

Operators, Input, and Type Casting

Dr. Dongchul Kim

Operators

List of Operators

Arithmetic Operators

Assignment Operators

Relational Operators

Logical Operators

Bitwise Operators

Membership Operators

Identity Operators

Arithmetic Operators

Operators	Name	Example
+	Addition	$2 + 3$
-	Subtraction	$5 - x$
*	Multiplication	$y * 4$
/	Division	$4 / 3$
%	Modulus	$10 \% 3$
**	Exponentiation	$2^{**} n$
//	Floor division	$5 // 2$

Assignment Operators

Operator	Example	Equivalent to
=	x = 5	x = 5
+=	x += 1	x = x + 1
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
//=	x //= 3	x = x // 3
**=	x **= 3	x = x ** 3

Relational Operators

Relational operators return a boolean value.

Operator	Name	Example
==	Equal	$x == 5$
!=	Not equal	$x != y$
>	Greater than	$4 > y$
<	Less than	$x < y$
>=	Greater than or equal to	$x >= 3$
<=	Less than or equal to	$x <= y$

Logical Operators

Operator	Description	Example
and	Returns True if both statements are true	<code>x < 5 and x < 10</code>
or	Returns True if one of the statements is true	<code>x < 5 or x < 4</code>
not	Reverse the result, returns False if the result is true	<code>not (x < 5 and x < 10)</code>

Bitwise Operators

Operator	Name	Description
&	AND	Sets each bit to 1 if both bits are 1
	OR	Sets each bit to 1 if one of two bits is 1
^	XOR	Sets each bit to 1 if only one of two bits is 1
~	NOT	Inverts all the bits
<<	Zero fill left shift	Shift left by pushing zeros in from the right and let the leftmost bits fall off
>>	Signed right shift	Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off

Membership Operators

Operator	Description	Example
<code>in</code>	Returns True if a sequence with the specified value is present in the object	<code>x in y</code>
<code>not in</code>	Returns True if a sequence with the specified value is not present in the object	<code>x not in y</code>

```
1 myList = ["apple", "banana", "cherry"]
2 tf = "apple" in myList
3 print(tf)
4 tf = "banana" not in myList
5 print(tf)
6
7
```

```
▶ ↑ /home/dkim/PycharmProjects/CSCI3328/bin/python /home/dkim/PycharmP
■ ↓ True
▣ ↺ False
🔍 ⚙️
★ 2: Favorites ↗ 🗑️
Process finished with exit code 0
```

Identity Operators

Operator	Description	Example
<code>is</code>	Returns True if both variables are the same object	<code>x is y</code>
<code>is not</code>	Returns True if both variables are not the same object	<code>x is not y</code>

```
1 x = ["apple", "banana"]
2 y = ["apple", "banana"]
3 z = x
4 print(x is z)
5 print(x is y)
6 print(x == y)
7 print(x is not y)
8
```

```
▶ ↑ /home/dkim/PycharmProjects/CSCI3328/bin/python /home/dkim/PycharmProject
■ ↓ True
■ ↕ False
■ ⌄ True
■ 🗑 True
Process finished with exit code 0
```

Input

input ()

Introduction to User Input:

- Python allows for user input, enabling interaction with the program.

Using `input ()` Function:

- In Python 3, user input is obtained using the `input ()` function.

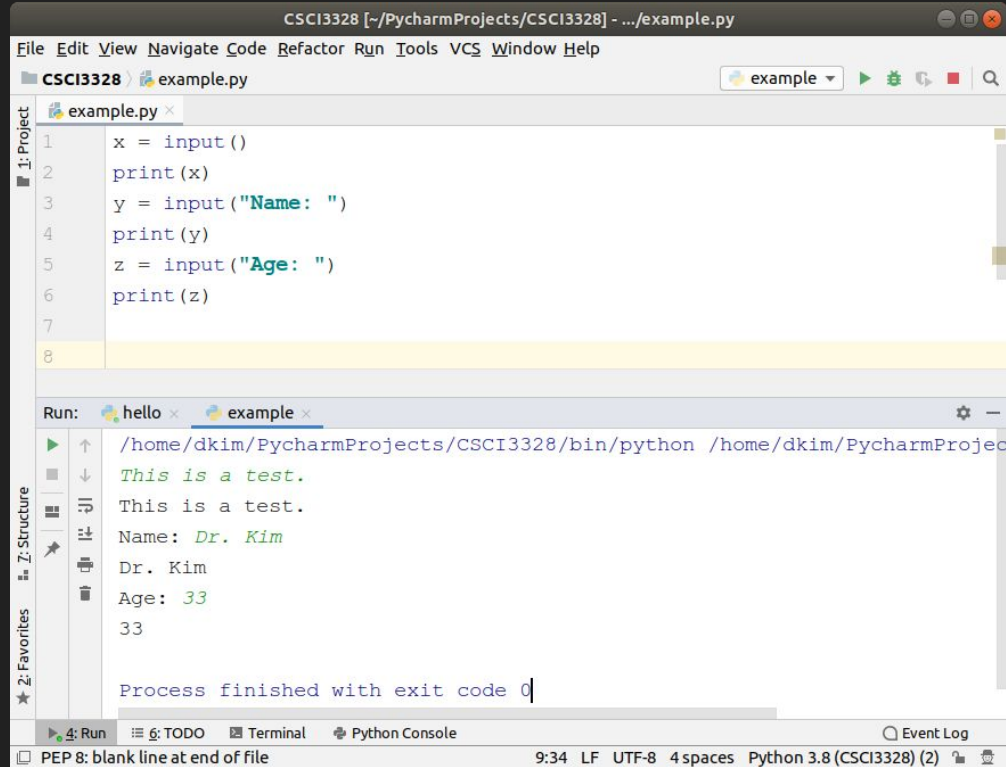
Data Type Consideration:

