*"""  
Henry Ang  
CSC 4800 Advanced Python  
2/14/17  
Lab 6 - Stock Quote XML Email  
  
This program accesses the yahoo finance website to obtain the latest stock price information  
about one or more stock symbols then coverts the information into a collection XML records  
and sends an email with this information.  
  
data file: stocksymbols.txt  
"""***import** urllib.request, re, sys, datetime  
**from** sendemailMIMEmsg **import** SendEmailMsgWithAttachmentFilename  
  
**def** processQuotes(strSyms, sym, xmlFile, rawlineFile):  
 *"""  
 Process a stock quote yahoo finance website and prints out in XML format* **:param** *strSyms:* **:param** *sym: User input of stock symbol  
 """* strUrl=**'http://finance.yahoo.com/d/quotes.csv?f=sd1t1l1bawmc1vj2&e=.csv'** strUrl = strUrl + strSyms  
 **try**:  
 f = urllib.request.urlopen(strUrl)  
  
 **except**:  
 *# catch the expection if cant read url* print(**"URL access failed:\n"** + strUrl)  
 **return  
  
 for** line **in** f.readlines():  
 line = line.decode().strip(); *# convert byte array to string* print(line, file = rawlineFile)  
 **if** line == **"\""** + sym + **"\""** + **",N/A,N/A,N/A,N/A,N/A,N/A,N/A,N/A,N/A,N/A"**: *# if stock symbol is unknown* print(**"Unknown symbol: match failed\n"**)  
 **else**:  
 *# print the stock quote data* print(**"<stockquote>"**, file=xmlFile)  
 symbol(line, xmlFile)  
 date(line, xmlFile)  
 time(line, xmlFile)  
 lastSalePrice(line, xmlFile)  
 bidPrice(line, xmlFile)  
 askPrice(line, xmlFile)  
 weekLow(line, xmlFile)  
 weekHigh(line, xmlFile)  
 todayLow(line, xmlFile)  
 todayHigh(line, xmlFile)  
 netChangePrice(line, xmlFile)  
 shareVolumeQty(line, xmlFile)  
 totalShares(line, xmlFile)  
 print(**"</stockquote>"**, file=xmlFile)  
  
**def** symbol(line, xmlFile):  
 *"""  
 Prints the stock quote symbol in XML format* **:param** *line:  
 """* symbol = re.match(**"[\"][a-zA-z]+[\"]"**, line).group()  
 symbols = symbol.strip(**"\""**)  
 print(**"\t<qSymbol>"** + symbols + **"</qSymbol>"**, file = xmlFile)  
  
**def** date(line, xmlFile):  
 *"""  
 Prints the stock quote date in XML format* **:param** *line:  
 """* date = re.match(**"(.\*?)(\d+/\d+/\d+)"**, line)  
 **if** date **is None**:  
 **pass  
 else**:  
 other, dateFinal = date.groups()  
 print(**"\t<qDate>"** + dateFinal + **"</qDate>"**, file = xmlFile)  
  
**def** time(line, xmlFile):  
 *"""  
 Prints the stock quote time in XML format* **:param** *line:  
 """* time = re.match(**"(.\*?)(\d+:\d\d[pm|am]+)"**, line)  
 **if** time **is None**:  
 **pass  
 else**:  
 other, timeFinal = time.groups()  
 print(**"\t<qTime>"** + timeFinal + **"</qTime>"**, file = xmlFile)  
  
**def** lastSalePrice(line, xmlFile):  
 *"""  
 Prints the stock quote lastSalePrice in XML format* **:param** *line:  
 """* lastSalePrice = re.match(**"(.\*?)(\d+[.]\d+)"**, line)  
 **if** lastSalePrice **is None**:  
 **pass  
 else**:  
 other, lastSalePriceFinal = lastSalePrice.groups()  
 print(**"\t<qLastSalePrice>"** + lastSalePriceFinal + **"</qLastSalePrice>"**, file = xmlFile)  
  
**def** bidPrice(line, xmlFile):  
 *"""  
 Prints the stock quote bidPrice in XML format* **:param** *line:  
 """* bidPrice = re.match(**"(.\*?)(\d+[.]\d+[,])(\d+[.]\d+[,])"**, line)  
 **if** bidPrice **is None**:  
 **pass  
 else**:  
 other, lastSalesPrice, bidPrices = bidPrice.groups()  
 bidPricesFinal = bidPrices.strip(**","**)  
 print(**"\t<qBidPrice>"** + bidPricesFinal + **"</qBidPrice>"**, file = xmlFile)  
  
