

Day 3



Typecasting in Python:

What is Typecasting?

Typecasting means changing one data type into another data type.

Example: Converting "10" (string) into 10 (integer).

Why do we need Typecasting?

- To perform calculations
- To take input from users
- To avoid errors
- To convert data into required format

Types of Typecasting in Python:

1. Implicit Typecasting (Automatic): Python converts data automatically.
2. Explicit Typecasting (Manual): We convert data manually using functions.

Example: without typecasting we expect the sum, but get the concatenation.

A screenshot of a terminal window in VS Code. The code in main.py is:

```
main.py > ...
1 a = "1"
2 b = "2"
3 print(a + b)
```

The terminal output shows:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
● PS E:\Python\Day1-10\Day3> python main.py
12
```

The output '12' is the result of concatenating the strings '1' and '2'.

Example: converting the string to the integer.

A screenshot of a terminal window in VS Code. The code in main.py is:

```
main.py > ...
1 a = "1"
2 b = "2"
3 print(int(a)+int(b))
```

The terminal output shows:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
● PS E:\Python\Day1-10\Day3> python main.py
3
```

The output '3' is the result of adding the integers 1 and 2 after converting them from strings using the int() function.

Summary:

1. Typecasting means changing data type
2. Two types:
 - Implicit → done automatically by Python
 - Explicit → done by programmer
3. Common functions:
 - int() → integer
 - float() → decimal
 - str() → string
 - bool() → True/False
4. Input is always string, so typecasting is needed

Taking User Input in Python :

What is User Input?

User input means taking data from the user while the program is running. Python uses the `input()` function to take input.

`input()` Function:

Basic Syntax:

```
input("message")
```

Example: a basic example

```
input.py > ...
1 name = input("Enter your name: ")
2 print("Your name is " + name)
3 print("Your name is " , name)

PROBLEMS      OUTPUT      DEBUG CONSOLE      TERMINAL      PORTS

● PS E:\Python\Day1-10\Day3> python .\input.py
Enter your name: Aditya
Your name is Aditya
Your name is Aditya
```

They differ in how the text and the variable are combined:

- In `print("Your name is " + name)`, the `+` joins (concatenates) the string and the variable. Both must be strings, or it will cause an error.
- In `print("Your name is ", name)`, the comma lets `print` handle them separately and automatically adds a space between them. It can also print different data types without error.

So, the main difference is that + joins strings manually, while , lets print combine values more safely and easily.

Important Point (Very Important): Input is always taken as a STRING.

Example:

```
5     a = input("Enter first number: ")
6     print(type(a))

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
```

- PS E:\Python\Day1-10\Day3> python .\input.py
Enter first number: 7
<class 'str'>

Example: as they are treated as strings, they get concatenated.

```
8     a = input("Enter first number: ")
9     b = input("Enter second number: ")
Q 10    print(a+b)

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
```

- PS E:\Python\Day1-10\Day3> python .\input.py
Enter first number: 7
Enter second number: 8
78

Example: typecasting the taken input

```
12    a = input("Enter first number: ")
13    b = input("Enter second number: ")
14    print(int(a)+int(b))

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
```

- PS E:\Python\Day1-10\Day3> python .\input.py
Enter first number: 7
Enter second number: 8
15

Using User Input with Numbers (Typecasting)

To perform calculations, we must convert input.

Example: another way to typecast

```
11
12     a = int(input("Enter first number: "))
13     b = int(input("Enter second number: "))
14     print(a+b)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

● PS E:\Python\Day1-10\Day3> **python .\input.py**
Enter first number: 5
Enter second number: 6
11

Summary:

- User input means taking data from the user
- Python uses `input()` function
- Input is always string by default
- Use typecasting (`int()`, `float()`) for calculations
- `split()` and `map()` help take multiple inputs

--The End--