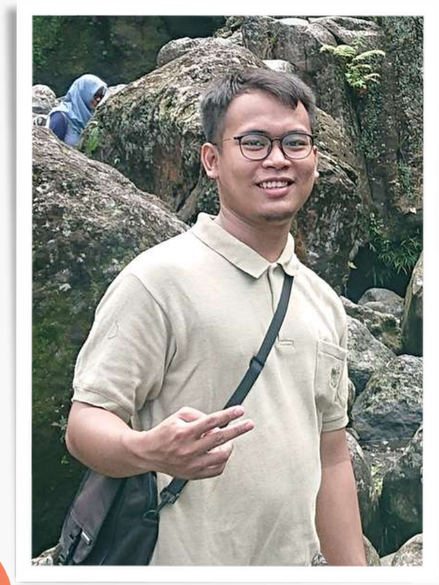
Data Science (Python)

Part 1

Created by
Group 2

Our Team Members

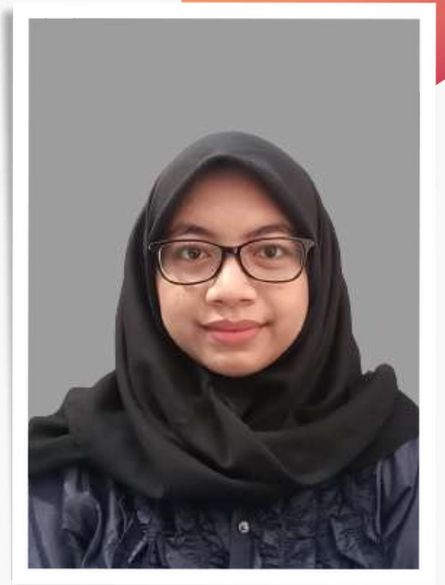


M. KAMAL JAZA

RISMA ASHALI

M. IRVAN A.

RIZAL M.K



SHAFIRA A. M.

1

Python Data Types

Numbers, String, and Boolean

The concept of data type is crucial in programming.
Different types of data can be stored in variables, and different types can perform various functions.
The following categories of data types are included by default in Python:

1. Numbers (int, float, complex)
2. String (str)
3. Boolean (bool)

```
a = 15
type(a)
int
b = 7.5
type(b)
float
c = 2-5j
type(c)
complex
```

```
x = 'Spongebob'
type(x)
str
d = 50
e = 52
d>e
False
e>d
True
```

List

List items are ordered, changeable, not have to get the same data type, and allow duplicate values. List uses []

```
a = [2, 2.7, 'bestie', 1.9+2j, 'patrick']
```

```
a[1]
```

```
2.7
```

```
a[2]
```

```
'bestie'
```

```
a[-2]
```

```
(1.9+2j)
```

```
type (a[3])
```

```
complex
```

note: in python order starts 0,1,2,3.....

Operation List

Range of Indexes

You can specify a range of indexes by specifying where to start and where to end the range. When specifying a range, the return value will be a new list with the specified items.

Note:
Start (index) = included
End (index) = not included

```
#syntax
a = [5, 6.7, "delapan"]
print(a[0:2])
#output
[5, 6.7]
```

```
#syntax
a = [5, 6.7, "delapan"]
print(a[:2])
#output
[5, 6.7]
```

```
#syntax
a = [5, 6.7, "delapan"]
print(a[1:])
#output
[6.7, 'delapan']
```

Range of Negative Indexes

If you want to start the search from the **end** of the list, you can use specify negative indexes.

```
#syntax
a = [5, 6.7, "delapan"]
print(a[-3:-1])

#output
[5, 6.7]
```

Note:
index n = index -1
index n-1 = index -2
index n-2 = index -3
etc.

Range of Indexes with 3 Parameters

Parameter	Description
(a) Start	An index specifying at which position to start (included).
(b) Stop	An index specifying at which position to stop (not included).
(c) Step	An integer number specifying the incrementation.

```
#Syntax
variable[a:b:c]
```

Example :

```
In [4]: c = [6,7,8,9,10,11,12,13]
        c[0:8:2]

Out[4]: [6, 8, 10, 12]
```

Case #1

Return the third, fourth, and fifth item!

```
In [1]: b = [5, 10, 12, 2.8, 6.7, 9.5, 13 >= 9, 5+3j, "kelompok 2","MyEduSolve"]

In [2]: b[2:5]

Out[2]: [12, 2.8, 6.7]
```

Note:
The search will start at index 2 (included) and end at index 5 (not included)

