MODULE 7: PYTHON LISTS

By www.HadoopExam.com

Note: These instructions should be used with the HadoopExam Apache Spark: Professional Trainings. Where it is executed and you can do hands on with trainer.

HORTONWORKS HDPCD (Hadoop Developer Certification available with total 74 solved problem scenarios. Click for More Detail)

Step 1: As we have seen in previous module that String is a sequence of characters. But now we need a collection for any data types. Hence, we will select the List data structure. Example of lists

```
course_fee=[3900,2900,1900] #Integer List
course_name=['Hadoop', 'Python','Spark'] #String List
name_and_fee=['Hadoop',2900,'Spark',2900]#Mixed List
no_classes=[] #Empty List

print (course_fee, course_name, name_and_fee,no_classes)
```

Step 2: As opposed to String list are mutable.

```
name='HadoopExam.com' #Create a String
name[7]='4' #Try to change 8th character values

course_fee[2]=2900 #Update 3rd value in list
course_fee
```

Step 3: Reading the values from list

- You cannot read values out of index
- If you use –ve index then it will count in reverse order.

```
name[len(name)-1] #String last character
name[-1] #String last character
name[-2]
```

Step 4: Traversing list

For Loop

```
for course in course_name:
        print(course)
```

Update the list: Change each course name in upper case.

```
for i in range(len(course_name)):
      course_name[i]=str(course_name[i]).upper()
```

A for loop over an empty list never runs the body:

```
for x in []:
        print('This code segment will never be executed : There are no classes available')
```

Step 5: Nested list example

```
sites = ['HadoopExam.com', ['Hadoop', 'Spark', 'Python'], 'QuickTechie.com', [1, 2, 3]]
len(sites)
```

Step 6: List concatenation

```
site1 = ['HadoopExam.com','Hadoop', 'Spark', 'Python' ]
site2=  ['QuickTechie.com' , 'Java' , 'SQL' , 'Unix' , 'Cloud Computing']

site1 + site2
```

Step 7: Repeat each elements in list

```
site1 * 3
```

Step 8: Slicing of list

```
course = ['Hadoop', 'Spark', 'Python', 'Java' , 'SQL' , 'Unix']
course[1:3]
course[:4]
course[3:]

course[1:3] = ['Cloud Computing', 'SAS'] #Updating list slice

course
```

Step 9: Sample methods on list

```
#append : Appending two lists
course = ['Hadoop', 'Spark', 'Python', 'Java' , 'SQL' , 'Unix']
course.append('Cloud Computing')
course
```

```
#extend : extend takes a list as an argument and appends all of the elements:
site1 = ['HadoopExam.com','Hadoop', 'Spark', 'Python' ]
site2=  ['QuickTechie.com' , 'Java' , 'SQL' , 'Unix' , 'Cloud Computing']

site1.extend(site2)
site1
```

```
#Sorting the list elements, Most list methods return void, Hence they modify the list and return None.
course.sort()
course
course=course.sort() #Check the content of course now.
course #We have lost the contents of our list
```

```
#Sum up all the values
course_fee=[3900,2900,1900] #Integer List
```

```
sum(course_fee)
```

Step 10: Writing word count application.

```
site1 = ['HadoopExam.com','Hadoop', 'Spark', 'Python','Hadoop', 'Spark', 'Python','Hadoop', 'Spark',
'Python','Hadoop', 'Spark', 'Python','Hadoop', 'Spark', 'Python' , 'Java' , 'SQL' , 'Unix' , 'Cloud
Computing' , 'Java' , 'SQL' , 'Unix' , 'Cloud Computing']

wordcount={}
for word in site1 :
  if word not in wordcount:
    wordcount[word] = 1
  else:
    wordcount[word] += 1
for k,v in wordcount.items():
  print k, v
```

Step 11: Filter the elements from list. All the elements which starts with ‘H’ and ends with ‘P’.

```
def filterList(mylist):
        res = []
        for name in mylist:
                if name.startswith('H') and name.endswith('p'):
                        res.append(name)
        return res

filterList(site1)
```

Step 12: Removing duplicate elements from list.

```
def removeDuplicate(mylist):
        res = []
        for name in mylist:
                if res.__contains__(name):
                        print('Its duplicate element')
                else:
                        res.append(name)
        return res

removeDuplicate(site1)
```

