nlist:  
 Condition = nlist[0]  
 AddreIP = nlist[1]  
 Name = nlist[2]  
 Age = nlist[3]  
 print(Condition + **","** + AddreIP + **","** + Name + **","** + Age)

*#Part A  
#create a csv file that contains the following  
#Condition  
#Name, Age, and Gender  
#1 for male and 2 for female  
#Proportion of hits, Near misses, and Full Misses over total number of trials  
#Mean happiness, mean willingness to continue per outcome type  
#Max and min reported happiness levels and the trial in which they occurred  
#For any repeat participants, record their first but ignore the second  
  
#DO A LIST OF DICTIONARIES... EACH DICTIONARY IS A PARTICIPANT...THE KEY WILL BE THE IP ADDRESS (TO AVOID REPEAT NAME PROBLEM  
#\*WILL STILL NEED TO ADDRESS THE REPEAT IP ADDRESS AND ONLY USE THE FIRST SET, NOT THE SECOND.  
#DOING A LIST OF DICTIONARIES.. CAN LOOP THROUGH THE LIST TO PULL EACH PARTICIPANT(DICTIONARY) INFORMATION  
#Need to find the file in via ideally a relative location rather than absolute location***from** os **import** listdir  
*#this is an absolute pathway*path =**"C:/Users/Amanda/Documents/UCL/Computing for Psychologists/Assessment 1/resultTest/"**files =listdir(path)  
  
print(files)  
  
**for** filenames **in** files:

**if "csv" in** filenames: *#all the files have csv so it applies to all of them* allfiles =open(path+filenames, **'r'**) *#open all the files in the read format* allLines =allfiles.readlines() *#pull all the lines from all the files* print(allLines)  
  
 **for** i **in** allLines:  
 **for** parts **in** allLines:  
*#1. Condition* Condition =allLines[0]  
 AddreIP =allLines[2]  
 AddreIP =AddreIP.replace(**"IP address: "**,**""**) *#remove the "IP address: " to get just the numbers. Easier to compare.  
  
#2. Name, Age, Gender* NAGgroup =allLines[3] *#Need to seperate the name, age, and gender* NAGgroup =NAGgroup.split(**","**)  
 Name =NAGgroup[0]  
 Age =NAGgroup[1]  
 Categories = allLines[4]  
 Gender =NAGgroup[2]  
  
*#3. Gender; Male = 1, Female = 2* Gender =Gender.upper()  
 **if** Gender ==**"FEMALE\n"**:  
 Gender =**"2"  
 elif** Gender ==**"MALE\n"**:  
 Gender =**"1"  
 else**:  
 print(**"error"**)  
 *#print(AddreIP)  
 #print(Condition+","+AddreIP+","+Name+","+Age+","+Gender)  
  
#4. Proportion of Hits, Near misses, Full misses over total number of trials* trials =allLines[5:]  
 Trials =str(trials)  
Trials =Trials.upper()  
Trials =Trials.replace(**"'"**, **""**) *#Need to clean up the parts of the list*Trials =Trials.replace(**"["**, **""**)  
Trials =Trials.replace(**" "**,**""**)  
Trials =Trials.split(**","**)  
print(Trials)  
  
wordTrial =Trials[0]  
print(wordTrial)

*#happiList =[]  
 #for Trials in allLines[5:]:  
 #splitTrials =Trials.split(",")  
 #if splitTrials[0] =="TRIAL":  
 #happi =splitTrials[7]  
 #happiList.append(happi)  
 #happiList =[int(i) for i in happiList]  
 #print(happiList)  
  
 #happiMax =max(happiList)  
 #happiMin =min(happiList)  
  
 #trialMax =happiList.index(happiMax)+1  
 #trialMin =happiList.index(happiMin)+1  
  
 #print(happiMax +trialMax)  
 #print(happiMin +trialMin)  
  
 #wordTrial =Trials[0]  
 #test2 =Trials[1] #don't need index 1,2, and 3 so block out for now  
 #test3 =Trials[2]  
 #results =Trials[4]  
 #happi =Trials[7] #I want to pull from all the trials, create a list/ must be list because need to be in ORDER  
 #will =Trials[8] # I want to pull from all the trials, create a list/ must be list because need to be in ORDER  
 #print(wordTrial)  
 #want to group the trial#, results, happiness, and willingness together.. a list of lists or dictionary  
  
#Need to pull the results for the participant* countTrials =Trials.count(**"TRIAL"**)  
 *#print(numTrials)  
 #numTrials = #number the trials... trial1, trial2, trial 3 etc.  
  