Case #2

Return the last item!

```
In [3]: b = [5, 10, 12, 2.8, 6.7, 9.5, 13 >= 9, 5+3j, "kelompok 2","MyEduSolve"]
        b[-1:]

Out[3]: ['MyEduSolve']
```

index 9 = index -1
index 8 = index -2
index 7 = index -3
index 6 = index -4
....
index 0 = index -10

List Length

To determine how many items a list has, use the **len()** function.

```
In [5]: c = [6,7,8,9,10,11,12,13,14,15]
        len(c)

Out[5]: 10
```

Replace

To change or replace the value of a specific item, refer to the index number.

```
In [7]: c = [6,7,8,9,10,11,12,13,14,15]
        c[0] = 1
        print (c)

        [1, 7, 8, 9, 10, 11, 12, 13, 14, 15]
```

Case

Replace the value of specific item then search for the length now!

```
In [3]: x = [5, 10, 12, 2.8, 6.7, 9.5, 13 >= 9, 5+3j, "kelompok 2","MyEduSolve"]
        len(x)

Out[3]: 10

In [4]: x[2] = 15
        len(x)

Out[4]: 10

In [5]: print(x)

        [5, 10, 15, 2.8, 6.7, 9.5, True, (5+3j), 'kelompok 2', 'MyEduSolve']
```

Add List Items

Append

To add an item to the **end** of the list.

```
In [1]: arr = [1,2,3]
        arr.append(9)
        print(arr)

        [1, 2, 3, 9]
```

Extend

To **merges** elements from another list to the current list.

```
In [2]: arr = [1,2,3,9]
        arr.extend([10,11])
        print(arr)

        [1, 2, 3, 9, 10, 11]
```

Insert

To insert a list item at a **specified index**.

```
In [3]: arr = [1, 2, 3, 9, 10, 11]
        arr.insert(3,7)
        print(arr)

        [1, 2, 3, 7, 9, 10, 11]
```

Case

Create a list of at least 10 elements then add the items at the end of the list, combine them with another list and insert an item on a specific order!

Solution

```
In [6]: b = [5, 10, 12, 2.8, 6.7, 9.5, 13 >= 9, 5+3j, "kelompok 2","MyEduSolve"]
        b.append("Kampus Merdeka")
        print(b)

        [5, 10, 12, 2.8, 6.7, 9.5, True, (5+3j), 'kelompok 2', 'MyEduSolve', 'Kampus Merdeka']

In [7]: b = [5, 10, 12, 2.8, 6.7, 9.5, 13 >= 9, 5+3j, "kelompok 2","MyEduSolve"]
        b.extend([20,22])
        print(b)

        [5, 10, 12, 2.8, 6.7, 9.5, True, (5+3j), 'kelompok 2', 'MyEduSolve', 20, 22]

In [15]: b = [5, 10, 12, 2.8, 6.7, 9.5, True, (5+3j), 'kelompok 2', 'MyEduSolve', 20, 22]
         b.insert(2,8)
```

Remove List Items

del

To removes the specified **index**.

```
In [7]: arr = [1, 2, 3, 7, 9, 10, 11]
        del arr[1]
        print(arr)

        [1, 3, 7, 9, 10, 11]
```

remove

To removes the specified **item**.

```
In [4]: arr = [1, 2, 3, 7, 9, 10, 11]
        arr.remove(9)
        print(arr)

        [1, 2, 3, 7, 10, 11]
```

pop

To removes the **last** element of a list.

```
In [5]: arr.pop()

Out[5]: 11

In [6]: print(arr)

        [1, 2, 3, 7, 10]
```

Case

Create a list then delete certain items, remove items at the end of the list, and remove items using a specified index.

Solution

```
In [9]:  b = [1,2,6,9,10,8,9,7,9,15]
         b.remove(10)
         print(b)

         [1, 2, 6, 9, 8, 9, 7, 9, 15]

In [10]: b = [1, 2, 6, 9, 8, 9, 7, 9, 15]
         b.pop()

Out[10]: 15

In [11]: b = [1, 2, 6, 9, 8, 9, 7, 9, 15]
         del b[1]
         print(b)

         [1, 6, 9, 8, 9, 7, 9, 15]
```

Index & Count List

Index

The **index()** method returns the position at the first occurrence of the specified value.

```
In [5]: arr = [1, 2, 3, 7, 9, 10, 11]
        arr.index(9)

Out[5]: 4
```

Count

The **count()** method returns the number of elements with the specified value.

```
In [4]: a = [2,1,1,1,3,5,4]
        a.count(1)

Out[4]: 3
```

Case

Determine the index of certain items and returns the number of elements with the specified value!