**def** askPrice(line, xmlFile):  
 *"""  
 Prints the stock quote askPrice in XML format* **:param** *line:  
 """* askPrice = re.match(**"(.\*?)(\d+[.]\d+[,])(\d+[.]\d+[,])(\d+[.]\d+[,])"**, line)  
 **if** askPrice **is None**:  
 **pass  
 else**:  
 other, lastSalesPrice, bidPrices, askPrices = askPrice.groups()  
 askingPrice = askPrices.strip(**","**)  
 print(**"\t<qAskPrice>"** + askingPrice + **"</qAskPrice>"**, file = xmlFile)  
  
**def** weekLow(line, xmlFile):  
 *"""  
 Prints the stock quote symbweekLow in XML format* **:param** *line:  
 """* weekLow = re.match(**"(.\*?)(\d+[.]\d+[ -])"**, line)  
 na = re.match(**"(.\*?)(\"\d+[.]\d+[ -]+\d+[.]\d+\")"**, line)  
 **if** na **is None**:  
 **pass  
 else**:  
 other, weekLows = weekLow.groups()  
 weekLowFinal = weekLows.strip(**" "**)  
 print(**"\t<q52WeekLow>"** + weekLowFinal + **"</q52WeekLow>"**, file = xmlFile)  
  
**def** weekHigh(line, xmlFile):  
 *"""  
 Prints the stock quote weekHigh in XML format* **:param** *line:  
 """* weekHigh = re.match(**"(.\*?)([- ]\d+[.]\d+)"**, line)  
 na = re.match(**"(.\*?)(\"\d+[.]\d+[ -]+\d+[.]\d+\")"**, line)  
 **if** na **is None**:  
 **pass  
 else**:  
 other, weekHighs = weekHigh.groups()  
 weekHighFinal = weekHighs.strip(**" "**)  
 print(**"\t<q52weekHigh>"** + weekHighFinal + **"</52WeekHigh>"**, file = xmlFile)  
  
**def** todayLow(line, xmlFile):  
 *"""  
 Prints the stock quote todayLow in XML format* **:param** *line:  
 """* todayLow = re.match(**"(.\*?)(.\*?\d+[.]\d+[ -])+"**, line)  
 na = re.match(**"(.\*?)(\"\d+[.]\d+[ -]+\d+[.]\d+\")([,]\"\d+[.]\d+[ -]+\d+[.]\d+\")"**, line)  
 **if** na **is None**:  
 **pass  
 else**:  
 other, todayLows = todayLow.groups()  
 a , b = todayLows.split(**","**)  
 todayLowFinal = b.strip(**" \""**)  
 print(**"\t<qTodaysLow>"** + todayLowFinal + **"</qTodaysLow>"**, file = xmlFile)  
  
**def** todayHigh(line, xmlFile):  
 *"""  
 Prints the stock quote todayHigh in XML format* **:param** *line:  
 """* todayHigh = re.match(**"(.\*?)(.\*?[-][ ]\d+[.]\d+)+"**, line)  
 na = re.match(**"(.\*?)(\"\d+[.]\d+[ -]+\d+[.]\d+\"[,])(\"\d+[.]\d+[ -]+\d+[.]\d+\"[,])"**, line)  
 **if** na **is None**:  
 **pass  
 else**:  
 other, todayHighs = todayHigh.groups()  
 a, b = todayHighs.split(**"-"**)  
 todayHighFinal = b.strip(**" "**)  
 print(**"\t<qTodaysHigh>"** + todayHighFinal + **"</qTodaysHigh>"**, file = xmlFile)  
  
**def** netChangePrice(line, xmlFile):  
 *"""  
 Prints the stock quote netChangePrice in XML format* **:param** *line:  
 """* netChangePrice = re.match(**"(.\*?)([+-]\d[.]\d+)"**, line)  
 **if** netChangePrice **is None**:  
 **pass  
 else**:  
 other, netChangePrices = netChangePrice.groups()  
 print(**"\t<qNetChangePrice>"** + netChangePrices + **"</qNetChangePrice>"**, file = xmlFile)  
  
**def** shareVolumeQty(line, xmlFile):  
 *"""  
 Prints the stock quote shareVolumeQty in XML format* **:param** *line:  
 """* shareVolumeQty = re.match(**"(.\*?)([+-]\d[.]\d+)([,]\d+)"**, line)  
 **if** shareVolumeQty **is None**:  
 **pass  
 else**:  
 other, netChangePrice, shareVolumeQtys = shareVolumeQty.groups()  
 shareVolumeQtysFinal = shareVolumeQtys.strip(**", "**)  
 print(**"\t<qShareVolumeQty>"** + shareVolumeQtysFinal + **"</qShareVolumeQty>"**, file = xmlFile)  
  
