GOVERNMENT COLLEGE
KASARAGOD

Open Data Lab
GCK

Micro-course in

DATA ANALYSIS

Session – 2
Introduction to Python

Open Data Lab
GCK

Why python popular ?

- Beginners friendly
- Cross platform
- Huge community
- Wide Ecosystem

Features of Python

- High level language
- Easy to learn
- Interpreted
- Object oriented

Applications of Python

- Data Analysis
- Scientific and computational applications
- Web development
- Game development

- Open-source distribution for Python and R
- Used for data science, machine learning, deep learning etc.
- More than 300+ libraries for data analysis
- Package versions are managed by package management system called Conda

- Open-source web application
- Allows you to create and share documents that contain live code, equations, visualisations and narrative text.
- Markdowns and other additional functionalities are available
- Best platform for getting started with data analysis using programing language.

Open Data Lab
GCK

Introduction to Jupyter Notebook

1. Creating a new Notebook
2. Renaming the Notebook
3. File menu
4. Kernel:
    1. portion of OS code that is always resident in the memory.
5. Command icon palette
6. Cell: Code/Markdown
7. Command mode and Edit mode
8. Running the code cells

Variables

- A way to label data
- Used to store information to be referred and manipulated in a computer program

Rules for variable names:

- Must start with a letter or underscore
- Cannot start with a number
- Can contain only alpha-numeric characters and underscores
- Are case sensitive (conventionally we use small case)
- Keywords in python can’t be used

Data Types

Determines the type of data that is stored

Different data types in Python:

- Number
- String
- List
- Tuple
- Dictionary

Python is a dynamic programming language

String Data Type

Len function

To get the length of the string

Count function

To count the number of occurrence of a single character in a variable

List

- Used to store a group of elements or objects
- Heterogeneous
- Dynamic
- Can store duplicate elements
- Mutable

- Indexing & Negative Indexing
- Slicing
- len & count
- Replace
- Append & insert
- Remove and pop
- Extend a list with another

Tuple

- Same as list, but immutable
- Sequence of immutable python objects

Dictionary

- Represents a group of objects as key-value pair
- Dictionaries are enclosed in curly braces { }
- Key and Values are separated by colon :
- Different key-value pairs are separated by commas
- Mutable
- Duplicate keys are not allowed, but duplicate values are allowed
- Can’t use indexing and slicing concept

Operators

Used to perform different operations on variables and values

Python divides operators into following groups:

- Arithmetic
- Relational
- Logical
- Membership
- Assignment
- Identity
- Bitwise

Arithmetic Operators

1. Addition (+)
2. Subtraction (-)
3. Multiplication (*)
4. Division (/)
5. Modulus (%)
6. Exponent (**)
7. Floor division (//)

Relational Operators

Here we are comparing operands to get a value

1. Greater than (>)
2. Less than (<)
3. Equal to (==)
4. Not equal to (!=)
5. Greater than or equal to (>=)
6. Less than or equal to (<=)

Normally return a Boolean value

Logical Operators

Used to construct compound condition.

Each single condition gives a Boolean value which is evaluated to return final Boolean value.

1. **and** : if both argument is true, result is true
2. **or** : if at least one argument is true, result is true
3. **not** : compliment the Boolean value

Identity Operators

Used to determine whether a value is of certain class or type

1. **is** : Evaluates to true if variable on either side of operator points to same object
2. **is not** : Evaluates to false if the variable on either side of operator points to same object

Membership Operators

Used to validate the membership of the value in a sequence, i.e., strings, lists or tuples

1. **in** : used to check if a value exist in a sequence or not.
2. **not in**: evaluates to true if it doesn't find the variable in the specified sequence.

