unacademy

unacademy

Python Lambda , Array

Course on Data Structure and Algorithms Using Python

Ankush Saklecha • Lesson 5 • Dec 13, 2023

unacademy

Expected Questions
from
Python Programming

What is output?

```
def fib(n):
    if n == 0 or n == 1:
        return 1
    else:
        return fib(n - 1) + fib(n - 2)

result = fib(7)
print("fib(7) =", result)
```

unacademy

What is the value returned by the function for *x=2, n=9?*

```
def what(x, n):
    value = 1
    if n > 0:
        if n % 2 == 1:
            value = value * x
        value = value * what(x * x, n // 2)
    return value
```

unacademy

What value would the following function return for the input x = 95?

```
def fun(x):
    if x > 100:
        return x - 10
    else:
        return fun(fun(x + 11))
```

unacademy

Q Consider the following python function definition

```
def Trial(a, b, c):
    if a >= b and c < b:
        return b
    elif a >= b:
        return Trial(a, c, b)
    else:
        return Trial(b, a, c)
```

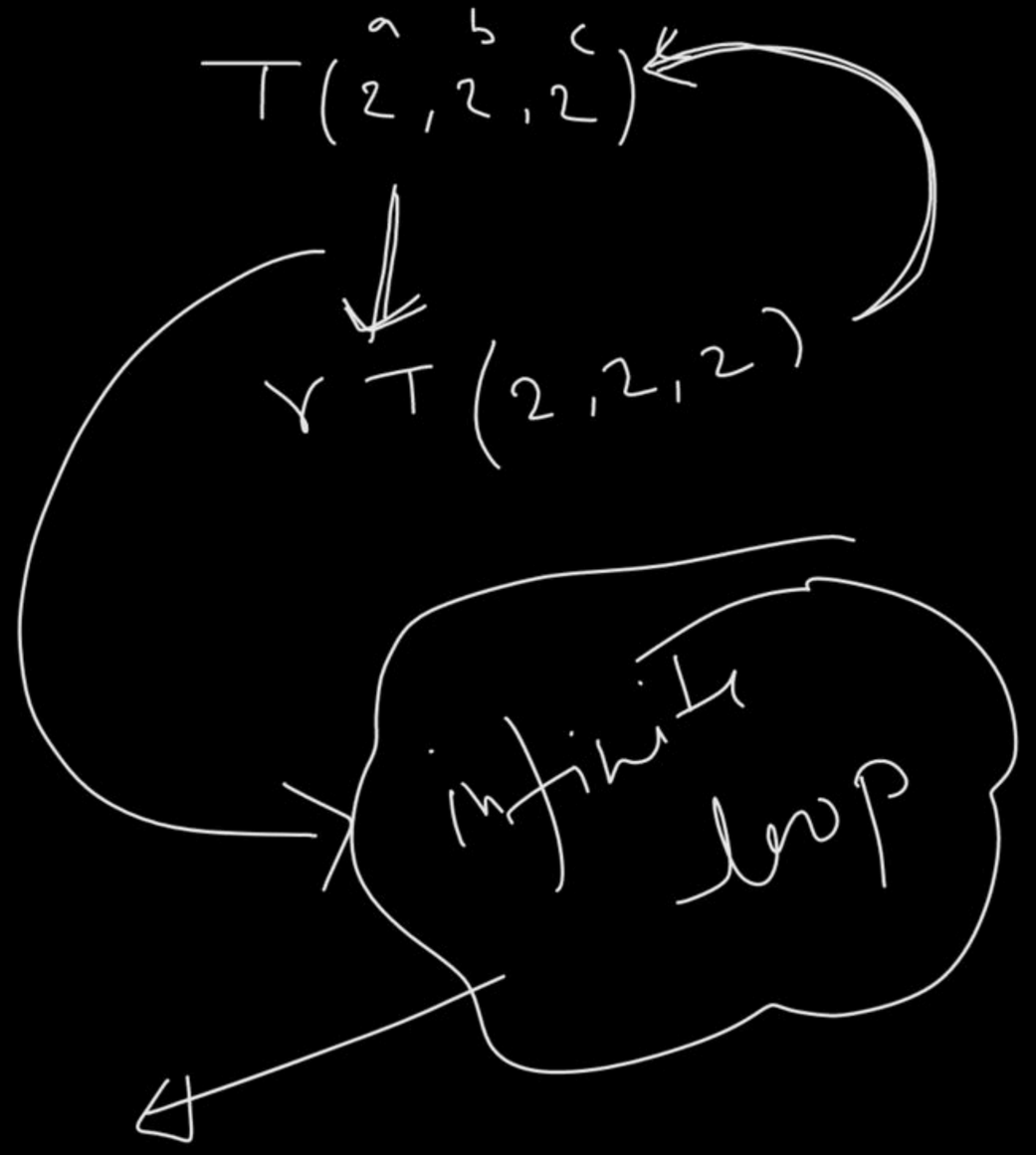
The function trial

(A) finds the maximum of a, b and c

(B) finds the minimum of a, b and c

(C) finds the middle number of a, b, c

(D) none of the above



unacademy

What value return by j at the end of program ?

```
def incr(i):
    if not hasattr(incr, 'count'):
        incr.count = 0
    incr.count += i
    return incr.count

def main():
    i, j = 0, 0
    for i in range(5):
        j = incr(i)
        print(f"i = {i}, j = {j}")

if __name__ == "__main__":
    main()
```

The value returned by **f(1) is**

```
i = 0
def f(n):
    if n >= 5:
        return n
    global i
    n = n + i
    i += 1
    return f(n)
Z = f(n)
Print(Z)
```

o/p => 7

Print(f(n))

```
def foo(n, sum):
    if n == 0:
        return
    k = n % 10
    j = n // 10
    sum += k
    foo(j, sum)
    print(f"{k},", end="")
def main():
    a = 2048
    sum = 0
    foo(a, sum)
    print(f"{sum}")
if __name__ == "__main__":
    main()
```

2, 0, 4, 8, 0

What does the above program print?

(A) 8, 4, 0, 2, 14 (B) 8, 4, 0, 2, 0

(C) 2, 0, 4, 8, 14 (D) 2, 0, 4, 8, 0

unacademy

f

Sum = 10

f"(sum)"

→ string 10

f"{ }"

f"{sum} Hi"

→ 10 Hi

unacademy

```
def f(n):
    global r
    if n <= 0:
        return 1
    if n > 3:
        r = n
        return f(n - 2) + 2
    return f(n - 1) + r
r = 0
result = f(5)
print("Result:", result)
```

r = ∅ 5	1 + 5
f(1)	r(f(0) + r)
f(2)	6 + 5 r(f(1) + r)
f(3)	11 + 5 r(f(2) + r)
f(5)	r = 5 16 + 2 r(f(3) + 2)

18

unacademy

```
def foo(n, r):
    if n > 0:
        return n % r + foo(n // r, r)
    else:
        return 0
```

Foo (513, 2)

1 + Foo (256, 2)

0 + F (128, 2)

0 + F (64, 2)

0 + F (32, 2)

0 + F (16, 2)

0 + f (8, 2)

0 + f (4, 2)

0 + f (2, 2)

0 + f (1, 2)

1 + F (0, 2)

0

2

1. What is the return value of the function foo when it is called as foo (513, 2)?
2. What is the return value of the function foo when it is called as foo (245, 8)?
3. What is the return value of the function foo when it is called as foo (245, 10)?

1. (2)

2. (14)

3. (11)

2+4+5

11

unacademy

$f(24, 8)$

$3 + F(0, 0)$ → 0

$6 + F(3, 8)$ → 3

$9 + f(30, 8)$

$5 + f(30, 8)$

14

unacademy

fun(245, 10)

5 + fun(24, 10) 6

4 + fun(2, 10) 2

2 + fun(0, 10)

0

11

unacademy

```
def func(num):
    count = 0
    while num:
        count += 1
        num >>= 1
    return count

num = 513
result = func(num)
print(result)
```

Count = 0 + 2

while 513:

C = 1

num => 256 => 128

num = 1

513 → 256 → 128 → 64 → 32 → 16 → 8 → 4 → 2 → 1 → 0

1 2 3 4 5 6 7 8 9 10

110

unacademy

What is output ?

```
m = 10
n, n1 = 0, 0
m += 1
n = m
m += 1
n1 = m
n -= 1
n -= n1
print(n)
```

m = 10 + 12

n = 0 + 10

n1 = 0 12

n - n1

10 - 12

-2

-2

unacademy

```
def jumble(x, y):
    x = 2 * x + y
    return x

def main():
    x = 2
    y = 5
    y = jumble(y, x)
    x = jumble(y, x)
    print(x)

if __name__ == "__main__":
    main()
```

$x = \cancel{2}\ 26$ $y = \cancel{5}\ 12$

$y = j(5, 2)$

$x = j(12, 2)$

$x = 26$

26

$j(\overset{x}{5}, \overset{y}{2})$

$x = 2 * 5 + 2$

return $\underset{12}{x}$;

$j(\overset{x}{12}, \overset{y}{2})$

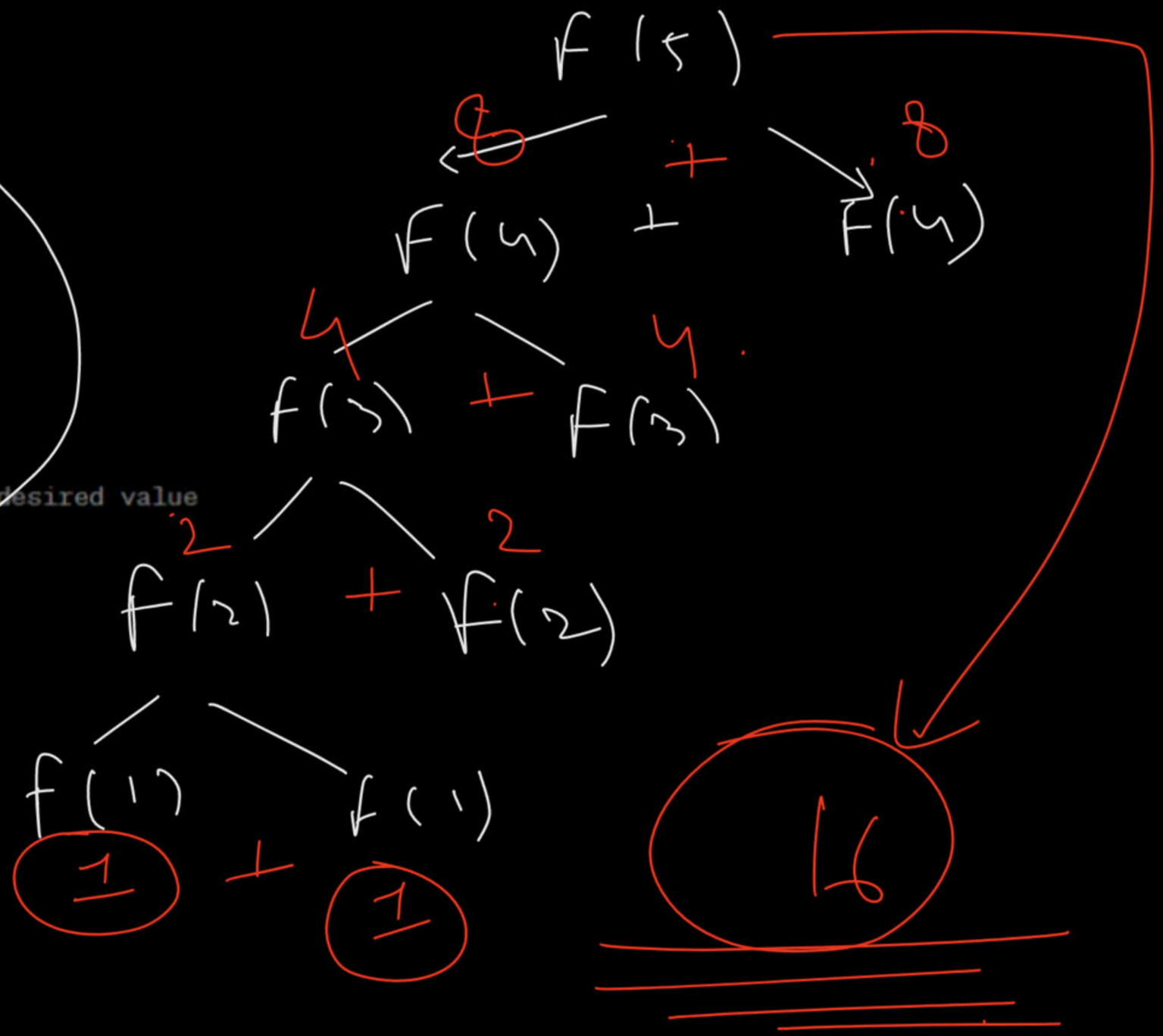
$x = 2 * 12 + 2$

$x = 26$

return $\underset{26}{x}$;

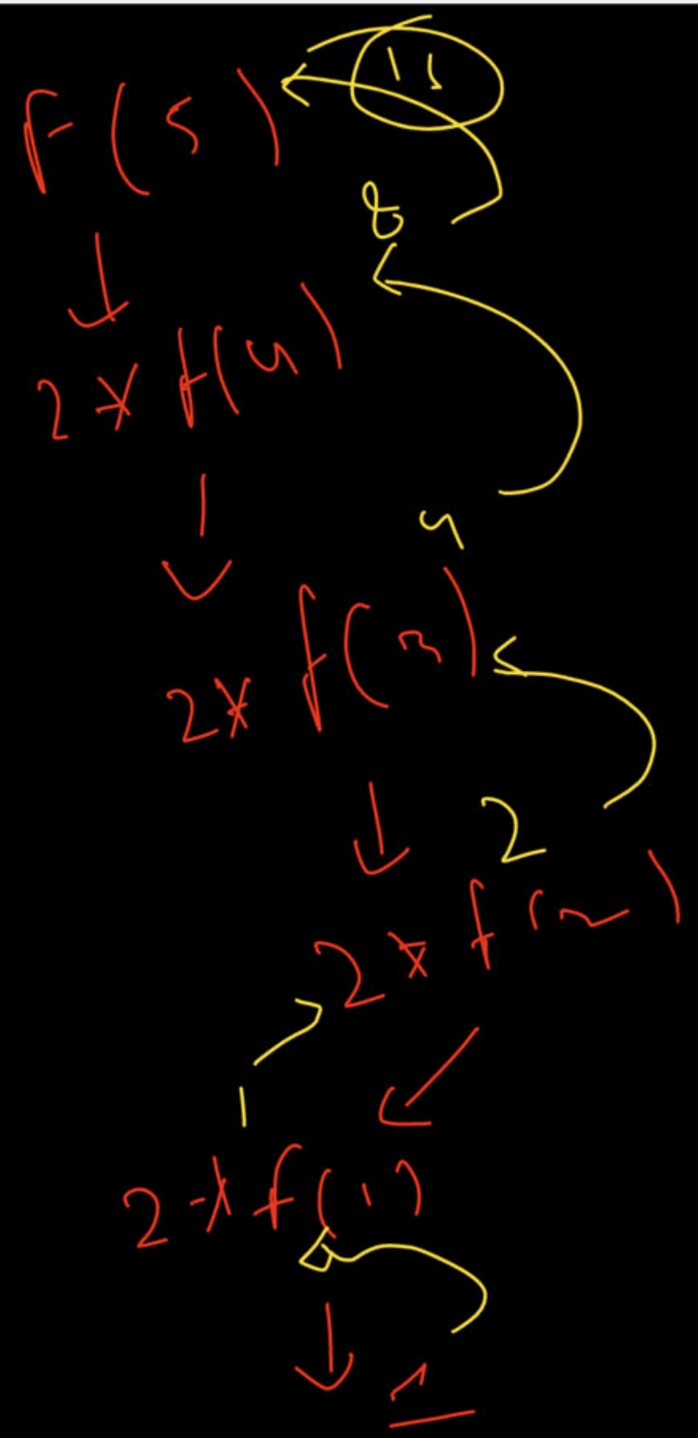
unacademy

```python
def fun(n):
    if n <= 1:
        return n
    else:
        return fun(n - 1) + fun(n - 1)

# Example usage:
result = fun(5)  # Replace 5 with the desired value
print(result)
```



unacademy

```python
def fun(n):
    if n <= 1:
        return n
    else:
        return 2 * fun(n - 1)

# Example usage:
result = fun(5)  # Replace 5 with the desired value
print(result)
```



unacademy

Python Programming

Python Collections (List and Set)

unacademy

Content

- List
- List Indexing and Slicing
- Nested List
- List Methods
- Set
- Set Methods
- Sample Problems

unacademy

List

List used to store multiple items in a single variable. It is one of 4 built-in data types in Python used to store collections of data. other three are Set, Tuple, Dictionary. It is defined as:

Syntax:

List_name=[item-1, item-2, item-3, item-4,……, item-n]

Example:

Designation=['Assistant', 'Supervisor', 'Manager', 'Technician']

unacademy

List

List items are ordered, changeable or mutable, and allow duplicate data items.

Ordered:

```
Age=[20,24,17,26,22,28,32,25,36,31]
print(Age)
```

Mutable:

```
Age=[20,24,17,26,22,28,32,25,36,31]
Age.append(45)
print(Age)
```

Duplicate:

```
Age=[20,24,17,26,22,28,32,25,36,31,32,42]
print(Age)
```

unacademy

List Indexing & Slicing

- Like a string, we can also define indexes to data items of list.
- Python support two types of indexing:

  --Positive Indexing

  --Negative Indexing

unacademy

Positive Indexing

- It begin with zero and end with list length-1 from left hand side.

List Positive Indexing Examples:

Age=[20,24,17,26,22,28,32,25,36,31,32,42]

Positive Index

0	1	2	3	4	5	6	7	8	9	10	11
20	24	17	26	22	28	32	25	36	31	32	42

Access List Data Items via Index:

Age=[20,24,17,26,22,28,32,25,36,31,32,42]

0	1	2	3	4	5	6	7	8	9	10	11
20	24	17	26	22	28	32	25	36	31	32	42

print(Age[1])

Let's Try…

Guess Output?

Age=[20,24,17,26,22,28,32,25,36,31,32,42]

```
print(Age[2*3])
print(Age[10//3])
print(Age[10%3])
print(Age[2**2])
print(Age[8>>2])
print(Age[2&4])
print(Age[6^3])
```

1 → 32
2 → 24
3 → 24
4 → 22
5 → 17
6 → 20
7 → 28

unacademy

unacademy

Negative Indexing

- It begin with -1 and end with list length in negative from right hand side

String Negative Indexing Examples:

Age=[20,24,17,26,22,28,32,25,36,31,32,42]

Negative Index

-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
20	24	17	26	22	28	32	25	36	31	32	42

Access String Character via Negative Index:

Age=[20,24,17,26,22,28,32,25,36,31,32,42]

-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
20	24	17	26	22	28	32	25	36	31	32	42

print(Age[-8])

Let's Try…

Guess Output?

Age=[20,24,17,26,22,28,32,25,36,31,32,42]

print(Age[-5]) → 25

print(Age[-6-2]) → 22

print(Age[-9+11]) → 12

print(Age[14-10]) → 22

print(Age[-8-3]) → 24

print(Age[-4+4]) → 20

print(Age[-12+6]) → 32

unacademy

List Slicing

- It is way to access set of data items from the given list. It is denotes as:

  list_name[startindex : endindex], ...]

- It slice list from startindex and end with endindex-1 i.e. it excludes endindex value. Slice index may be positive or negative range.

[24, 17, 26, 22]

String Slicing Examples:

Age=[20,24,17,26,22,28,32,25,36,31,32,42]

Positive Index

0	1	2	3	4	5	6	7	8	9	10	11
20	**24**	**17**	**26**	**22**	**28**	**32**	**25**	**36**	**31**	**32**	**42**
-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

Negative Index

print(Age[0:3])

print(Age[-11:-7])

[20, 24, 17]

unacademy

Let's Try…

Guess Output?

```
Age=[20,24,17,26,22,28,32,25,36,31,32,42]
print(Age[1+1:4+0])
print(Age[0:2**3])
print(Age[-12:-7+3])
print(Age[-1:-7])
print(Age[:10])
print(Age[:-6])
print(Age[2:-2])
print(Age[-8:14])
```

Change List Data Items

Example:

```
EmpName=['Jhon','Bob','Marry','Cherry']
EmpName[1]='Romy'
print(EmpName)
EmpName[0:2]=['Romy','Jack']
print(EmpName)
```

['Jhon', 'Romy', 'Marry', 'Cherry']

['Romy', 'Jack', 'Marry', 'Cherry']

Insert New Data Items in List

Example:

```
EmpName=['Jhon','Bob','Marry','Cherry']
EmpName.insert(0,'Jam')
print(EmpName)
```

['Jam', 'Jhon', Bob . - - - -

unacademy

Append New Data Items at End of the List

Example:

```
EmpName=['Jhon','Bob','Marry','Cherry']
EmpName.append('Jam')
print(EmpName)
```

unacademy

Remove Specified Data Items from the List

Example:

```
EmpName=['Jhon','Bob','Marry','Cherry']
EmpName.remove('Bob')
print(EmpName)
EmpName.pop(0)
print(EmpName)
del EmpName[1]
print(EmpName)
del EmpName
print(EmpName)
EmpName.clear()
print(EmpName)
```

unacademy

Sort

- sort() is a method of list class and can only be used with lists.

- ***Syntax:*** *List_name.sort(key, reverse=False)*
  Parameters:
  key: *A function that serves as a key for the sort comparison.*
  reverse: *If true, the list is sorted in descending order.*
  Return type: *None*

unacademy

Sort List Data Items

Example:

```
EmpName=['Jhon','Bob','Marry','Cherry']
EmpName.Sort()                          # Ascending Order
print(EmpName)
EmpName.Sort(reverse=True)              # Descending Order
print(EmpName)
```

```
def myFunc(e):
 return len(e)

cars = ['Tata', 'creta', 'BMW', 'Ford']

cars.sort(key=myFunc)
print(cars)
```

4 5 3 4

3 4 4 5

myfunc() Stable Sorting

```
def myFunc(e):
 return len(e)

cars = ['Tata', 'creta', 'BMW', 'Ford']

cars.sort(key=myFunc, reverse = True)
print(cars)
```

5 4 4 3

unacademy

T

unacademy

Copy Lists

Example:

```
EmpName=['Jhon','Bob','Marry','Cherry']
Name=EmpName.copy()                    # Copy method
print(Name)

Name=list(EmpName)                     # List Method
print(Name)
```

Join Lists

Example:

```
EmpName=["Jhon","Bob","Marry","Cherry"]
Age=[24,22,28,19]
NameAge=EmpName+Age                    # + Operator
print(NameAge)
for x in Age:
  EmpName.append(x)
print(EmpName)
EmpName.extend(Age)                    # Extend() Method
print(EmpName)
```

["Jhon", "Bob" . . . . . , 24, 22, 28, 19]

[ ] . . . . . , [ ] . . . . . , [ ] . . . . .]