- Note that user input from the keyboard is treated as a **string** by default, even if numeric values are entered.

Example:

- Example of user input: `name = input("Enter your name: ")`
- In this example, the user is prompted to enter their name, and the input is stored in the `name` variable as a string.

input()



The screenshot shows the PyCharm IDE interface. The main editor window displays a Python script named `example.py` with the following code:

```
1 x = input()
2 print(x)
3 y = input("Name: ")
4 print(y)
5 z = input("Age: ")
6 print(z)
7
8
```

Below the editor, the Run console shows the execution output:

```
Run: hello x example x
/home/dkim/PycharmProjects/CSCI3328/bin/python /home/dkim/PycharmProject
This is a test.
This is a test.
Name: Dr. Kim
Dr. Kim
Age: 33
33
Process finished with exit code 0
```

The status bar at the bottom indicates the file encoding is UTF-8 with 4 spaces, and the Python version is 3.8 (CSCI3328) (2).

```
1 x = input()
2 x = x + 3
3 print(x)
4
```

```
↑ /home/dkim/PycharmProjects/CSCI3328/bin/python /home/dkim/PycharmProjects/CSCI3328/ex
```

```
2
```

```
Traceback (most recent call last):
```

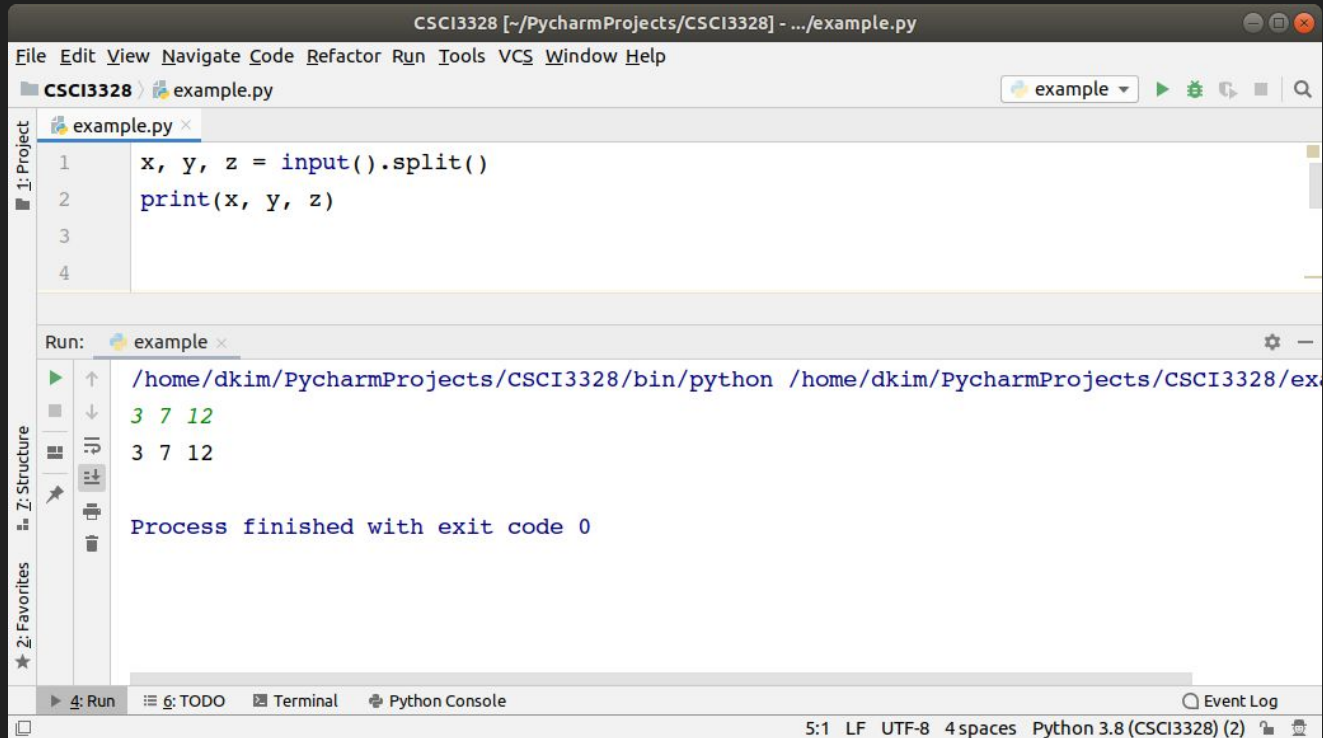
```
  File "/home/dkim/PycharmProjects/CSCI3328/example.py", line 2, in <module>
```

```
    x = x + 3
```

```
TypeError: can only concatenate str (not "int") to str
```

```
Process finished with exit code 1
```