**def** totalShares(line, xmlFile):  
 *"""  
 Prints the stock quote totalShares in XML format* **:param** *line:  
 """* totalShares = re.match(**"(.\*?)(.\*?\d+[, ]\*)+"**, line)  
 **if** totalShares **is None**:  
 **pass  
 else**:  
 other, totalShares = totalShares.groups()  
 todayHighFinal = totalShares.strip(**", "**)  
 print(**"\t<qTotalOutstandingSharesQty>"** + todayHighFinal + **"</qTotalOutstandingSharesQty>"**, file = xmlFile)  
  
**def** main():  
 *"""  
 Main function of the program. Ask user to input file name and sends email with XML stock quote.   
 """* fileName = input(**'Enter stock symbol data filename: '**)  
 stockSymbol = []  
  
 **try**: *# try-except for IO error* **with** open(fileName, **"r"**) **as** file:  
 **for** line **in** file:  
 symbol = line.strip(**"\n"**)  
 stockSymbol.append(symbol)  
  
 **except** IOError: *# exception error-handling if file not found* print(**'The file \''**, fileName, **'\''' could not be found.\n'**, end=**""**)  
  
 minRange = 0  
 maxRange = 5  
 fiveSym = **""  
 with** open(str(datetime.datetime.today().strftime(**'%Y-%m-%d'**) + *# create XML document* **" stockquotes.xml.txt"**), **'w'**) **as** xmlFile:  
 **with** open(str(**"raw line.txt"**), **'w'**) **as** rawlineFile: *# create raw line file* **while**(len(stockSymbol) + 1 > minRange): *# check if 5 symbols is grabbed* **for** i **in** range(minRange, maxRange): *# grabs 5 symbols* **try**: *# if maxRange > stockSymbol indexes* sym = stockSymbol[i] + **" "** fiveSym += sym *# combine 5 symbols* **except** IndexError: *# catch index error* **pass  
 if** (len(sym) == 0):  
 **break** strSyms = **'&s='** + fiveSym  
 processQuotes(strSyms, sym, xmlFile, rawlineFile) *# generate stock quote record* fiveSym = **""** *# clear fiveSym to process next 5 sym* minRange += 5 *# adjust minRange* maxRange += 5 *# adjust maxRange* rawlineFile.close() *# close raw line file* xmlFile.close() *# close xml file  
  
 # send email with stock quote information* print(**"Sending message with attachment"**)  
 SendEmailMsgWithAttachmentFilename(**'angh@spu.edu'**,  
 [**'angh@spu.edu'**],  
 **'CSC 4800 XML Stock Quotes - Henry Ang'**,  
 open(**'raw line.txt'**, **"r"**).read(),  
 str(datetime.datetime.today().strftime(**'%Y-%m-%d'**) + **" stockquotes.xml.txt"**))  
 print(**"Done"**)  
  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 main()

*# sendemailMIMEmsg.py  
# Mike Tindall, example sending MIMEMultipart email message  
# MIME Mail Interchange Message Extension format***import** smtplib  
**from** email.mime.multipart **import** MIMEMultipart  
**from** email.mime.text **import** MIMEText  
*#import mhtpwd  
  
#-- End SendEmailMsgWithAttachment -------------------------------------------------------***def** SendEmailMsgWithAttachment(fromuser, tolist, subject, message, \  
 attachmentfilename=**"No Filename.txt"**, \  
 attachmentContent=**"No Content"**):  
  
 msg = MIMEMultipart()  
 msg.attach(MIMEText(message))  
  
 msg[**'From'**] = fromuser  
 msg[**'To'**] = **', '**.join(tolist)  
 msg[**'Subject'**] = subject  
  
 *#attachment = MIMEText(open(attachmentfilename,"r").read())* attachment = MIMEText(attachmentContent)  
 attachment.add\_header(**'Content-Disposition'**, **'attachment'**, filename=attachmentfilename+**".txt"**)  
 msg.attach(attachment)  
  
 smtpObj = smtplib.SMTP(**'smtp.office365.com'**, 587)  
 smtpObj.starttls()  
 smtpObj.login(**'angh@spu.edu'**, **"mypassword"**)  
 smtpObj.sendmail(fromuser, tolist, msg.as\_string())  
 smtpObj.quit()  
  
 **return  
  
def** SendEmailMsgWithAttachmentFilename(fromuser, tolist, subject, message, attachmentfilename):  
  
 SendEmailMsgWithAttachment(fromuser, tolist, subject, message, \  
 attachmentfilename, \  
 open(attachmentfilename, **"r"**).read())  
 **return***#-- End SendEmailMsgWithAttachment -------------------------------------------------------***if** \_\_name\_\_ == **"\_\_main\_\_"**: