Python

Library: import random

Use:  used to generate random numbers

Functions: seed(), getstate(), setstate(), getrandbits(), randrange(), randint(), choice(), choices(), sample(), random() etc.

Functions that operate on other functions are called "higher-order functions.Ex = seed(getstate())

key in higher-order function helps to find maximum answer for called function.Ex = print(max(10,45,15,key = sub) here max is for maximum no and sub is defined function.

Dictiontionary in python is also called heap table.

Variables :

1. Flag : Flag variable is used as a signal in programming to let the program know that a certain condition has met. It usually acts as a boolean variable indicating a condition to be either true or false.

Functions:

* Len() = len() function returns the number of items in an object.
* append() = The append() function in Python takes a single item as an input parameter and adds it to the end of the given list.
* map() = map(fun,item)
* round()=round(app,5):Limits decimal value to 5 numbers.Ex= print(round(338442,-3) = 338000 . if -2 then 33 if -1 then 3
* a.remove()= removes particular data.
* a.append()=adds particular data.
* min(a) = finds minimum number in list
* max(a)= finds maximum number in list
* sum(a) = finds sum of data in list
* abs(a) = gives absolute value ex. 32 =32,-32=32.

**In Python, there are several methods to remove duplicates from a list. Here are some common approaches:**

**1. Using a set**

The simplest and most efficient way to remove duplicates from a list is to convert it to a set, which automatically removes duplicates, and then convert it back to a list if needed

array = [1, 2, 2, 3, 4, 4, 5]

unique\_array = list(set(array))

print(unique\_array) # [1, 2, 3, 4, 5]

**2. Using a set with list to preserve order**

If you want to preserve the order of elements, you can use a set along with a list comprehension:

array = [1, 2, 2, 3, 4, 4, 5]

seen = set()

unique\_array = [x for x in array if not (x in seen or seen.add(x))]

print(unique\_array) # [1, 2, 3, 4, 5]

**3.Using dict.fromkeys**

This method also preserves the order of elements and works with Python versions 3.7 and later (where dictionaries preserve insertion order).

array = [1, 2, 2, 3, 4, 4, 5]

unique\_array = list(dict.fromkeys(array))

print(unique\_array) # [1, 2, 3, 4, 5]

**4. Using a Custom Function**

If you prefer a more explicit approach or need additional processing, you can write a custom function:

def remove\_duplicates(lst):

seen = set()

result = []

for item in lst:

if item not in seen:

result.append(item)

seen.add(item)

return result

array = [1, 2, 2, 3, 4, 4, 5]

unique\_array = remove\_duplicates(array)

print(unique\_array) # [1, 2, 3, 4, 5]

**Adding Element in array**

**a.insert(1,4)** :#here 1 is position and 4 is inserted

**a.append(4):** #here 4 is inserted at last

**Removing an element in array**

**a.pop(1):** #here 1 is position

**a.remove(1):** #here 1 is the no removed

**Slicing an array**

slicied\_array = **a[1:3**] :#it gives an array from position 1 and 2

slicied\_array = **a[2:]** : #it gives nos from 2nd element to end

slicied\_array = **a[:]** : # prints all elements

**searching an element**

**a.index(2)** :# gives first occurrence of value 2

print(**a.index(2))**

**updating values**

**a[2] = 4** : #it updates the value at position 2 to 4

**counting no of occurrence**

n = **a.count(3)** :#Calculates no of occurrence of no 3

**reversing array**

**a.reverse():** # reverses the array

print(a)

**extending an array**

**a.extend([6,7,8,9]):** # Extends the array

**Slicing In array**

**nums[:i + 1]**: This is a slice of the list nums, which includes elements from the start of the list up to, but not including, the index i + 1

Here array sliced to give specifix numbers at indexes for eg: nums[1:3] prints no from index 1 to 3