`input().split()`



The screenshot shows the PyCharm IDE interface. The main editor window displays a Python script named `example.py` with the following code:

```
1 x, y, z = input().split()
2 print(x, y, z)
3
4
```

Below the editor, the Run console shows the execution of the script. The command executed is `/home/dkim/PycharmProjects/CSCI3328/bin/python /home/dkim/PycharmProjects/CSCI3328/ex`. The output of the script is `3 7 12`, which is printed on two lines. The console also indicates that the process finished with exit code 0.

The status bar at the bottom of the IDE shows the following information: `5:1 LF UTF-8 4 spaces Python 3.8 (CSCI3328) (2)`.

Type Casting

Type Casting

- Introduction to Type Casting:
 - Type casting allows you to specify a data type for a variable in Python.
- Casting Functions:
 - Python provides casting functions for different data types:
 - `int()`: Converts to an integer from integer, float, or string literals.
 - `float()`: Converts to a float from integer, float, or string literals.
 - `str()`: Converts to a string from various data types, including strings, integers, and floats.
- Example:
 - Example of type casting:
 - `num_str = "123"`
 - `num = int(num_str)`
 - In this example, the string "123" is cast to an integer.

CSCI3328 [~/PycharmProjects/CSCI3328] - .../example.py

File Edit View Navigate Code Refactor Run Tools VCS Window Help

CSCI3328 example.py example

example.py x

```
1 x = "324"
2 print(23 + int(x))
3 y = "7.8"
4 print(32 + float(y))
5 z = 123
6 print("z is "+str(z))
7
8
```

Run: hello x example x

```
↑ /home/dkim/PycharmProjects/CSCI3328/bin/python /home/dkim/PycharmProjec
↓ 347
: 39.8
: z is 123
: Process finished with exit code 0
|
```

4: Run 6: TODO Terminal Python Console Event Log

PEP 8: blank line at end of file 7:1 LF UTF-8 4 spaces Python 3.8 (CSCI3328) (2)



```
1 x = int(input())
2 x = x + 3
3 print(x)
4
```



```
/home/dkim/PycharmProjects/CSCI3328/bin/python /home/dkim/PycharmProjects/CSCI3328/ex
```

```
2
```

```
5
```

```
Process finished with exit code 0
```

Lab 6-1 ~ 6-6

- Submission
 - Capture the output of your program.
 - Upload both the captured image files and Python files on Blackboard.
 - Create a separate file for each lab; do NOT combine multiple labs into a single file.

Lab 6-1

The output format displayed on the console should adhere to the following pattern.
The value you input via the keyboard will be represented in blue.

```
How old are you? 36(enter)
You are 36 years old.
```

Lab 6-1

Correct Input Prompt (30 Points):

Accurately displaying the prompt "How old are you?" on the console (30 Points)

Incorrect or missing prompt (0 Points)

Input Handling and Display (40 Points):

Correctly implementing and displaying age input (40 Points)

Inaccurate input handling or display issues (0 Points)

Output Formatting (30 Points):

Output correctly formatted as "You are [age] years old." (30 Points)

Incorrect or incomplete output format (0 Points)

Total Points: 100

Lab 6-2

The output format displayed on the console should adhere to the following pattern.
The value you input via the keyboard will be represented in blue.

```
When were you born? 1990 (enter)
Processing.....
I think you are 31 years old.
```

Lab 6-2

Correct Input Prompt (30 Points):

Accurately displaying the prompt "When were you born?" on the console (30 Points)

Incorrect or missing prompt (0 Points)

Input and Processing Display (30 Points):

Correctly capturing the year of birth and displaying "Processing....." (30 Points)

Inaccurate input function or missing processing message (0 Points)

Age Calculation and Output (40 Points):

Correctly calculating and displaying age with the format "I think you are [calculated age] years old." (40 Points)

