##--
Introduction--

- Python is a high-level, interpreted, general-purpose programming language.
- It is designed to be easy to read and write, and it has a wide range of applications, including web development, data science, and machine learning.
- Wide range of libraries that helps in the field of Artificial Intelligence

##---Variables and Data Type---

- Variables are used to store data in Python.
- Data types define the type of data that can be stored in a variable.
- The most common data types are:
  - `int`: integers (e.g., 1, 2, 3)
  - `float`: floating-point numbers (e.g., 1.23, 4.56)
  - `str`: strings (e.g., "hello", "world")
  - `bool`: booleans (e.g., True, False)

```
course_name = " Big Data and Data Science"
print(course_name)

 Big Data and Data Science
```

-- Operators--

- Operators are used to perform operations on data.
- The most common operators are:
  - `+`: addition
  - `-`: subtraction
  - `*`: multiplication

  - `/`: division
  - `%`: modulus
  - `==`: equality
  - `!=`: inequality
  - `<`: less than
  - `<=`: less than or equal to
  - `>`: greater than
  - `>=`: greater than or equal to

##---Control Flow--

- Control flow statements are used to control the execution of code.
- The most common control flow statements are:
  - `if`: executes a block of code if a condition is true
  - `elif`: executes a block of code if a condition is true and all previous `if` conditions are false
  - `else`: executes a block of code if all previous `if` and `elif` conditions are false
  - `for`: iterates over a sequence of values
  - `while`: executes a block of code while a condition is true

```
# If-else statement
y = -5
if y > 0:
    print("y is positive")
else:
    print("y is non-positive")
```

```
y is non-positive
```

```
# For loop
for i in range(1, 6):
    print(i)
```

```
1
2
3
4
5
```

##Question: Write python code to print the sum of even numbers between 1 and 100

```
sum_of_even_numbers_variable=0
for num in range(1, 100):
  if num%2==0:
    sum_of_even_numbers_variable+=num;
# Print the sum of even numbers
print("the total sum is ",sum_of_even_numbers_variable)
```

```
the total sum is  2450
```

###Important: The outer bound value of range function will not be included for the iteration

##---
Functions---

- Functions are used to group code together and perform a specific task.
- Functions can be defined using the `def` keyword.
- Functions help in redundancy

```
# Function definition
def greet(name):
    """This function greets the person passed in as a parameter."""
    return f"Hello, {name}!"

# Function call
user_name = "David"
greeting_message = greet(user_name)

# Print the result
print(greeting_message)

Hello, David!
```

##Question:Complete the below code

```
#calculate the area of the Rectangle
#hint:  take length and width as parameters and get the values of the
parameters from the user
def calculate_area():
  area_of_rectangle=length*width
  return area_of_rectangle



# Get user input
length = float(input("Enter the length of the rectangle: "))
width = float(input("Enter the width of the rectangle: "))

# Calculate and print the area
area = calculate_area()
print(f"The area of the rectangle is:{area}")

Enter the length of the rectangle: 1.1
Enter the width of the rectangle: 2.2
The area of the rectangle is:2.4200000000000004
```

#---LISTS IN
PYTHON---

In Python, a list is a versatile and mutable data structure that allows you to store and manipulate sequences of items. Lists are widely used for various tasks due to their flexibility and ease of use. This tutorial will cover the basics of Python lists, including creation, manipulation, and common operations.

#Creating Lists You can create a list in Python by enclosing a comma-separated sequence of elements within square brackets []. Elements in a list can be of different data types, and a list can contain elements of various types.

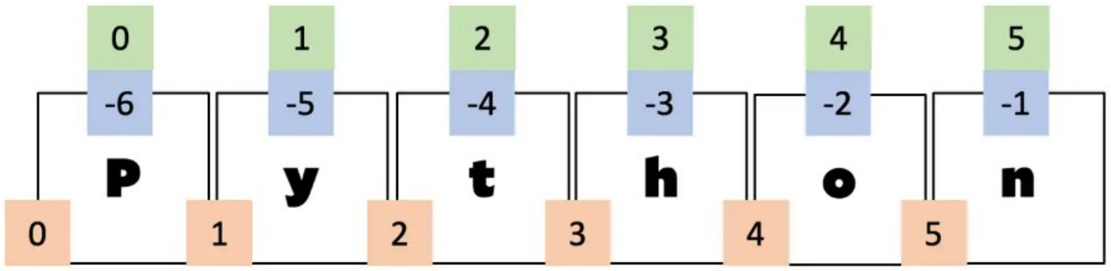
```
# Creating a list
my_list = [1, 2, 3, "apple", "banana", True, 3.14]
```