List Comprehension

Example:

```
EmpName=['Jhon','Bob','Marry','Cherry']
newlist = [x for x in EmpName ]
print(EmpName)
```

Print (newlist)

newlist = [1 for x in EmpName]

newlist = [1, 1, 1, 1]

unacademy

Nested Lists

Nested Lists are used to store multiple sublists in a single list..\

It is defined as:

Syntax:
List_name=[[item-1, item-2], [item-3, item-4],[……, item-n]]
Example:
Student=[['Allen','Bob','Jhon','Jerry'],[21,23,19,24],[7.5,8.0,8.5,7.6]]

unacademy

Nested Lists Indexing & Slicing

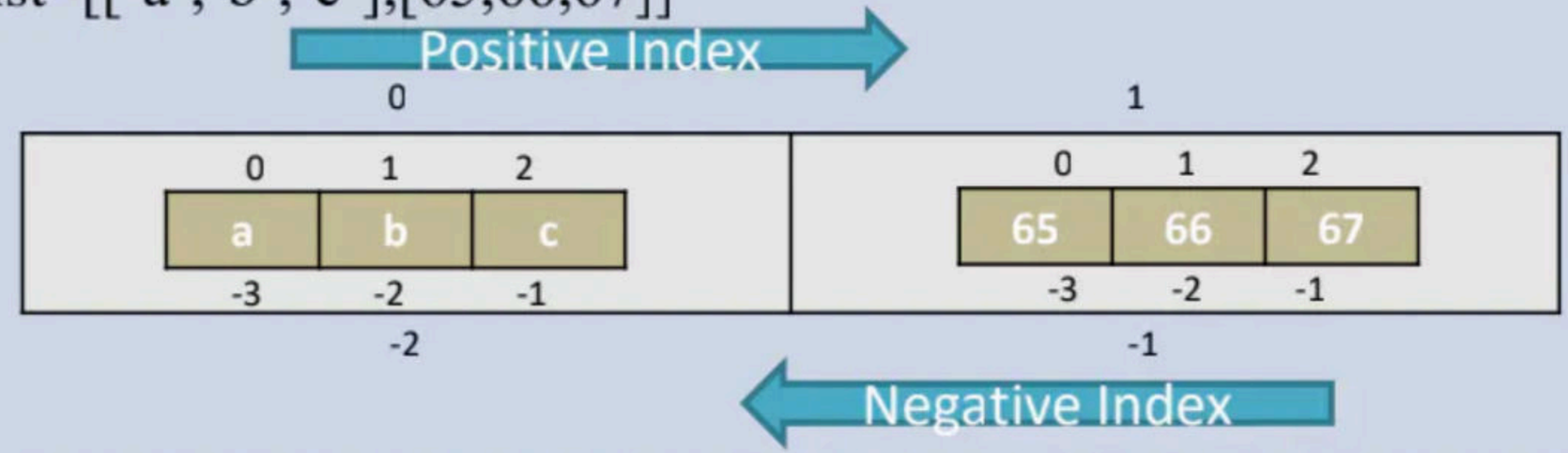
- It is way to access set of data items from the given list. It is denotes as:

  list_name[[startindex : endindex],[],[]]

- It slice list from startindex and end with endindex-1 i.e. it excludes endindex value. Slice index may be positive or negative range.

