Validation OTP for CodeTantra | CodeTantra Edu | Convert JPG to PDF. Images JP

csmssengg.codetantra.com/secure/topic-details1.jsp?cid=65969c70422c5e79de0c160e&tid=65969c71422c5e79de0c1672&bd=AMTc3M19jdF9jaA==

Q1

100%

L6/Lists/Understanding List Creation

List is a data type of **Python** used to store multiple values of different types of data at a time. List are represented with `[]`.

A list can be created by putting comma separated values between square brackets `[]`.

The following program shows creation of two lists namely **list1** and **list2** :

```
list1 = [1, 2, "one", "hi"]
list2 = [4, 5, "hello"]
```

Values stored in the list are accessed using an **index**.

Index range between `0 to n-1`, where `n` is the number of values in the list

Python allows negative indexing for lists. The index of **-1** refers to the **last value of the list**, **-2** refers to the **second last value of the list** and so on.

Printing lists: The lists can be printed using the built-in function **print()** as

List1.py

Submit

```
1  value1 = input("Enter the first value: ")
2  value2 = input("Enter the second value: ")
3  value3 = input("Enter the third value: ")
4  list1 =[value1, value2, value3]
5  print("List1:",list1)
6  # create llist one with above three variables
7
8  #print the list1
9
10 value4 = input("Enter the first value: ")
11 value5 = input("Enter the second value: ")
12 value6 = input("Enter the third value: ")
13 list2 = [value4, value5, value6]
14 print("List2:", list2)
15 # create llist one with above three variables
16
17
18 #print the list2
19
20 #Concatenate list1 and list2 and print the result
21
22 l3=list1+list2
23 print("List after Concatenation:", l3)
```

Close | Reset | Submit

Type here to search

39°C Sunny

ENG US 4:54 PM 07/05/2024

L6 / Lists / Accessing items in a List

We can access the elements of a list by using **index** similar to accessing the individual characters of a string.

Unlike strings, lists are **mutable** i.e. we can modify the list (change the items, add items and reassign an item).

The following operations can be performed on the list :

1. Modify or reassign an item present at a particular index.
2. Add an item to the list at a particular index.
3. Remove an item from the list.

Consider a list `chars` equal to `['a', 'b', 'c', 'd' ]`, the code to change the item at 3rd index from `'d'` to `'xyz'` is as follows :

```
chars = ['a', 'b', 'c', 'd']
chars[3] = 'xyz'
print(chars) # will print output as follows

Output :
['a', 'b', 'c', 'xyz']
```

Trying to access an index that does not exist in a list results in an error called "IndexError".

List2.py

Submit

```
value1 = input("Enter the first value: ")
value2 = input("Enter the second value: ")
value3 = input("Enter the third value: ")

list1 = [value1, value2, value3]
for i in list1:
    print(i)
# create list1

# print every element in list1 using index

list1[2] = "Python"

# print list1
print(list1)
# add "Code is Life" to list1
list1.append("Code is Life")
# print list1
print(list1)

value4 = input("Enter the first value: ")
value5 = input("Enter the second value: ")
value6 = input("Enter the third value: ")

list2 = [value4, value5, value6]
# create list2

# Extend list1 with the elements from list2
list1.extend(list2)
# print list1
print(list1)
```

Close Reset Submit

L6/Sets/Introduction to Set

A **set** is a **mutable** data type that contains an **unordered** collection of items. A **set** is represented with **{ }**.

Every element in the set should be **unique** (no duplicates) and must be **immutable** (which cannot be changed). But the set itself is **mutable**. We can **add** or **remove** items / elements from it.

> **Note**: Mutable data types like list, set and dictionary **cannot** become elements of a set.

The **set** itself is mutable i.e. we can add or remove elements from the set.

The main uses of sets are:

- Membership testing
- Removing duplicates from a sequence
- Performing mathematical operations such as intersection, union, difference, and symmetric difference

Creation of sets

1. One way to create a set is by using built-in function **set()**

Empty set

An empty set can be created using built-in **set()** function.

- [ ] A set is an ordered collection of unique items.
- [ ] myset = { }, creates an empty set.
- [x] A set is represented using curly braces { } .
- [ ] Set allows duplicate elements.
- [ ] Set is a immutable data type.
- [ ] The elements of a set are mutable.

