Python 101

Lecture Slide - 03

Rahul Sharma

Formatting output

example1.py

```
n = input("enter your name:")
a = int(input("enter your age:"))
a = a + 1
print("Hi", n, "!", "You will be", a, "years old next year.")
```

output

```
enter your name:Batman
enter your age:30
Hi Batman ! You will be 31 years old next year.
```

F-strings

example1.py

```
n = input("enter your name:")
a = int(input("enter your age:"))
a = a + 1
print(f"Hi {n}! You will be {a:2} years old next year.")
```

output1

```
enter your name:Batman
enter your age:30
Hi Batman! You will be 31 years old next year.
```

output2

```
enter your name:Bruce
enter your age:7
Hi Batman! You will be  7 years old next year.
```

F-strings

Option	Meaning
<	Forces the field to be left-aligned within the available space (this is the default for most objects).
>	Forces the field to be right-aligned within the available space (this is the default for numbers).
=	Forces the padding to be placed after the sign (if any) but before the digits. This is used for printing fields in the form ‘+000000120’. This alignment option is only valid for numeric types. It becomes the default when ‘0’ immediately precedes the field width.
^	Forces the field to be centered within the available space.

F-strings

var.py

```
x = 45
print(f"2x is {2*x}")

s = f"It will return a string {x}"
print(s)

# right align with 5 space
print(f"x = {x:>5}!")
# left align with 5 space
print(f"x = {x:<5}!")
# center align with 5 space
print(f"x = {x:^5}!")
```

output

```
2x is 90
It will return a string 45
x =    45!
x = 45   !
x =   45 !
```

F-strings

var.py

```
x = 5
# force print zeros
print(f"x = {x:05}!")
# right align with 5 space
print(f"x = {x:>05}!")
# left align with 5 space
print(f"x = {x:<05}!")
# center align with 5 space
print(f"x = {x:^05}!")

x = 16
# force output to be hexadecimal
print(f"x = {x:5x}!")
```

output

```
x = 00005!
x = 00005!
x = 50000!
x = 00500!
x =    10!
```

F-strings

var.py

```
x, y = 5, -3
# this will always show sign
print(f"{x:+} {y:+}")
# only show sign when -ve (default)
print(f"{x:-} {y:-}")
# put space for +ve and - for -ve
print(f"{x: } {y: }")
# you can combine this with other options
print(f"{x:+3} {y:+3}")
print(f"{x:>+3} {y:>+3}")
# you can use _ as digit separator
z = 45_500_000
print(f"{z}")
# _ or , can be used on output as well
print(f"{z:,}")
print(f"{z:_}")
```

output

```
+5 -3
5 -3
 5 -3
 +5  -3
 +5  -3
45500000
45,500,000
45_500_000
```

F-strings

var.py

```
pi = 3.141592
print(f"{pi}")
# print 3 decimal places including .
print(f"{pi:.3}")
# print 3 decimal places including .
# with total 5 characters
print(f"{pi:5.3}")
# You can initialize with e as well
G = 6.67e-10
# G will be 6.67 * 10**10
# f is for Fixed-point notation
print(f"{pi:f} {G:f}")
# e is for Exponent notation
print(f"{pi:e} {G:e}")
# g is for General notation
print(f"{pi:g} {G:g}")
```

output

```
3.141592
3.14
 3.14
3.141592 0.000000
3.141592e+00 6.670000e-10
3.14159 6.67e-10
```

Comparison Operators

Operator	Purpose	Examples		
>	Greater than	5 > 3 = True	3 > 5 = False	
>=	Greater than or Equals to	5 >= 3 = True	3 >= 3 = True	3 >= 5 = False
<	Less than	3 < 5 = True	5 < 3 = False	
<=	Less than or Equals to	3 <= 5 = True	3 <= 3 = True	5 <= 3 = False
==	Equal to	3 == 3 = True	5 == 3 = False	
is	Equal to	3 is 3 = True	5 is 3 = False	
is not	Not Equals to	5 is not 3 = True	3 is not 3 = False	

Comparison Operators

comp.py

```
print("5 > 3 =", 5 > 3)
print("3 > 5 =", 3 > 5)
print("5 >= 3 =", 5 >= 3)
print("3 >= 3 =", 3 >= 3)
print("3 >= 5 =", 3 >= 5)
print("3 < 5 =", 3 < 5)
print("5 < 3 =", 5 < 3)
print("3 <= 5 =", 3 <= 5)
print("3 <= 3 =", 3 <= 3)
print("5 <= 3 =", 5 <= 3)
print("3 == 3 =", 3 == 3)
print("5 == 3 =", 5 == 3)
print("3 is 3 =", 3 is 3)
print("5 is 3 =", 5 is 3)
print("5 is not 3 =", 5 is not 3)
print("3 is not 3 =", 3 is not 3)
```

output

```
5 > 3 = True
3 > 5 = False
5 >= 3 = True
3 >= 3 = True
3 >= 5 = False
3 < 5 = True
5 < 3 = False
3 <= 5 = True
3 <= 3 = True
5 <= 3 = False
3 == 3 = True
5 == 3 = False
3 is 3 = True
5 is 3 = False
5 is not 3 = True
3 is not 3 = False
```

If statement

example2.py

```
x = int(input("Enter x: "))
y = int(input("Enter y: "))
if x > y:
    print("x is greater than y")
print("end of program")
```

output1

```
Enter x: 5
Enter y: 3
x is greater than y
end of program
```

output2

```
Enter x: 3
Enter y: 5
end of program
```

If statement

Write a program to take age from user and print whether he/she can vote or not. Assuming citizens can vote after the age of 18.

example3.py

```
a = int(input("Enter your age: "))

if a < 18:
    print("you can not vote yet.")

if a >= 18:
    print("you can vote.")
```

If-else statement

Write a program to take age from user and print whether he/she can vote or not. Assuming citizens can vote after the age of 18.

example4.py

```
a = int(input("Enter your age: "))

if a < 18:
    print("you can not vote yet.")
else:
    print("you can vote.")
```

If-elif-else statement

Write a program to take marks from student and grade him A, B, C or F. Assuming range 100-85 is A, 84-70 is B, 69-50 is C and below 50 is fail.

example5.py

```
m = int(input("Enter your marks: "))

if m >= 85:
    print("A")
elif m >= 70:
    print("B")
elif m >= 50:
    print("C")
else:
    print("F")
```

Logical Operators

and

False and False	False
False and True	False
True and False	False
True and True	True

or

False or False	False
False or True	True
True or False	True
True or True	True

not

not False	True
not True	False

If-else with condition

example6.py

```
a = int(input("Enter apples: "))
o = int(input("Enter oranges: "))
if a >= 5 and o >= 5:
    print("You have plenty of apples and oranges.")
elif a >= 5 or o >= 5:
    print("You have plenty of fruits.")
else:
    print("You need more fruits.")
```

output-1

```
Enter apples: 6
Enter oranges: 7
You have plenty of apples
and oranges.
```

output-2

```
Enter apples: 3
Enter oranges: 7
You have plenty of fruits.
```

output-3

```
Enter apples: 2
Enter oranges: 3
You need more fruits.
```

Assignments

That's all folks!