Solution

```
In [12]: b = [1, 6, 9, 8, 9, 7, 9, 15]
         b.index(8)

Out[12]: 3

In [13]: b = [1, 6, 9, 8, 9, 7, 9, 15]
         b.count(9)

Out[13]: 3
```

Sort List

Ascending

sort() method that will sort the list ascending, by default.

```
In [1]: a = [2,1,1,1,3,5,4]
        a.sort()
        print(a)

        [1, 1, 1, 2, 3, 4, 5]
```

Descending

To sort descending, use the keyword argument **reverse = True**.

```
In [2]: a = [2,1,1,1,3,5,4]
        a.sort(reverse=True)
        print(a)

        [5, 4, 3, 2, 1, 1, 1]
```

Reverse

The **reverse()** method reverses the current sorting order of the elements.

```
In [3]: a = [2,1,1,1,3,5,4]
        a.reverse()
        print(a)

        [4, 5, 3, 1, 1, 1, 2]
```

Case

Create a list of some random elements then sort of the smallest to the largest and vice versa. After that reverses the current sorting order of the elements!

Solution

```
In [14]: b

Out[14]: [1, 6, 9, 8, 9, 7, 9, 15]

In [15]: b.sort()
         print(b)

         [1, 6, 7, 8, 9, 9, 9, 15]

In [16]: b.sort(reverse=True)
         print(b)

         [15, 9, 9, 9, 8, 7, 6, 1]

In [17]: b.reverse()
         print(b)

         [1, 6, 7, 8, 9, 9, 9, 15]
```

Tuple

Tuple is a collection of objects which ordered and immutable.

Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.

Example

```
data = ('Data','Scientist','Kampus Merdeka','MES')
```

```
data[2]
```

```
'Kampus Merdeka'
```

```
data[-3:-1]
```

```
('Scientist', 'Kampus Merdeka')
```

```
data[0:4:3]
```

```
('Data', 'MES')
```

```
data
```

```
('Data', 'Scientist', 'Kampus Merdeka', 'MES')
```

SET

A set is an unordered collection of items. Every set element is unique (no duplicates) and must be immutable (cannot be changed).

However, a set itself is mutable. We can add or remove items from it.
Sets can also be used to perform mathematical set operations like union, intersection, symmetric difference, etc.

Example

```
data = {3,3,3,7,7,7,6,6,6,12,1}
```

```
data
```

```
{1, 3, 6, 7, 12}
```

```
data2 = {2,3,5,1}
```

```
data.union(data2)
```

```
{1, 2, 3, 5, 6, 7, 12}
```

```
data.intersection(data2)
```

```
{1, 3}
```

Dictionary

Python dictionary is an unordered collection of items. Each item of a dictionary has a key/value pair.

Dictionaries are optimized to retrieve values when the key is known.

Example

```
data = {
    'Name' : 'irvanggtq',
    'Age' : 20
    }
```

```
data['Name']
```

```
'irvanggtq'
```

Nested Dictionary

Nested dictionary is a dictionary inside a dictionary. It's a collection of dictionaries into one single dictionary.

Example :

```
data = {1 : {'Name' : 'irvanggtq', 'Age' : 20},
        2 : {'Name' : 'jaza', 'Age' : 20},
        3 : {'Name' : 'rizal', 'Age' : 20},
        4 : {'Name' : 'risma', 'Age' : 20},
        4 : {'Name' : 'shafira', 'Age' : 20}
       }
```

```
print(data[1]['Name'])
```

```
irvanggtq
```

2

Number, Character, and String Transformations

String Slicing

String slicing is the process of obtaining a portion (substring) of a string by using its indices.

Example :

```
a = 'Data Science'
```

```
print(a[2])
```

```
t
```

```
print(a[2:9])
```

```
ta Scie
```

```
print(a)
```

```
Data Science
```

Number, Character, and String Transformations

There are several functions to perform transformations on numbers, strings and characters, such as:

- upper()
- lower()
- rstrip()
- lstrip()
- strip()
- startswith()
- endswith()

