CYS Programming with python

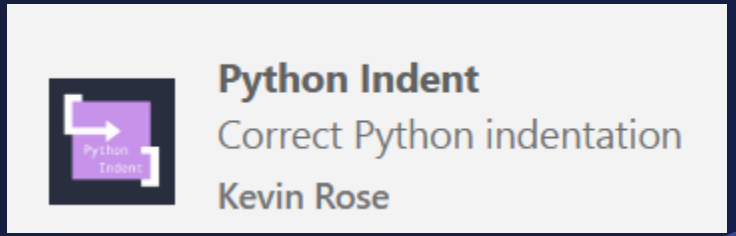
Session 1-2

Done By: Maryam Hajeb

01 Setting up Environment

- Installing python.
- Installing IDE: PyCharm - VC code and python extensions.

VC code python extensions



02

Variables

- A variable is a container for storing data value.

  X=9 name=“Tom” grade=20.52

  marks1, marks2, marks3 = 90, 75, 85

  s1, s2, s3 = 90

- Variable Scope:

  - Local Variable: Defined inside a function.
  - Global Variable: Declared outside all functions and can be accessed anywhere.

Example

```
1   student_name= "Larrissa"
2
3   def total_marks():
4
5       marks1=90
6
7       marks2=50
8
9       total=marks1+marks2
10
11      print(student_name)
12      print(total)
13
14  print(marks1,marks2,total)
```

03 Inputs and Outputs

- input() Function:Used to get user input (always returns a string).
- print() Function: Used to display output to the screen.
- E.g:

```
name = input(“Enter you name: ”)
age = int(input(“Enter your age: ”))
print("My name is “, name)
print(f"My name is {name} and I am {age} years old.")
```

04

Escape characters

- Escape sequences start with backslash \.
- Common escapes: \n (newline), \t (tab), \\ (backslash), \' \" (quotes)

05 Statements

- Single-line Statement: x = 10; y = 20; print(x + y)
- Multi-line Statements: Use backslash \ or parentheses ()

```
total = (10 + 20 +
          30 + 40)
```

- Types of Statements:

  Assignment - Conditional

  Iterative - Transfer/Jump

06

Spacing and Comments

- Indentation: Python uses indentation (spaces/tabs) to define code.
- Comments:
  - single line comments: #
  - multiline comments: ‘‘’ ’’’ OR """ """

07

Data Types

- Numeric: int, float, complex(3-8j)
- String: str
- Boolean
- Sequence:
    * List:[], add-modify-delete, repeat.
    * Tuple: (), repeat
    * Rang: range(10)
- Set: {}, add-delete.
- Dictionary: key-value pairs, {"key" : "value"}

datatypes.py

```
fruits = ["apple", "banana", "cherry"]

fruits.append("mango")

fruits[1] = "kiwi"

del fruits[0]

print(fruits)

colors = ("red", "green", "blue")

print(colors[0])

for i in range(5):

    print(i)

```

datatypes.py

```
15
16
17
18
19
20
21
22
23
24
25
26
27
28
```

```python
numbers = {1, 2, 3, 2}

numbers.add(4)

numbers.remove(1)

print(numbers)

student = {"name": "Ali", "age": 21, "grade": "A"}

print(student["name"])

student["age"] = 22
```

08 Operators

Category	Example
Arithmetic	+ - * / % // **
Comparison	== != > < >= <=
Logical	and or not
Bitwise	`&
Assignment	= += -= *= /=
Membership	in, not in
Identity	is, is not

08 Math functions

- Use Python built-in math functions for basic operations and the math module for advanced math.
- Built-in functions: abs(), round(), pow(), min(), max().
- math module functions: sqrt(), ceil(), floor(), factorial(), pi, sin(), log().

Math_funtions.py

```
print(abs(-7))

print(round(3.14159, 2))

print(pow(2, 3))

import math

print(math.sqrt(25))

print(math.ceil(2.3))

print(math.floor(2.7))

print(math.pi)


```

09

random functions

- Use random module to generate pseudo-random numbers.
- Important functions:

  random(), randint(),randrange()

  choice(), choices(), shuffle(), seed().

random_funtions.py

```
1   import random
2
3   print(random.random())
4
5   print(random.randint(1, 6))
6   print(random.randrange(0, 10, 2))
7
8   lst = [1,2,3,4]
9   print(random.choice(lst))
10
11  random.shuffle(lst)
12  print(lst)
13
14  random.seed(0)
```

10 String operators

- Concatenation: +
- Repetition: *
- Membership: in / not in
- Indexing & slicing: s[i], s[start:stop:step].
- Strings are immutable (operations return new strings).

string_operators.py

```python
a = "Hello"
b = "python"

print(a + " " + b)
print("ha" * 3)
print(b[0])
print(b[1:4])
print("y" in b)
```

11 string functions

- Used to manipulate text.
- lower(), upper(), capitalize(), title()
- strip(), lstrip(), rstrip()
- split(), join()
- replace(), find(), index()
- startswith(), endswith(), format()

string_funtions.py

```
1   s = "  Python Programming  "
2
3   print(s.strip())
4
5   print(s.upper())
6
7   words = s.strip().split()
8
9   print("-".join(words))
10
11  t = "I have 2 computers"
12
13  print(t.replace("2", "three"))
14
15  print(t.find("com"))
```

12

List functions

- Common methods:

  append(), extend(), insert(), remove(), pop(), clear(), index(), count(), sort(), reverse()

- Lists are mutable.

list_funtions.py

```
1   lst = [3, 1, 4]
2
3   lst.append(2)
4
5   lst.extend([5,6])
6
7   lst.insert(1, 9)
8
9   lst.remove(4)
10
11  x = lst.pop()
12
13  print(lst, x)
14
15  lst.sort()
16
17  lst.reverse()
```

13

dictionary functions

- Useful methods:

  get(), keys(), values(),

  items(), update(), pop(),

  popitem(), clear(), setdefault().

dictionary_funtions.py

```python
d = {"a": 1, "b": 2}

print(d.get("a"))

print(d.get("z", 0))

d["c"] = 3

print(list(d.keys()))

print(list(d.values()))

print(list(d.items()))
```

dictionary_funtions.py

```
15
16
17
18
19
20
21
22
23
24
25
26
27
28
```

```python
d.update({"b": 20, "d": 4})

print(d)

print(d.pop("a"))

k, v = d.popitem()

print(k, v)
```

Thanks!

Do you have any questions?