ASCII
American Standard Code for Information Interchange
a-z : 65-91

In [5]:
```
#print first 3 letters using slicing
D='ABCDEFG'
print(D[:3])
```

```
ABC
```

In [10]:
```
#print every 2nd letter
E='clocrkr1e1c1t'
print(E[::2])
```

```
correct
```

In [11]:
```
#print backslash
print('//')
```

```
//
```

In [12]:
```
#convert to uppercase
F='You are wrong'
print(F.upper())
```

```
YOU ARE WRONG
```

In [16]:
```
#find the starting index of 'snow' in the string
G = "Mary had a little lamb Little lamb, little lamb Mary had a little lamb \
Its fleece was white as snow And everywhere that Mary went Mary went, Mary went \
Everywhere that Mary went The lamb was sure to go"
print(G.find('snow'))
```

```
95
```

In [17]:
```
print(G.replace('Mary','Bob'))
```

```
Bob had a little lamb Little lamb, little lamb Bob had a little lamb Its fleece was white as snow And everywhere that Bob went Bob went, Bob went Everywhere that Bob went The lamb was s
ure to go
```

tuples-( ) -immutable
lists-[ ] - mutable
dictionaries-{ } - mutable

TUPLES
there are diff data types. they can be stored in one single using this.
once stored address can't be changed.

In [18]:
```
tuple1=('disco','1','1.2')
print(type(tuple1[0]))
```

```
<class 'str'>
```

In [20]:
```
#negative indexing
tuple1[-1]
```

Out[20]: '1.2'

In [21]:
```
tuple1[4]
#index doesn't exist
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-21-aafa0be6d5ad> in <module>
----> 1 tuple1[4]

IndexError: tuple index out of range
```

Concatenate Tuples

In [23]:
```
tuple2= tuple1 + ('hard rock',10)
print(tuple2)
```

```
('disco', '1', '1.2', 'hard rock', 10)
```

Slicing

In [24]:
```
tuple2[3:5]
```

Out[24]: ('hard rock', 10)

In [25]:
```
tuple2[0:3]
```

Out[25]: ('disco', '1', '1.2')

In [26]:
```
#get length of tuples
len(tuple2)
```

Out[26]: 5

Sorting

we have a predefined function 'sorted'.
return type of this function is 'list'.

In [29]:
```
rating=(0,1,2,3)
ratingsorted= sorted(rating)
print(type(ratingsorted))
```

```
<class 'list'>
```

Nested Tuple

tuple inside tuple

In [34]:
```
NestedT=(1,2,('pop','rock'),(4,5),('disco',(1,2)))
```

In [31]:
```
print(NestedT[4])
```

```
('disco', 1, 2)
```

In [33]:
```
NestedT[2][1]
```

Out[33]:
```
'rock'
```

In [35]:
```
NestedT[4][1]
```

Out[35]:
```
(1, 2)
```

In [37]:
```
#first element in 2nd nested tuple
NestedT[2][1][0]
```

Out[37]:
```
'r'
```

In [38]:
```
genres_tuple = ("pop", "rock", "soul", "hard rock", "soft rock", \
                "R&B", "progressive rock", "disco")
print(len(genres_tuple))
```

```
8
```

In [39]:
```
genres_tuple[3]
```

Out[39]:
```
'hard rock'
```

In [51]:
```
genres_tuple[:3]
```

Out[51]:
```
('pop', 'rock', 'soul')
```

In [52]:
```
genres_tuple[:4]
```

Out[52]:
```
('pop', 'rock', 'soul', 'hard rock')
```

In [53]:
```
genres_tuple[:5]
```

Out[53]:
```
('pop', 'rock', 'soul', 'hard rock', 'soft rock')
```

In [54]:
```
genres_tuple[:2]
```

Out[54]:
```
('pop', 'rock')
```

In [57]:
```
#first index of tuple 'disco'
genres_tuple.index('disco')
```

Out[57]:
```
7
```

In [59]:
```
C_tuple=(-5,1,-3)
C=sorted(C_tuple)
```

Lists

to write list we write the values inside []

