COMPUTER SCIENCE AND DA

Data Structures through Python

Lists and Arrays

Lecture No. 01

By- Kashif Sir

Topics to be covered

1) Data Structure ?

2) Classification of Data Structures

3) Array implementation using List

Data Structure, Types

Data types

Programming
(Set of instructions)

{ Data Structure }
⇓
how the data is organised

Algorithm
↳ step by step solution

Data
⇓
collection of raw facts $\xrightarrow{\text{process}}$ Information

Data Types

str

a = "GATE"

$a = \underline{a/10}$

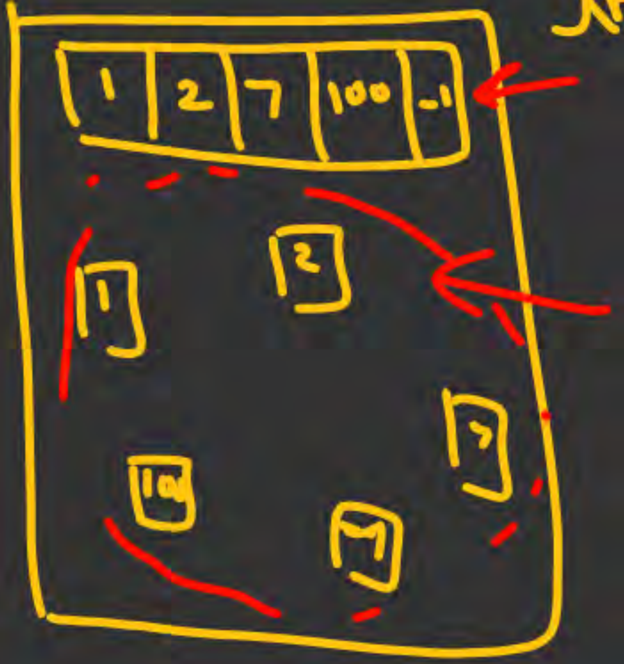
$a = \underbrace{\text{input}(\ )}$ ⇒ 60

$a = a/2$ → Error

a = input( )
b = input( )
$c = \underbrace{a+b}$
✓ concatenate

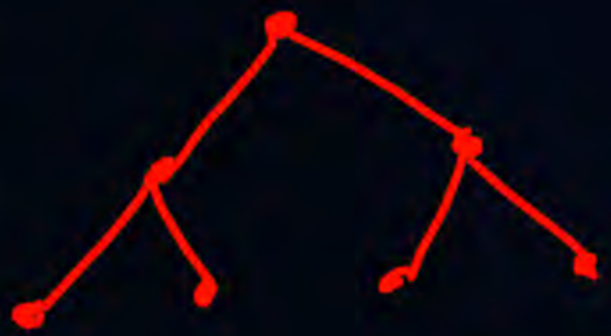
Data Structure

Represents how the data will be organized in the memory



1, 2, 7, 100, -1

Data Structure, Types



[1, 2, 9, 10] ←

Data Types:

Basic / Primitive / Fundamental : int, float, str, bool, ..

Derived / Collections : sets, tuple, dictionary, list

Data Structures

Linear Data Structures: Array, Stack, Queue, Linked list, Hashmap
(All the data will be at same level)

Non-Linear Data Structures : Tree, Graphs → Algorithms
(multiple levels)

Lists and Arrays in Python

Array

Collection of similar type of data

Theoretically.