 #numHits =Trials.count("HIT")  
 #numNearMiss =Trials.count("NEARMISS")  
 #numFullMiss =Trials.count("FULLMISS")  
 #print(numFullMiss)  
#Calculate proportions (of hits, near misses, and full misses)out of #of trials  
 #numTrials =str(countTrials) #need to convert to string in order to get it to print the proportion; mixed use  
 #numHits =str(numHits)  
 #numNearMiss =str(numNearMiss)  
 #numFullMiss =str(numFullMiss)  
 #PHits =numHits+":"+countTrials  
 #PNearMiss =numNearMiss+":"+countTrials  
 #PFullMiss =numFullMiss+":"+countTrials  
 #print(PHits)  
  
#Find mean happiness, mean willingness to continue per outcome type (hit, near misses, and full misses)  
  
 #for results in Trials:  
 #if "HIT" in Trials:  
 #pull from happiness index  
 #hithappi =  
 #hitwill =  
 #pull from willingness index  
 #count in lists..  
 #elif "NEARMISS" in Trials:  
 # pull from happiness index  
 # nMisshappi =  
 #pull from willingness index  
 # nMisswill =  
 # count in lists..  
 #elif "FULLMISS" in Trials:  
 # pull from happiness index  
 # fMisshappi =  
 #pull from the willingness index  
 # fMisswill =  
 # count in lists..  
 #else:  
 #print(none)  
  
 #meanHitHappi =hithappi/numHits  
 #meanHitWill =hitwill/numHits  
 #meanNmissHappi =nMisshappi/numNearMiss  
 #meanNmissWill =nMisswill/numNearMiss  
 #meanFmissHappi =fMisshappi/numFullMiss  
 #meanFmissWill =fMisswill/numFullMiss  
  
#The Maximum and Minimum reported Happiness levels  
#and the trials they occured in #tricky... need a dictionary??? or can count the in... enumerate function the list?  
 #join hithappi, nMisshappi, fMisshappi into list?  
 #or pull from the happiness indexes from all the trials per participant and create the Happi variable  
 #Max function  
 #find the index number of the value to find which trial it was in  
 #Min function  
 #find the index number of the value to find which trial it was in  
  
 #aList =(AddreIP, Condition, Name, Age, Categories, Gender)  
 #print(aList)  
#may need to put a conditional format to prevent repeats based on the IP address. Would need to stop and skip over repeats at this point in the program  
 #participants ={  
 #AddreIP: {"Condition":aList[1], "Name":aList[2] , "Age":aList[3] , "Gender":aList[5], "Categories":aList[4], "Test": 15}  
 #}  
 #print(str(participants[AddreIP]['Test'])) #this should be printing from the participants dictionary.. use the Test key.. see pp datastructures slide 66  
 #print(str(participants)) #another test to make sure it is pulling from the dictionary  
  
  
  
  
  
#allLines contains the contents of ALL the files pulled  
#create a list from allLines  
  
#masterList =list(allLines)  
#print(masterList)  
#masterList =str(masterList)  
  
#define all the indices in allLines  
#for parts in masterList:  
 #Condition =masterList(0)  
 #IPaddre =masterList(1)  
 #Name =masterList(2)  
 #Age =masterList(3)  
 #print(Condition)  
 #print(IPaddre)  
 #print(Name)  
 #print(Age)  
 #need to assign 1 for male and 2 for female either here or later  
 #Gender =masterList(4)  
 #may need to do a dictionaries inside of the list. The keys would be T1, T2, T3...  
 #ShipT1 =  
 #AimedT1 =  
 #LandedT1 =  
 #ResultT1 =  
 #LatencT1 =  
 #MovesT1 =  
 #HappiT1=  
 #WantsMT1 =  
  
#How to pull the information from this file format  
#Create a dictionary  
#Assign names to parts of the dictionary ie condition, name, age, gender, etc*

hitHappi = []  
hitWill = []  
nMissHappi = []  
nMissWill = []  
fMissHappi = []  
fMissWill = []  
numHits = int(numHits)  
numNearMiss = int(numNearMiss)  
numFullMiss = int(numFullMiss)  
**for** Trials **in** allLines[5:]:  
 splitTrials = Trials.split(**","**)  
 **if "HIT" in** Trials:  
 HHappi = splitTrials[7]  
 HWill = splitTrials[8]  
 hitHappi.append(HHappi)  
 hitWill.append(HWill)  
 hitHappi = [int(i) **for** i **in** HHappi]  
 hitWill = [int(i) **for** i **in** HWill]  
 sumHHappi = sum(hitHappi)  
 sumHWill = sum(hitWill)  
 meanHitHappi = int(sumHHappi/numHits)  
 meanHitWill = int(sumHWill/numHits)  
 **elif "NEARMISS" in** Trials:  
 nMHappi = splitTrials[7]  
 nMWill = splitTrials[7]  
 nMissHappi.append(nMHappi)  
 nMissWill.append(nMWill)  
 nMissHappi = [int(i) **for** i **in** nMHappi]  
 nMissWill = [int(i) **for** i **in** nMWill]  
 sumNMHappi = sum(nMissHappi)  
 sumNMWill = sum(nMissWill)  
 meanNMissHappi = int(sumNMHappi/numNearMiss)  
 meanNMissWill = int(sumNMWill/numNearMiss)  
 **elif "FULLMISS" in** Trials:  
 fMHappi = splitTrials[7]  
 fMWill = splitTrials[8]  
 fMissHappi.append(fMHappi)  
 fMissWill.append(fMWill)  
 fMissHappi = [int(i) **for** i **in** fMHappi]  
 fMissWill = [int(i) **for** i **in** fMWill]  
 sumFMHappi = sum(fMissHappi)  
 sumFMWill = sum(fMissWill)  
 meanFMissHappi = int(sumFMHappi/numFullMiss)  
 meanFMissWill = int(sumFMWill/numFullMiss)  
  