Assignment Operators

Used to assign value to a variable

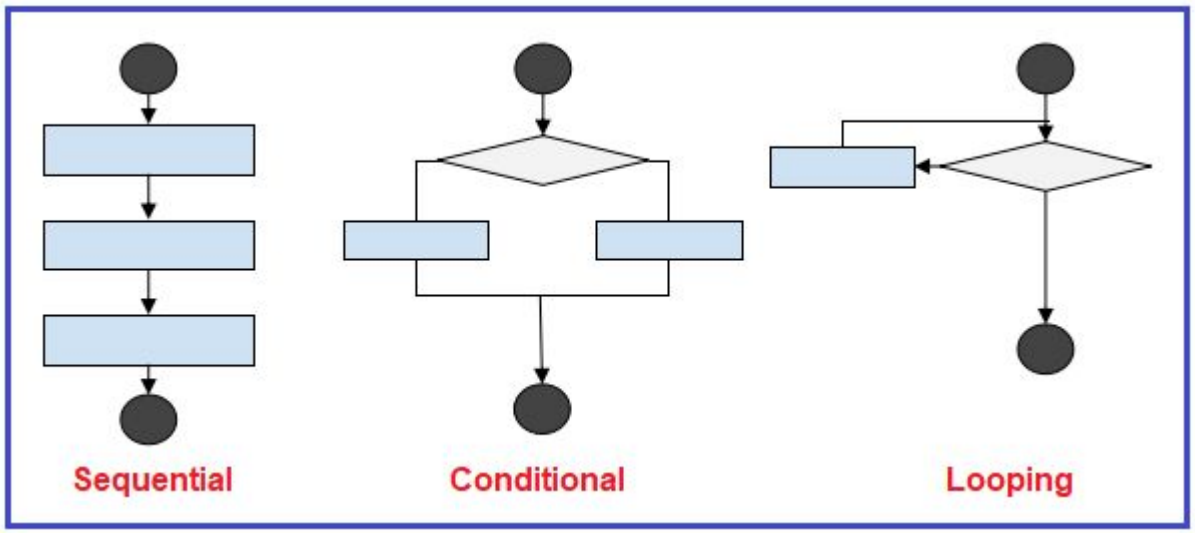
Python Assignment Operators		
Operator	**Example**	**Equal to**
=	a = 20	a = 20
+=	a += b	a = a + b
-=	a -= b	a = a - b
*=	a *= b	a = a * b
/=	a /= b	a = a / b
%=	a %= b	a = a % b
//=	a //= b	a = a // b
**=	a **= b	a = a ** b
&=	a &= b	a = a & b
|=	a |= b	a = a | b
^=	a ^= b	a = a ^ b
>>=	a >>= b	a = a >> b
<<=	a <<= b	a = a << b

Control Structures

A way to specify flow of control in a programme

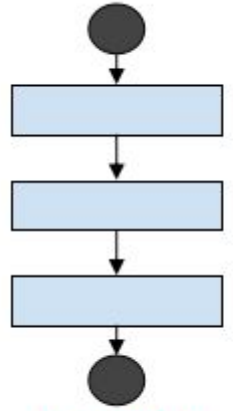
It analyses and chooses in which direction a program flows based on certain parameters and conditions.

Three ways in which a statement can be executed:



Sequential:

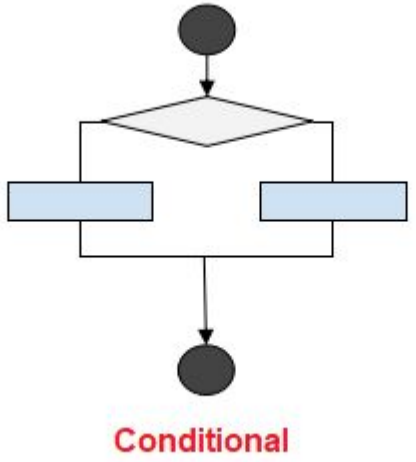
Statements are executed one after the other in a sequential manner



Conditional:

Statements are executed based on certain condition

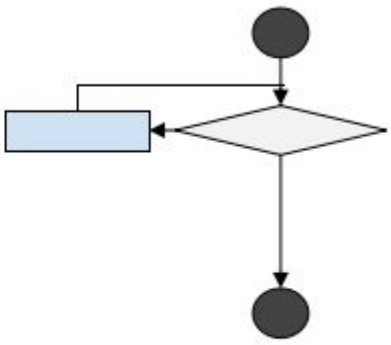
if..else control structures are used in this case



Looping (or Iteration):

Set of statements are executed repeatedly until the condition becomes false.

for and while are used for looping.



Looping

for loop:

Iterates over a compound data type like a string, list or tuple.

During each iteration one member of compound data type is assigned to loop variable

while loop

Logical expression in front of *while* loop is evaluated and if it is true, body of the while loop is executed.

Process is repeated until condition becomes false.

⚠️ we should have some statement inside body of *while* loop which makes the condition falls after few iteration

Modify loops: break and continue

break statement is used to terminate a loop, if some condition is met.

continue keyword is used to suspend a particular iteration in a for/while loop and continues to next iteration

Functions

A block of organised, reusable code that is used to perform a single related action.

1. In-built functions: input(), print(), etc.

2. User-defined functions: functions created by user.

   we define a function with keyword **def**


Thank
You
Open
Data
Lab
GCK