Incorrect age calculation or format (0 Points)

Total Points: 100

Lab 6-3

The output format displayed on the console should adhere to the following pattern. The value you input via the keyboard will be represented in blue.

```
Please input the first integer value? 23(enter)
Please input the second integer value? 10(enter)
Processing.....
Addition: 33
Subtraction: 13
Multiplication: 230
Division: 2.3
Modulus: 3
```

Lab 6-3

Initial Input Prompts (10 Points Each, Total 20 Points):

Accurate prompt and capture of the first integer (10 Points)

Accurate prompt and capture of the second integer (10 Points)

Incorrect or missing prompts (0 Points for each)

Processing Display (10 Points):

Correctly displaying "Processing....." after inputs (10 Points)

Missing or incorrect processing message (0 Points)

Arithmetic Calculations and Output (15 Points Each, Total 60 Points):

Correctly calculating and displaying Addition (15 Points)

Correctly calculating and displaying Subtraction (15 Points)

Correctly calculating and displaying Multiplication (15 Points)

Correctly calculating and displaying Division (15 Points)

Inaccurate calculations or incorrect display (0 Points for each)

Modulus Operation (10 Points):

Correctly calculating and displaying the Modulus (10 Points)

Incorrect modulus calculation or display (0 Points)

Total Points: 100

Lab 6-4

The console output should adhere to the following format. The value you input will be shown in blue. (For example, π is approximately 3.142592.)

```
Please input the height of the cylinder? 17.34(enter)
Please input the radius of the cylinder? 5.6(enter)
Processing.....
The volume of the cylinder is 1708.3424375808 .
```

Lab 6-4

Height Input Prompt (20 Points):

Correctly prompting and capturing the height of the cylinder (20 Points)

Incorrect or missing height prompt (0 Points)

Radius Input Prompt (20 Points):

Correctly prompting and capturing the radius of the cylinder (20 Points)

Incorrect or missing radius prompt (0 Points)

Processing Display (10 Points):

Displaying "Processing....." after inputting height and radius (10 Points)

Missing or incorrect processing display (0 Points)

Volume Calculation and Output (50 Points):

Accurately calculating and displaying the volume of the cylinder with the given inputs (50 Points)

Incorrect volume calculation or display (0 Points)

Total Points: 100

Lab 6-5

The output format displayed on the console should adhere to the following pattern.
The value you input via the keyboard will be represented in blue.

```
What is your name? Dongchul (enter)
Hi Dongchul
What is your favorite sport team? Texas Rangers (enter)
Do you think Texas Rangers will win in the league? Yes (enter)
Thank you!
```

Lab 6-5

Name Input and Response (25 Points):

Correctly prompting for the user's name and responding with "Hi [Name]" (25 Points)

Incorrect prompt or response (0 Points)

Favorite Sport Team Query (25 Points):

Accurately asking for the user's favorite sport team and capturing the response (25 Points)

Inaccurate or missing query for the favorite sport team (0 Points)

Opinion on Team's Success (25 Points):

Prompting for the user's opinion on the team's potential to win in the league and capturing the response (25 Points)

Failure to ask for or capture the user's opinion correctly (0 Points)

Closing Acknowledgement (25 Points):

Displaying a polite closing message such as "Thank you!" after the interaction (25 Points)

Missing or incorrect closing message (0 Points)

Total Points: 100

Lab 6-6

- Create a program that calculates the length of a line segment when provided with the coordinates of its two endpoints.

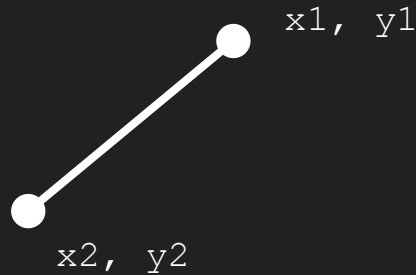
- Input format

`x1 y1`

`x2 y2`

- Output format

`Length`



```
Please input the first end point: 3 4(enter)
```

```
Please input the second end point: 6 8(enter)
```

```
5
```

Lab 6-6

Input Prompt for First Endpoint (20 Points):

Accurately prompting for the first endpoint coordinates (x1 y1) and capturing the input (20 Points)

Incorrect or missing prompt for the first endpoint (0 Points)

Input Prompt for Second Endpoint (20 Points):

Accurately prompting for the second endpoint coordinates (x2 y2) and capturing the input (20 Points)

Incorrect or missing prompt for the second endpoint (0 Points)

Length Calculation (40 Points):

Correctly calculating the length of the line segment using the provided coordinates (40 Points)

Incorrect length calculation or formula usage (0 Points)

Correct Output Format (20 Points):

Displaying the calculated length in the specified output format (20 Points)

Incorrect or missing output format (0 Points)

Total Points: 100