 print(meanHitHappi)  
 *#NameError: name ' ' is not defined \*\*\* Need to find solution*

*#Part A  
  
#Need to find the file in via ideally a relative location rather than absolute location***from** os **import** listdir  
*#this is an absolute pathway \*\*\*Need to change*path = **"C:/Users/Amanda/Documents/UCL/Computing for Psychologists/Assessment 1/resultTest/"**files = listdir(path)  
  
print(files)  
  
**for** filenames **in** files:  
 **if "csv" in** filenames:  
 *#all the files have csv so it applies to all of them* allfiles = open(path+filenames, **'r'**)  
 *#open all the files in the read format* allLines = allfiles.readlines()  
 *#pull all the lines from all the files* print(allLines)  
  
 **for** i **in** allLines:  
 **for** parts **in** allLines:  
*#1. Condition* Condition = allLines[0]  
 AddreIP = allLines[2]  
 AddreIP = AddreIP.replace(**"IP address: "**,**""**)  
 *#remove the "IP address: " to get just the numbers. Easier to compare.  
 #print(AddreIP)  
  
#2. Name, Age, Gender* NAGgroup = allLines[3]  
 *#Need to seperate the name, age, and gender* NAGgroup = NAGgroup.split(**","**)  
 Name = NAGgroup[0]  
 Age = NAGgroup[1]  
 Categories = allLines[4]  
 Gender = NAGgroup[2]  
  
*#3. Gender; Male = 1, Female = 2* Gender = Gender.upper()  
 **if** Gender == **"FEMALE\n"**:  
 Gender = **"2"  
 elif** Gender == **"MALE\n"**:  
 Gender = **"1"  
 else**:  
 print(**"error"**)  
 *#print(Condition+","+AddreIP+","+Name+","+Age+","+Gender)  
  
#4. Proportion of Hits, Near misses, Full misses over total number of trials* trials = allLines[5:]  
 Trials = str(trials)  
 Trials = Trials.upper()  
 Trials = Trials.replace(**"'"**, **""**)  
 *# Need to clean up the parts of the list* Trials = Trials.replace(**"["**, **""**)  
 Trials = Trials.replace(**" "**, **""**)  
 Trials = Trials.split(**","**)  
 *#print(Trials)* wordTrial = Trials[0]  
 *#print(wordTrial)  
  
#don't need the following parts.. delete at some part  
 #results = Trials[4]  
 #happi = Trials[7]  
 #will = Trials[8]  
 #want to group the trial#, results, happiness, and willingness together.. a list of lists or dictionary  
  
#4. Calculate proportions (of hits, near misses, and full misses)out of #of trials* countTrials = Trials.count(**"TRIAL"**)  
 *#print(countTrials)* numHits = Trials.count(**"HIT"**)  
 numNearMiss = Trials.count(**"NEARMISS"**)  
 numFullMiss = Trials.count(**"FULLMISS"**)  
 *#print(numFullMiss)* numTrials = str(countTrials)  
 *#need to convert to str to print the proportion; mixed use* numHits = str(numHits)  
 numNearMiss = str(numNearMiss)  
 numFullMiss = str(numFullMiss)  
 PHits = numHits+**":"**+numTrials  
 PNearMiss = numNearMiss+**":"**+numTrials  
 PFullMiss = numFullMiss+**":"**+numTrials  
 *#print(PFullMiss)  
  
#5. Find mean happiness, mean willingness to continue per outcome type (hit, near misses, and full misses)* hitHappiness = []  
 nMissHappiness = []  
 fMissHappiness = []  
 numHits = int(numHits)  
 numNearMiss = int(numNearMiss)  
 numFullMiss = int(numFullMiss)  
 meanHitHappiness = ()  
 meanNMissHappiness = ()  
 meanFMissHappiness = ()  
  
 hitWill = []  
 nMissWill = []  
 fMissWill = []  
 numHits = int(numHits)  
 numNearMiss = int(numNearMiss)  
 numFullMiss = int(numFullMiss)  
 meanHitWill = ()  
 meanNMissWill = ()  
 meanFMissWill = ()  
  