Character, and String Transformations

upper()

```
nama = "seo dal mi"
```

```
nama.upper()
```

```
'SEO DAL MI'
```

used to convert entire characters to capital letters

lower()

```
nama.lower()
```

```
'seo dal mi'
```

used to convert entire characters to lowercase

rstrip()

```
a = "Data "
```

```
a
```

```
'Data '
```

```
a.rstrip()
```

```
'Data'
```

remove whitespace to the right of the string or the end of the string

lstrip()

```
a = "   Data"
```

```
a
```

```
'   Data'
```

```
a.lstrip()
```

```
'Data'
```

remove whitespace on the left or beginning of the string

strip()

```
a = "Data "
```

```
a
```

```
'Data '
```

```
a.strip()
```

```
'Data'
```

remove whitespace at the beginning or end of a string

Character, and String Transformations

startswith()

```
materi = "Transformasi angka, karakter dan string"
```

```
materi.startswith("Transformasi")
```

```
True
```

This method returns **True** if the string ends with the specified value, otherwise False.

endswith()

```
materi.endswith("string")
```

```
True
```

This method returns **true** if the string starts with the specified value

Conversion between data types

```
a = 123
```

```
type(a)
```

```
int
```

```
float(a)
```

```
123.0
```

```
type(float(a))
```

```
float
```

To **convert integer to float** in python, you can use the **float()** with the int passed as argument to it.

```
b = 99.8
```

```
type(b)
```

```
float
```

```
int(b)
```

```
99
```

```
type(int(b))
```

```
int
```

To **convert float to integer** in python, you can use the **int()** with the float passed as argument to it.

REPLACING STRING ELEMENTS

The replace() method replaces a specified phrase with another specified phrase.

Case: we want to replace "Matematike" to "Biologi"

```
x = "Hari ini ujian Matematika"
```

```
print(x.replace("Matematika","Biologi"))
```

```
Hari ini ujian Biologi
```

The third parameter on replace can be filled in the number of substrings that want to be replaced.

Case: we want to replace "merah" (first one) into "hijau"

```
b = "Bestie makan bubur kacang merah di rumah"
```

```
print(b.replace("merah","hijau",1))
```

```
Bestie makan bubur kacang hijau di rumah
```

Note: All occurrences of the specified phrase will be replaced, if nothing else is specified.

3

Input Output in Python

Input Output in Python

There are some ways to enter variables on the string, such as:

- **Directly merge variables in the print() statement**

```
a =3
```

```
type(a)
```

```
int
```

```
print('Nilai variabel a adalah',a)
```

```
Nilai variabel a adalah 3
```

Input Output in Python

- **Displaying text(string) can use the format string mechanism**

```
print('Nama saya {}'.format('Na Hae Do'))
```

```
Nama saya Na Hae Do
```

Input Output in Python

- **Using the '%' operator coupled with 'specifier argument'**

Example arguments:
- %s- String
- %d- Int
- %f- decimal

```
nama = "Sweety"
umur = 21
tinggi = 154.5
```

```
print("Nama saya",nama + " umur saya",umur, "dan tinggi badan saya", tinggi)
```

```
Nama saya Sweety umur saya 21 dan tinggi badan saya 154.5
```

```
print('Nama saya {}'.format('Sweety'))
```

```
Nama saya Sweety
```

```
print("tinggi badan %s adalah %.2f cm" % (nama, tinggi))
```

```
tinggi badan Sweety adalah 154.50 cm
```

Input Output in Python

Python allows for user input. That means we are able to ask the user for input. There are some implementation about input() code:

```
umur = input("Umur saya: ")
```

```
Umur saya: 19
```

enter the desired value, for example 19

```
umur = input("Umur saya: ")
```

```
Umur saya: 19
```

```
print(umur)
```

```
19
```

Input Output in Python

```python
nama = input("Masukkan nama:")
usia = input("Usia:")
tinggi = input("Tinggi badan:")
```

```
Masukkan nama: Shin Ha Ri

Usia: 30

Tinggi badan: 170
```

enter the desired value of each variabel

```python
nama = input("Masukkan nama:")
usia = input("Usia:")
tinggi = input("Tinggi badan:")
```

```
Masukkan nama:Shin Ha Ri
Usia:30
Tinggi badan:170
```

4

Check String in Python

Check String in Python

The **isupper()** method returns **True** if all the characters are in upper case, otherwise **False**.
Numbers, symbols and spaces are not checked, only alphabet characters.

```
a = "DATA SCIENCE"
```

```
a.isupper()
```

```
True
```

The **islower()** method returns **True** if all the characters are in lower case, otherwise **False**.
Numbers, symbols and spaces are not checked, only alphabet characters.

```
b = "data science"
```

```
b.islower()
```

```
True
```

