



Baghdad-ul-Jadheed campus

---

**Submitted to:**

**S Rehan Faheem**

**Submitted by:**

**Alia Saeed**

**Program:**

**BS CS Eve**

**Semester:**

**7<sup>th</sup>**

**Subject:**

**Web design & framework**

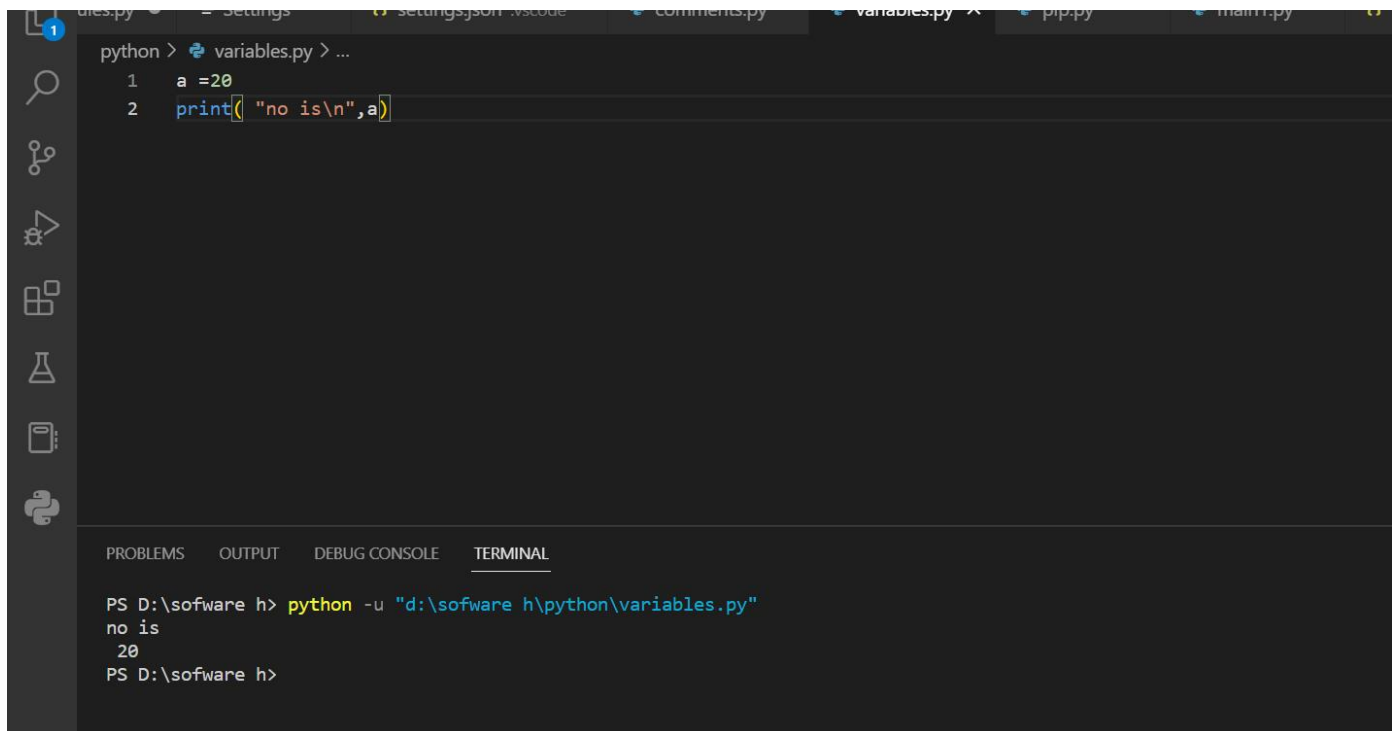
**Roll no:**

**SP20M2BB024**

# Chapter # 2

## ➤ **Variables:**

A Python variable is a reserved memory location to store values.



The screenshot shows a VS Code editor with a file named `variables.py` open. The code in the file is:

```
python > variables.py > ...  
1 a = 20  
2 print("no is\n",a)
```

Below the editor, the `TERMINAL` panel shows the command `python -u "d:\software h\python\variables.py"` being executed. The output of the script is:

```
PS D:\software h> python -u "d:\software h\python\variables.py"  
no is  
20  
PS D:\software h>
```

## ➤ **Data types in Python:**

Data types are the classification or categorization of data items. It represents the kind of value that tells what operations can be performed on a particular data

### **PYTHON DATA TYPES:**

- **Numeric data types: int, float, complex.**

```
python > datatypes.py > ...
1 # 1: Integers
2 b= 12
3 print(b, "\n is int no")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS D:\software h> python -u "d:\software h\python\datatypes.py"
12
is int no
PS D:\software h>
```

- String data types: str.

```
python > datatypes.py > ...
1 # 3: string type
2 name= "alia saeed"
3 print("name is string\n", name)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS D:\software h> python -u "d:\software h\python\datatypes.py"
name is string
alia saeed
PS D:\software h>
```

- Float types:

```
python > datatypes.py > ...
1 # 1: float
2 b= 12.543
3 print(b, "\n is float no")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS D:\software h> python -u "d:\software h\python\datatypes.py"
12.543
is float no
PS D:\software h>
```