 **for** Trials **in** allLines[5:]:  
 Trials = Trials.upper()  
 splitTrials = Trials.split(**","**)  
 *#print(splitTrials)* **if "HIT" in** Trials:  
 HHappiness = splitTrials[7]  
 HWill = splitTrials[8]  
 HWill = HWill.replace(**"\n"**, **""**)  
 hitHappiness.append(HHappiness)  
 hitWill.append(HWill)  
 hitHappiness = [int(i) **for** i **in** HHappiness]  
 hitWill = [int(i) **for** i **in** HWill]  
 sumHHappiness = sum(hitHappiness)  
 sumHWill = sum(hitWill)  
 meanHitHappiness = sumHHappiness / numHits  
 meanHitWill = sumHWill / numHits  
 *#print(HHappiness)  
 #print(meanHitWill)* **elif "NEARMISS" in** Trials:  
 nMHappiness = splitTrials[7]  
 nMWill = splitTrials[8]  
 nMWill = nMWill.replace(**"\n"**, **""**)  
 nMissHappiness.append(nMHappiness)  
 nMissWill.append(nMWill)  
 nMissHappiness = [int(i) **for** i **in** nMHappiness]  
 nMissWill = [int(i) **for** i **in** nMWill]  
 sumNMHappiness = sum(nMissHappiness)  
 sumNMWill = sum(nMissWill)  
 meanNMissHappiness = sumNMHappiness / numNearMiss  
 meanNMissWill = sumNMWill / numNearMiss  
 *#print(meanNMissHappiness)  
 #print(meanNMissWill)* **elif "FULLMISS" in** Trials:  
 fMHappiness = splitTrials[7]  
 fMWill = splitTrials[8]  
 fMWill = fMWill.replace(**"\n"**, **""**)  
 fMissHappiness.append(fMHappiness)  
 fMissWill.append(fMWill)  
 fMissHappiness = [int(i) **for** i **in** fMHappiness]  
 fMissWill = [int(i) **for** i **in** fMWill]  
 sumFMHappiness = sum(fMissHappiness)  
 sumFMWill = sum(fMissWill)  
 meanFMissHappiness = sumFMHappiness / numFullMiss  
 meanFMissWill = sumFMWill / numFullMiss  
 *#print(meanFMissHappiness)  
 #print(meanFMissWill)  
  
#6. The Max and Min Happiness levels and the trial it is in* happiList = []  
 **for** Trials **in** allLines[5:]:  
 Trials = Trials.upper()  
 splitTrials = Trials.split(**","**)  
 **if** splitTrials[0] == **"TRIAL"**:  
 happi = splitTrials[7]  
 happiList.append(happi)  
 happiList = [int(i) **for** i **in** happiList]  
 print(happiList)  
  
 happiMax = max(happiList)  
 happiMin = min(happiList)  
  
 trialMax = happiList.index(happiMax)+1  
 trialMin = happiList.index(happiMin)+1  
 trialMax = str(trialMax)  
 trialMin = str(trialMin)  
 happiMax = str(happiMax)  
 happiMin = str(happiMin)  
  
 print(happiMax +**","**+ trialMax)  
 print(happiMin + **","** +trialMin)  
  
*#7. Need to put a conditional format to prevent repeats based on the IP address.  
# Would need to stop and skip over repeats at this point in the program  
 #participants = {  
 #AddreIP: {"Condition":aList[1], "Name":aList[2] , "Age":aList[3] , "Gender":aList[5], "Categories":aList[4], "Test": 15}  
 #}  
 #print(str(participants[AddreIP]['Test']))  
 # #this should be printing from the participants dictionary.. use the Test key.. see pp datastructures slide 66  
 #print(str(participants))  
 # #another test to make sure it is pulling from the dictionary*

Partb #1.

**from** os **import** listdir  
data = listdir(**"results"**)  
expA = []  
expB = []  
allExp = []  
numData =len(data)  
numExpA = ()  
numExpB = ()  
percExpA = ()  
percExpB = ()  
print(numData)  
**for** i **in** data:  
 files = open(**"results/"** + i,**"r"**)  
 files = i  
 **if "exp" in** files:  
 allExp += [i]  
 **if "expA" in** files:  
 expA += [i]  
 numExpA = len(expA)  
 percExpA = numExpA/numData  
 percExpA = percExpA \* 100  
 **elif "expB" in** files:  
 expB += [i]  
 numExpB = len(expB)  
 percExpB = numExpB/numData  
 percExpB = percExpB \* 100  
 print(percExpB)