Close Reset Submit

L6/Tuples/Introduction to Tuple

A tuple in **Python** is similar to a list, but with important distinctions:

- Lists are enclosed in square brackets `[]`, and their elements and size can change (mutable). Tuples are enclosed in parentheses `()` and are unchangeable (immutable).
- Tuples are similar as read-only lists, providing a performance advantage during iteration due to their immutability.
- Once defined, you can't add or remove elements from a tuple.
- You can convert a tuple to a list to modify its elements and revert it to a tuple later.

Creating a tuple:

- Use the built-in `tuple()` function.
- An empty tuple can be created with `tuple1 = tuple()`.

```
tuple1 = tuple()
print(tuple1) # Result: ()
```

- Tuples can be defined by placing items within parentheses, separated by commas.
- The parentheses are optional but it is a good practice to write them.
- A one-element tuple must require a trailing comma, like `(1,)`.

```
mytuple = (1, 2, 3, "Data types")
print(mytuple)        # Result: (1, 2, 3, "Data types")
print(type(mytuple))  # Result: <class 'tuple'>
```

- [ ] Tuples are used to store similar type of data.
- [ ] tuple1 = (1.0) is correct way to create a tuple with single element.
- [x] tuples are immutable.
- [ ] Lists are immutable.
- [x] Converting a tuple into a list and list into tuple is possible.
- [ ] Lists are faster than tuples.

Close Reset Submit

Validation OTP for CodeTantra | CodeTantra Edu | Convert JPG to PDF. Images JP

csmssengg.codetantra.com/secure/topic-details1.jsp?cid=65969c70422c5e79de0c160e&tid=65969c71422c5e79de0c1672&bd=AMTc3M19jdF9jaA==

Q2

100%

L6/Tuples/Accessing elements in tuples

Individual elements of the tuple can be accessed using `index` similar to lists.

```
tuple1 = ("hi", "hello", 55, 66)
print(tuple1[0]) # Printing the element at the  0th index of the
tuple, tuple1

Output:
hi
```

Negative index values can also be used to access the individual elements. `-1` represents the last element, `-2` represents the second last element and so on.

```
tuple1 = ("hi", "hello", 55, 66)
print(tuple1[-2]) # Printing the second last element of the tuple.

Output:
55
```

Trying to access an element using an index outside the range results in an

Tuple1.py

Submit

```
1  value1 = input("first value: ")
2  value2 = input("second value: ")
3  value3 = input("third value: ")
4  value4 = input("fourth value: ")
5  value5 = input("fifth value: ")
6  mytuple =  (value1, value2, value3, value4, value5)
7  # create tuple with the above 5 inputs
8  print(mytuple)
9  # Print value at index 0
10 print(mytuple[0])
11 # Print value at index 1
12 print(mytuple[1])
13 # Print value at index -1
14 print(mytuple[-1])
15 # Print all the values from index 0
16 print(mytuple)
17 # Print all the values except the last value\
18
19 print(mytuple[0:4:1])
20 print(mytuple[::-1])
```

Close | Reset | Submit

Type here to search

39°C Sunny

ENG US 4:55 PM 07/05/2024

Validation OTP for CodeTantra | CodeTantra Edu | Convert JPG to PDF. Images JP

csmssengg.codetantra.com/secure/topic-details1.jsp?cid=65969c70422c5e79de0c160e&tid=65969c71422c5e79de0c1672&bd=AMTc3M19jdF9jaA==

Q3

100%

L6/Tuples/Convertin...

It is possible to convert tuples into lists and vice versa using built-in functions like `tuple()` and `list()`

Converting a tuple into a list

A tuple can be converted into a list using the built-in `list()` function.

```
tuple1 = ('hi', 'hello', 55, 66) # Creating a tuple, tuple1
print(tuple1) # Printing tuple1 , output contains values enclosed in parentheses indicating a tuple
print(type(tuple1)) # Printing the type of tuple1

list1 = list(tuple1) # Converting the tuple1 into a
```

Tuple2.py

Submit

```
1  mytuple = ("i", "love", "python")
2
3  #Write the missing code here
4  print("Given Tuple:",mytuple)
5
6  l1=list(mytuple)
7
8  print("After Converting Tuple into List:", l1)
9
10 l1[1]="practice"
11
12 print("List after changing element:", l1)
13
14 t1=tuple(l1)
15
16 print("After Converting List into Tuple:", t1)
17
18
```

Close | Reset | Submit

Type here to search

39°C Sunny

ENG US 4:55 PM 07/05/2024