In [5]:
```
#Creating a list
L=['Stuti Singh',1,2,3]
L
```

Out[5]:
```
['Stuti Singh', 1, 2, 3]
```

In [6]:
```
#-1 last index
L[-1]
```

Out[6]:
```
3
```

In [8]:
```
#positive & negative index of same list element
print(L[0],L[-4])
```

```
Stuti Singh Stuti Singh
```

In [13]:
```
K= ['michael Jackson',10.1, 1982, [1,2],('A',1)]
```

In [14]:
```
print(K[4][0])
```

```
A
```

In [15]:
```
type(K[4][0])
```

Out[15]:
```
str
```

In [16]:
```
l= ['michael Jackson',10.1, 1982,'MJ',1]
```

In [17]:
```
#2 new elements added to list
l.extend(['pop',10])
l
```

Out[17]:
```
['michael Jackson', 10.1, 1982, 'MJ', 1, 'pop', 10]
```

In [19]:
```
#1 new element added
l= ['michael Jackson',10.1, 1982,'MJ',1]
l.append(['pop',10])
l
```

Out[19]:
```
['michael Jackson', 10.1, 1982, 'MJ', 1, ['pop', 10]]
```

In [20]:
```
A=['disco',10,1.2]
print(A)
A[0]='hard rock'
print(A)
```

```
['disco', 10, 1.2]
['hard rock', 10, 1.2]
```

In [21]:
```
#element deletion
del(A[0])
print('After deleting',A)
```

```
After deleting [10, 1.2]
```

Split

converts string to list separated by space

In [22]:
```
'hard rock'.split()
```

Out[22]:
```
['hard', 'rock']
```

In [23]:
```
#split by comma
'A,B,C,D'.split(',')
```

Out[23]:
```
['A', 'B', 'C', 'D']
```

Copy and Clone

In [24]:
```
#Copy (copy by reference) the list A
A=['hard rock', 10, 1.2]
B=A
print(A)
print(B)
```

```
['hard rock', 10, 1.2]
['hard rock', 10, 1.2]
```

In [1]:
```
#multiline comment
"""aheajfk
djhskfjk
ksjdskjf"""
```

Out[1]:
```
'aheajfk\ndjhskfjk\nksjdskjf'
```

In [2]:
```
var=input("Enter number")
```

```
Enter number15
```

In [3]:
```
var
```

Out[3]:
```
'15'
```

In [4]:
```
#type of input fn is always string. typecast for manipulation
type(var)
```

Out[4]:
```
str
```

In [14]:
```
#write a prog to enter 2 nos from user, print it and swap it and then print it
a=input()
b=input()
print(a,b)
a,b = b,a
print(a,b)
```

```
12
13
12 13
13 12
```

In [16]:
```
a=int(input("Enter first number"))
b=int(input("Enter second number"))
print("Original Numbers",a,b)
c=a*b
b=c//b
a=c//b
print("Swapped Numbers",a,b)
```

```
Enter first number12
Enter second number13
Original Numbers 12 13
Swapped Numbers 13 12
```

In [17]:
```
#copy by reference
A=['hard rock', 10, 1.2]
B=A
print('B[0]',B[0])
A[0]="banana"
print('B[0]',B[0])
```

```
B[0] hard rock
B[0] banana
```

In [18]:
```
#clone by value
B =A[:]
B
```

Out[18]:
```
['banana', 10, 1.2]
```

In [23]:
```
list1=[1,2,3,4]
list2=[5,6,7,8]
print(list1[:2])
print(list1[1:4])
```

```
[1, 2]
[2, 3, 4]
```

In [27]:
```
a_list=[1, 'hello', [1,2,3], True]
a_list
```

Out[27]:
```
[1, 'hello', [1, 2, 3], True]
```

In [28]:
```
a_list[1]
```

Out[28]:
```
'hello'
```

In [30]:
```
#print elements at index 1,2,3
print(a_list[1:4])
```

```
['hello', [1, 2, 3], True]
```

In [31]:
```
#concatenate 2 lists
A = [1,'a']
B = [2,1,'d']
A+B
```

Out[31]:
```
[1, 'a', 2, 1, 'd']
```

In [32]:
```
#concatenation for tuples
X = (1,2,3)
Y = (5,6,7)
X + Y
```

Out[32]:
```
(1, 2, 3, 5, 6, 7)
```