There are three way to implement array in Python : [, 2

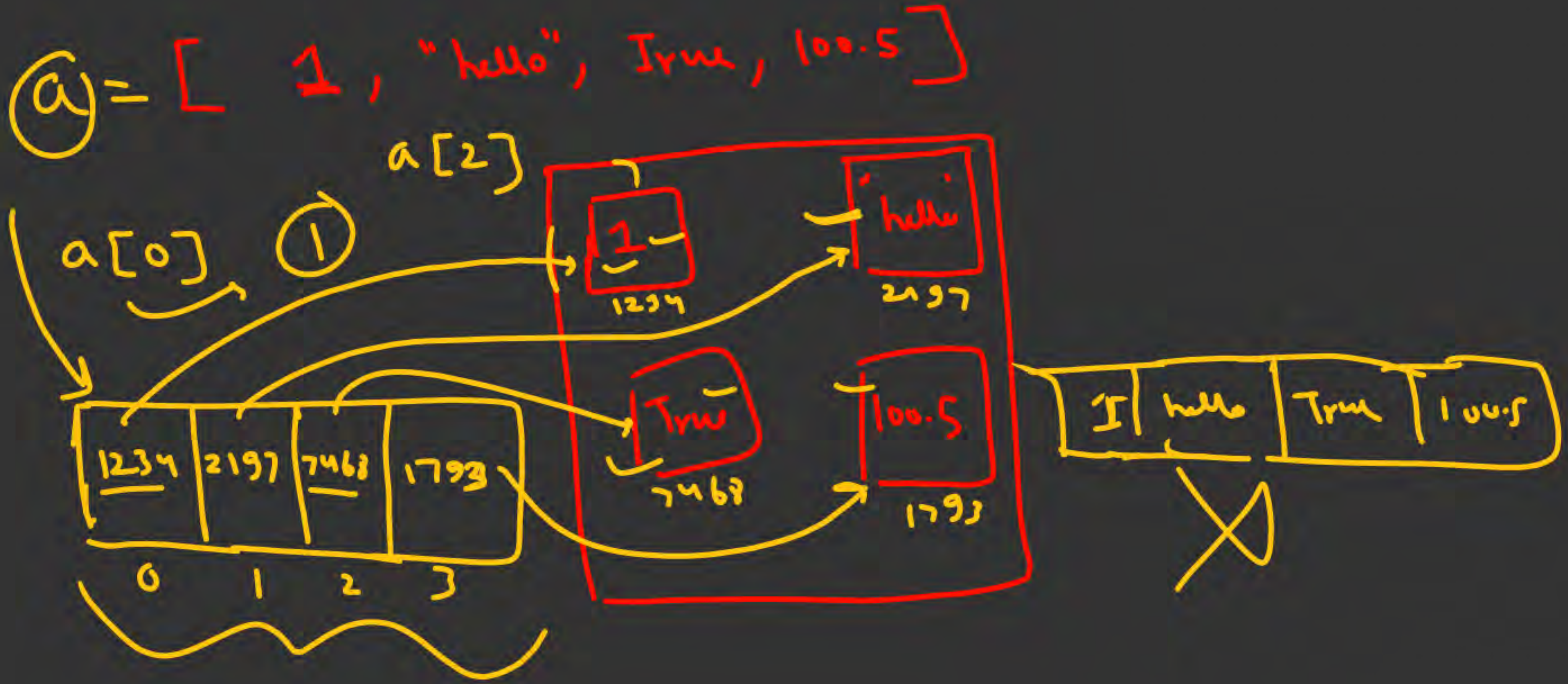
1) array( ) method}

GATE ⟹ 2) list →

3) numpy module }

contiguous

non-contiguous



```
a = [ 1, "hello", True, 100.5 ]
```

Array implementation using List

1) Ordered Collection : List elements can be accessed through index

2) Mutable => modify

3) Multiple datatypes

4) List creation : [ ] , list( )

a=[10,20,30] , a= list(10,20,30)

Core Array Operations Using Python Lists

1) Creation

```
arr = [10, 20, 30, 40]
```

2) Accessing Elements

```
print(arr[0])    # Output: 10
print(arr[-1])   # Output: 40 (last element)
```

3) Modifying Elements ✓

```
arr[1] = 99
print(arr)  # Output: [10, 99, 30, 40]
```

4) Appending (Add at End)

```
arr.append(50)
print(arr)  # Output: [10, 99, 30, 40, 50]
```

5) Inserting (Add at Specific Position)

```
arr.insert(2, 25)
print(arr)  # Output: [10, 99, 25, 30, 40, 50]
```

$$\begin{array}{lcccccc} & 0 & 1 & 2 & 3 & & \\ arr = [& 10, & \cancel{20}\ 99, & 30, & 40, & 50 &] \\ & -4 & -3 & -2 & -1 & & \end{array}$$

arr[1]=99

arr = [10, 99, 25, 30, 40, 50]

Core Array Operations Using Python Lists

6) Extending (Merging with Another List)

```
arr.extend([60, 70])
print(arr)  # Output: [10, 99, 25, 30, 40, 50, 60, 70]
```

7) Removing by Value ✓

```
arr.remove(99)
print(arr)  # Removes first occurrence of 99
```

8) Removing by Index (Pop)

```
arr.pop(2)
print(arr)  # Removes item at index 2
```

[10, 25, 30, 40, 50, 60, 70]

[10, $\boxed{25}$, 40, 50, 60, 70,]

$\boxed{30}$

9) Searching

```
print(arr.index(40))  # Returns index of value 40
print(60 in arr)      # Returns True if 60 exists
```

$\boxed{2}$

True

10) Length

```
print(len(arr))  # Number of elements in the list
```

$\boxed{6}$

Core Array Operations Using Python Lists

11) Slicing

```
print(arr[1:4])  # Output: Elements from index 1 to 3
```

12) Reversing

```
arr.reverse()
print(arr)
```

13) Sorting

```
arr.sort()  # Ascending
print(arr)

arr.sort(reverse=True)  # Descending
```

arr[1 : 4]

4-1

1, 2, 3

a = [1, 2, 3, 4]

b = [100, 200]

[1, 2, 3, 4, 100, 200]

Summary

1) Data Structure

2) Array

3) List

4) List functions

THANK - YOU