Check String in Python

We can perform operations on the results of the operation (chain of method)

```
print("Data".upper().lower())
```

```
data
```

```
print("Data".lower().upper())
```

```
DATA
```

```
print("Data".upper().lower().islower())
```

```
True
```

```
print("Data".upper().lower().isupper())
```

```
False
```

For example, **the last one** is ordered to convert all letters to capital and then changed to small all, after checking whether all letters are uppercase (capital) the answer is no/false

Check String in Python

The **isalpha()** method returns **True** if all the characters are alphabet letters (a-z).

```
a = "Data"
a.isalpha()
```

```
True
```

The **isalnum()** method returns **True** if all the characters are alphanumeric, meaning alphabet letter (a-z) and numbers (0-9).

```
a = "Data2022"
a.isalnum()
```

```
True
```

Check String in Python

The **isdecimal()** method returns **True** if all the characters are decimals (0-9).

```
a = "2022"
a.isdecimal()
```

```
True
```

The **isspace()** method returns **True** if all the characters in a string are whitespaces, otherwise False.

```
a = " "
a.isspace()
```

```
True
```

Check String in Python

```
a = "Data Analyst"
a.istitle()
```

```
True
```

The **istitle()** method returns **True** if all words in a text start with a upper case letter, and the rest of the word are lower case letters, otherwise **False**.

Check String in Python

With a little program [ex: istitle()]

```python
while True:
    print('Insert your name:')
    name = input()
    if name.istitle():
        print(' ')
        print('Hello,', name)
        break
    print('Insert your name correctly')
```

```
Insert your name:

Kim Min Joo
```

if it's **True**

```python
while True:
    print('Insert your name:')
    name = input()
    if name.istitle():
        print(' ')
        print('Hello,', name)
        break
    print('Insert your name correctly')
```

```
Insert your name:
Kim Min Joo

Hello, Kim Min Joo
```

Check String in Python

With a little program [ex: istitle()]

```python
while True:
    print('Insert your name:')
    name = input()
    if name.istitle():
        print(' ')
        print('Hello,', name)
        break
    print('Insert your name correctly')
```

```
Insert your name:

in your dream 007
```

if it's **False**

```python
while True:
    print('Insert your name:')
    name = input()
    if name.istitle():
        print(' ')
        print('Hello,', name)
        break
    print('Insert your name correctly')
```

```
Insert your name:
in your dream 007
Insert your name correctly
Insert your name:
```

5

Formatting in String

Formatting in String

The **zfill()** method adds **zeros** (0) at the beginning of the string, until it reaches the specified length.
If the value of the len parameter is less than the length of the string, no filling is done.

```python
a = 7
b = -0.55
c = 'her'
print(str(a).zfill(2))
print(str(b).zfill(8))
print(c.zfill(10))
```

```
07
-0000.55
0000000her
```

The **center()** method will **center align** the string, using a specified character (space is default) as the fill character.
Syntax: string.center(length, character)

```python
'Data'.center(10)
```

```
'   Data   '
```

```python
'Data'.center(10, 'x')
```

```
'xxxDataxxx'
```

Formatting in String

The **ljust()** method will **left align** the string, using a specified character (space is default) as the fill character.
Syntax: string.ljust(length, character)

```
'Data'.ljust(10)
```

```
'Data      '
```

```
'Data'.ljust(10, 'X')
```

```
'DataXXXXXX'
```

The **rjust()** method will **right align** the string, using a specified character (space is default) as the fill character.
Syntax: string.rjust(length, character)

```
'Data'.rjust(10)
```

```
'      Data'
```

```
'Data'.rjust(10, 'X')
```

```
'XXXXXXData'
```

Formatting in String

Raw string, when you want to type symbol in string but won't read as function using **"r"** or **(\)** if you want insert **escape char/symbol**.

I want to type **Data\nScience** without **\n** read as new line

```
print(r'Data\nScience')
```

```
Data\nScience
```

I want to type **It's good**

```
print('It\'s good')
```

```
It's good
```

Formatting in String

Example with a paragraph

```python
print('My Story Today'.center(78))
print('I\'m a student. I going to school at 7 a.m. for learn code.'.ljust(78))
print('It\'s good to learn code in this era. You can learn too with this link below.'.ljust(78))
print('MyEduSolve.com'.rjust(78))
```

```
                              My Story Today
I'm a student. I going to school at 7 a.m. for learn code.
It's good to learn code in this era. You can learn too with this link below.
                                                                MyEduSolve.com
```

Thank You !

Created by
Group 2