Step 13: Deleting elements from list.

```
#Use pop function
```

```
#pop modifies the list and returns the element that was removed. If you don't provide an index, it deletes and returns the last element.
x = site1.pop(1)
x
```

```
#Remove elements (Only first occurrence)
val=site1.remove('Hadoop')
site1
val
```

```
#Using del operator
del site1[0]
site1
del site1[1:5]
site1
```

Step 14: Working between list and strings

```
#Convert sentence the character list
sen = 'We are learning Python from HadoopExam.com Learning Resources '
char_list = list(sen)

print(char_list)
```

```
#Convert sentence to list of words
word_list = sen.split()
print(word_list)
```

```
#Joining the list
site1 = ['HadoopExam.com','Hadoop', 'Spark', 'Python' ]
delimiter = '~'
my_data = delimiter.join(site1)
print(my_data)
```

Step 15: Object and values of String and List

```
site1 = 'HadoopExam'
site2 = 'HadoopExam'

#Python  created only one string object
site1 is site2
site1 == site2


#Now in case of list
site1 = ['HadoopExam.com','Hadoop', 'Spark', 'Python' ]
site2 = ['HadoopExam.com','Hadoop', 'Spark', 'Python' ]

#Python  created two different list object
site1 is site2
```

```
site1 == site2
```

Step 16: List passed as a reference

```
def removeWords(mylist,word):
        for name in mylist:
                if mylist.__contains__(word):
                        mylist.remove(word)


removeWords(site1,'Hadoop')
```

Step 17: Distinguish between operations that modify lists and operations that create new lists. For example, the append method modifies a list, but the + operator creates a new list.

```
site1 = ['HadoopExam.com','Hadoop', 'Spark', 'Python' ]

site3 = site1 + ['Cloud Computing']
x = site1.append('SAS')

print (x, site1,site3)
```

Remember:

1. Most list methods modify the argument and return **None**. This is the opposite of the string methods, which return a new string and leave the original alone.
2. There are too many ways to do things. For example, to remove an element from a list, you can use pop, remove, del, or even a slice assignment.
3. Make copies to avoid aliasing: If you want to use a method like sort that modifies the argument, but you need to keep the original list as well, you can make a copy.
4. To add an element, you can use the append method or the + operator.

Select the Package or Products from the Below Combination and Get Great Discount

Customize Your Package : Following Products are available. You can customize your own package. Please select Products from below list and send an email to hadoopexam@gmail.com . So, our team will reply with discounted price.

Recorded Trainings (With Hands On Lab) 1. Hadoop BigData Professional Training (3500INR/$79) 2. HBase (NoSQL) Professional Training (3500INR/$79) 3. Apache Spark Professional Training (3900INR/$89 for a week 3500INR/$75) 4. Apache OOzie (Hadoop Workflow Engine) Training	5. Apache Spark Oreilly Developer Certification Hortonworks Certification : 6. HDPCD : No Java	EMC :Data Science Certification Practice Questions 7. EMC E20-007 8. EMC E20-065
Cloudera Certification Practice Question Bank 9. CCA50X : Hadoop Administrator 10. CCA-175 Cloudera® (Hadoop and Spark Developer) 11. CCP:DE575 : Cloudera® Data Engineer Certification 12. CCA159 : Cloudera® Data Analyst Certification	AWS : Amazon WebService Certification Practice Question 13. AWS Solution Architect : Associate 14. AWS Solution Architect: Professional 15. AWS Developer : Associate 16. AWS Sysops Admin : Associate	SAS Certffication Practice Questions 17. SAS Base A00-211 18. SAS Advanced A00-212 19. SAS Analytics : A00-240 20. SAS Administrator : A00-250
MapR Hadoop/BigData Certification Practice Questions 21. MapR Hadoop Developer Certification 22. MapR HBase NoSQL Certifcation 23. MapR Spark Developer Certification	Microsoft Azure Cloud Certification 24. Azure 70-532 25. Azure 70-533	Oracle Certfication Practice Questions 26. Java 1z0-808 27. 1z0-060 (Oracle 12c) 28. 1z0-061 (Oracle 12c)

DON'T FORGET TO SUBSCRIBE FOR UPDATE ON PRODUCTS

Subscribe

Request an Exam or Training