#Accessing Elements You can access individual elements of a list using indexing. Python uses zero-based indexing, where the first element has an index of 0.

```
print(my_list[0])
print(my_list[1])
print(my_list[-1])  # Negative index counts from the end
```

```
1
2
3.14
```

#Slicing Lists Slicing allows you to create a new list that is a subset of an existing list. It is done using the colon : operator.



```
# Slicing
subset = my_list[1:4]  # Elements at index 1, 2, and 3
print(subset)
```

```
[2, 3, 'apple']
```

```
# Adding elements
my_list.append("orange")  # Adds "orange" to the end of the list
print(my_list)
```

```
[1, 2, 3, 'apple', 'banana', True, 3.14, 'orange']
```

```python
my_list.pop()
print(my_list)
```

```
[1, 2, 3, 'apple', 'banana', True, 3.14]
```

##QUESTION: Complete the below code Consider the below list and and write the code for the output given

```python

# Creating a list
list_1 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Slicing the list from index 2 to 6 (excluding 6)
sliced_list = list_1[2:6]

# Printing the sliced list
print(sliced_list)
```

```
[3, 4, 5, 6]
```

Output: [3, 4, 5, 6]

```python
#prompt: write code to insert element at two places in a list

#Create a list
list_2 = [1, 2, 3, 4, 5]

list_2.insert(2,'a')
list_2.insert(5,'b')

# Print the modified list
print(list_2)
```

```
[1, 2, 'a', 3, 4, 'b', 5]
```

Output: [1, 2, 'a', 3, 4, 'b', 5]

##Calculating length of the list

```python
# Create a list
list_3 = [1, 2, 3, 4, 5]

# Get the length of the list
list_length = len(list_3)

# Print the original list and its length
print("Original List:", list_3)
print("Length of the List:", list_length)
```

```
Original List: [1, 2, 3, 4, 5]
Length of the List: 5
```

##SUMMARY

Explain your take on the techniques covered in the lecture. Your Answer:

variables and datatypes

1. Here ,the python the variable are declared using snake case or the camel case where all the lower case letters are being used in snake case and in camel case , the first letter of second word is being captialized.

2. Regarding datatypes , there are like 6 data types These are inbuilt datatypes and these are used to represent different types / or groups of values .

We mainipulate these in order to store our data effectively and accordingly

Here , some of these are numeric and some other are sequence types , set types , map types like dict , boolean types and none type to represeent a null .

- int
- float
- string = str
- list
- tuple
- set
- dictionary = dict
- boolean = bool
- none

Control Flow elements

Here , these are used to control the flow of our code These basically execute only if a set of conditions are true . only then the code is being channeled. THink of code as a water flow and control flow elements are like tunnels which allow entry if only certain conditions are met .

1. if - this allows only the conditon is matched in if statement .
2. elif = this is used to check multiple conditions in a sequence so that if it matches , that block of elif gets executed .
3. else - this is usually executed if no condition is executed and the remaining if ,elif are all falsee .
4. while - a block of code is executed under it only untill the condition is met .

operators

these are used in performing operations on data.

Lists in python .

1. it is one of the sequence type data type that is used in most cases . This data structure allows us to store a collection of datatypes in one type .
2. these are mutable means the 'modify' ability can be acheived .
3. Lists can be created using the [] sign .
4. elements are seperated by ','.
5. So , the data structures follow a style to address its elements one by one . i.e "indexing" . a value is given to each element and appends to the next one . This is done to locate the element easily .
6. In python programming lang - it follows zero based indexing .

accessing lists

print(your list name [index number]) this allows you to access a desired element in your list . . Note that negative indexing is also supported which is rearely used .

slicing the lists in python

slicing can done using a syntax new_list_name= list[index range 1 : index range 2 : step value] this allows us to get a sub portion from a list .

modifying lists

the mutability feature of lists allows you to change the elements present in the list.

exisiting_list=["blah","blah","blah"] modded_list[0]="moreblahhhhh"

discussion about inbuilt methods.

There are several in built methods to manipulate lists such as apend , insert , remove , pop and extend .

- insert can allow to insert element at any index location .
- append allows addition of element at a end
- pop can delete a element , its based on index specific.
- remove is a value based remover , which removes the provided value from the list .

- len method can result in calculating the length of the elemnt .