String Slicing Examples:

nlist=[['a','b','c'],[65,66,67]]



print(nlist[0][2])

unacademy

```
EmpName=[["Jhon","Bob","Marry","Cherry"],[1,2,3,4],[1.1,1.2,1.3]]
print(EmpName[0])
print(EmpName[1])
print(EmpName[0:3])
```

unacademy

Expected Questions
from
Python Programming

unacademy

List Methods

count( ):

```
Alpha=['a','b','a','c','d','d']
print(Alpha.count('e'))
```

index( ):

```
Alpha=['a','b','a','c','d','d']
print(Alpha.index('a' ))
```

sort & reverse( ):

```
Alpha=['a','b','a','c','d','d']
print(Alpha.sort( ))
print(Alpha.reverse( ))
```

unacademy

Set

Set is also used to store multiple items in a single variable. It is defined as:

Syntax:
Set_name={item-1, item-2, item-3, item-4,……, item-n}
Example:
Designation={'Assistant', 'Supervisor', 'Manager', 'Technician'}

unacademy

Set

Set items are unordered, unchangeable, and do not allow duplicate values.

Unordered:

```
Age={20,24,17,26,22,28,32,25,36,31}
print(Age)
```

Immutable or Unchangeable:

```
Age={20,24,17,26,22,28,32,25,36,31}
Age[2]=35
print(Age)
```

Duplicate Not Allowed:

```
Age=[20,24,17,26,22,28,32,25,36,31,32,42]
print(Age)
```

unacademy

Add Set Items

Example:

```
EmpName={'Jhon','Bob','Marry','Cherry'}
EmpName.add('Jam')
print(EmpName)
Age={21,24,26,28}
EmpName.update(Age)
print(EmpName)
```

unacademy

Remove Set Items

Example:

```
EmpName={'Jhon','Bob','Marry','Cherry'}
EmpName.remove('Bob')          #remove()  method that cause errors if items not found
print(EmpName)

EmpName.discard('Jam')         #discard()  method does not cause errors if items not found
print(EmpName)

print(EmpName.pop())           #pop() method remove random element from the set

print(EmpName.clear())         # clear() method remove all elements from the set

print(EmpName)                 # del keyword will delete set automatically
```

unacademy

Join Sets

Example:

```
EmpName={'Jhon','Bob','Marry','Cherry'}
print(EmpName)
Age={21,24,26,28}
print(Age)
EmpAge=EmpName.union(Age)
print(EmpAge)
```

unacademy

Keep Duplicates

Example:

```
A={12,4,5,18,25,16,30}
B={25,12,2,18}
print(Age)
A.intersection_update(B)
print(A)
C=A.intersection_update(B)
print( C )
```

unacademy

Keep All Except Duplicates

Example:

```
A={12,4,5,18,25,16,30}
B={25,12,2,18}
print(Age)
A.symmetric_difference_update(B)
print(A)
C=A.symmetric_difference(B)
print( C )
```

unacademy

Set Methods

In python, several built-in methods are available.

Method Name	Description
add()	Adds an element to the set
clear()	Removes all the elements from the set
copy()	Returns a copy of the set
difference()	Returns a set containing the difference between two or more sets
difference_update()	Removes the items in this set that are also included in another, specified set
discard()	Remove the specified item
intersection()	Returns a set, that is the intersection of two other sets
intersection_update()	Removes the items in this set that are not present in other, specified set(s)
isdisjoint()	Returns whether two sets have a intersection or not
issubset()	Returns whether another set contains this set or not
issuperset()	Returns whether this set contains another set or not

unacademy

Set Methods

In python, several built-in methods are available.

Method Name	Description
pop()	Removes an element from the set
remove()	Removes the specified element
symmetric_difference()	Returns a set with the symmetric differences of two sets
symmetric_difference_update()	inserts the symmetric differences from this set and another
union()	Return a set containing the union of sets
update()	Update the set with the union of this set and others

unacademy

Set Methods

isdisjoint( ):

```
A={12,4,5,18,25,16,30}
B={25,12,2,18}
z = A.isdisjoint(B)
print(z)
```

issubset( ):

```
A={12,4,5,18,25,16,30}
B={25,12,2,18}
z = A.isdisjoint(B)
print(z)
```

issuperset( ):

```
A={12,4,5,18,25,16,30}
B={25,12,2,18}
z = A.issuperset(B)
print(z)
```

unacademy

Thank You…

unacademy

unacademy

unacademy

unacademy

unacademy