## ➤ Rules for defining variable:

- A variable name must start with a letter or the underscore character.
- A variable name cannot start with a number.
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and \_)
- Variable names are case-sensitive

```
python / ...
1  # 1: single line cooments i.e
2  # print your name
3  print("Alia saeed")
4
5  """ 2:
6  multiline comment i.e
7  print two no
8  take input for user
9  """
10 no = input("ENTER A NO\n")
11 print("NO IS:" , no)
12
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

```
PS D:\software h> python -u "d:\software h\python\comments.py"
Alia saeed
ENTER A NO
234
NO IS: 234
PS D:\software h> |
```

## Operator in python:

- Arithmetic operators

```
python > operators.py > ...
1  # 1:Arithmetic opeartors
2  #input for user
3  a = int(input("enter 1st no :"))
4  b = int(input("enter 2nd no :"))
5  #sum
6  add = a + b
7  #sub
8  sub =a - b
9  #multiple
10 multiple = a * b
11 #div
12 divd =a / b
13 #modulas
14 modulas =a % b
15 print("sum is : " , add)
16 print("subtract is " ,sub)
17 print("multiple is:" , multiple)
18 print("divide is:" , divd)
19 print("modulus is:" , modulas)
20

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

enter 1st no :12
enter 2nd no :4
sum is : 16
subtract is 8
multiple is: 48
divide is: 3.0
modulus is: 0
PS D:\software h> 
```

- **Assignment operator**

```
python > operators.py > ...
20 # 1:Assignment opeartors
21 #input for user
22 a = int(input("enter no :"))
23 a= a+5
24 print("assignment oper increment" , a)
25 a= a-15
26 print("assignment oper decrement " ,a)
27 a=25
28 print("assignment oper equal " ,a)
29

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS D:\software h> python -u "d:\software h\python\operators.py"
enter no :25
assignment oper increment 30
assignment oper decrement 15
assignment oper equal 25
PS D:\software h> 
```

- **Logical operator:**

```

python > logical.py > ...
1  #logical operator
2  # and ,not ,or
3  age= 25
4  #and opear
5  if age >=28 and age <12:
6  | print("you are eligible\n")
7  else:
8  | print("you are not eligible\n")
9
10 #OR op
11 temp = 10
12 if temp <19 or temp >6:
13 | print("temprature is good\n")
14 else:
15 | print("temprature is not good\n")
16
17 # not op
18 cloud = False
19 if not cloud:
20 | print("whether is cloudy\n")
21
22
23

```

## Output:

```

python > logical.py > ...
10  #OR op
11  temp = 10
12  if temp <19 or temp >6:
13  | print("temprature is good\n")
14  else:
15  | print("temprature is not good\n")
16
17  # not op
18  cloud = False
19  if not cloud:
20  | print("whether is cloudy\n")
21

```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

```

PS D:\software h> python -u "d:\software h\python\logical.py"
you are not eligible

temprature is good

whether is cloudy

PS D:\software h>

```

- **Comparison operator:**

```

python > comparison.py > ...
1  # comparison opearator
2  # >, >= ,< <=, ==, !=
3  #we compare the value wil be boolean  data type ( true or false )
4  value = 25
5  #less than
6  print(value >23)
7
8  # less than equal
9  print (value >=12)
10
11 # greater than
12 print (value < 26)
13
14 # greater than equal to
15 print (value >=22)
16
17 # equal to
18 print (value == 26)
19
20 # != equal to
21 print (value != 45)
22

```

## Output :

```

10
11 # greater than
...
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS D:\software h> python -u "d:\software h\python\comparison.py"
True
True
True
True
False
True
PS D:\software h>

```

## ➤ **Type () function:**

Python `type()` is a built-in function that returns the type of the objects/data elements stored in any data type .

- **EXAMPLE:**

```
python > type casting.py > ...
1 a=3
2 print(type(a))

PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL
PS D:\software h> python -u "d:\software h\python\type casting.py"
<class 'int'>
PS D:\software h>
```

AND

```
python > type casting.py > c
1 c=3.45
2 print(type(c))

PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL
PS D:\software h> python -u "d:\software h\python\type casting.py"
<class 'float'>
PS D:\software h>
```

AND

```
python > type casting.py > ...
1 name ="alia"
2 print(type(name))

PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL
PS D:\software h> python -u "d:\software h\python\type casting.py"
<class 'str'>
PS D:\software h>
```

## ➤ **Type casting**

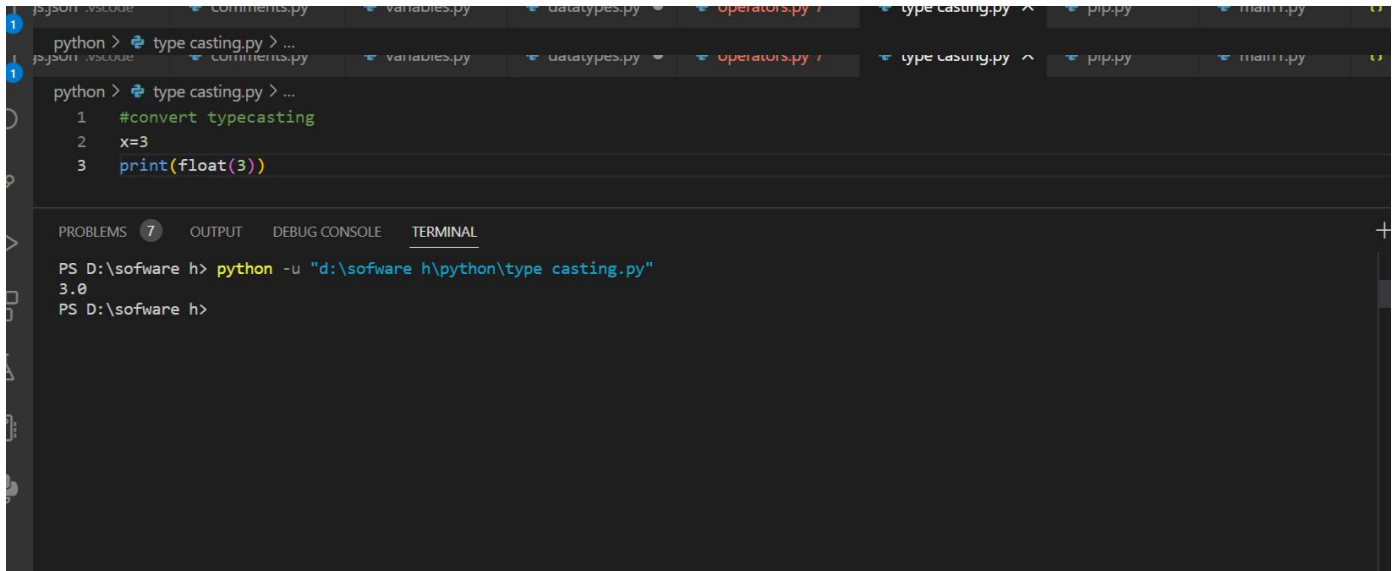


**Type Casting** is the method to convert the variable data type into a certain data type

- **Types of casting:**

- I. - Explicit Conversion(Explicit type casting in python),
- II. Implicit Conversion(Implicit type casting in python)

- **Int to float casting**



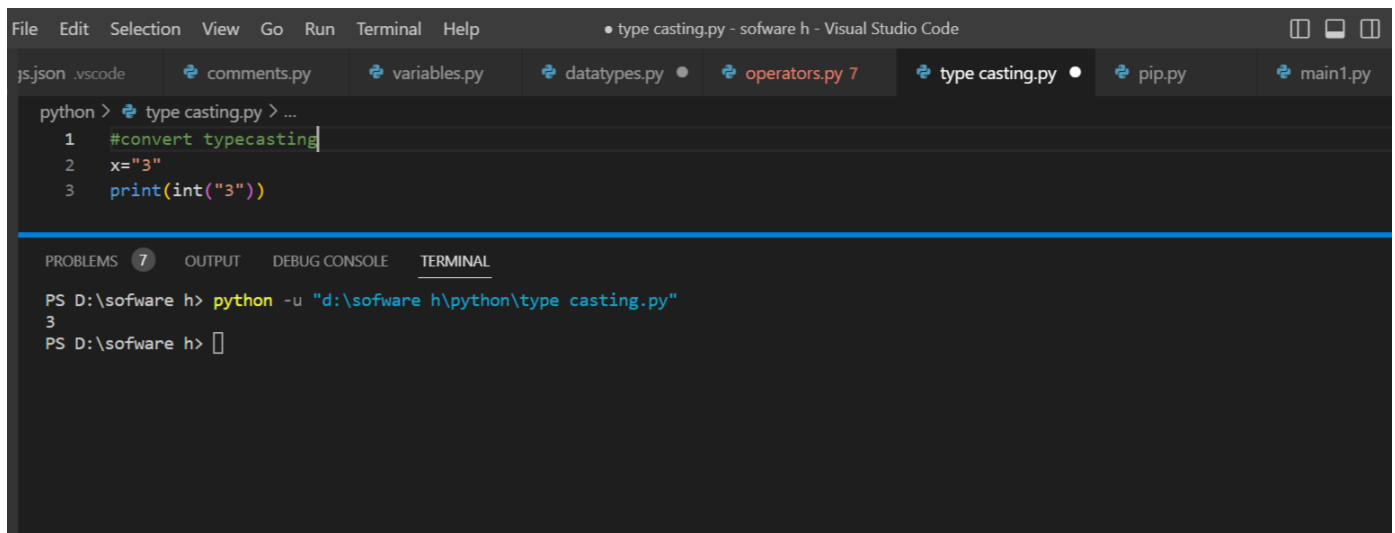
The screenshot shows the Visual Studio Code editor with a file named 'type casting.py'. The code in the editor is:

```
python > type casting.py > ...
1 #convert typecasting
2 x=3
3 print(float(3))
```

The terminal output at the bottom shows the command being executed and the result:

```
PS D:\software h> python -u "d:\software h\python\type casting.py"
3.0
PS D:\software h>
```

- **string to int casting**

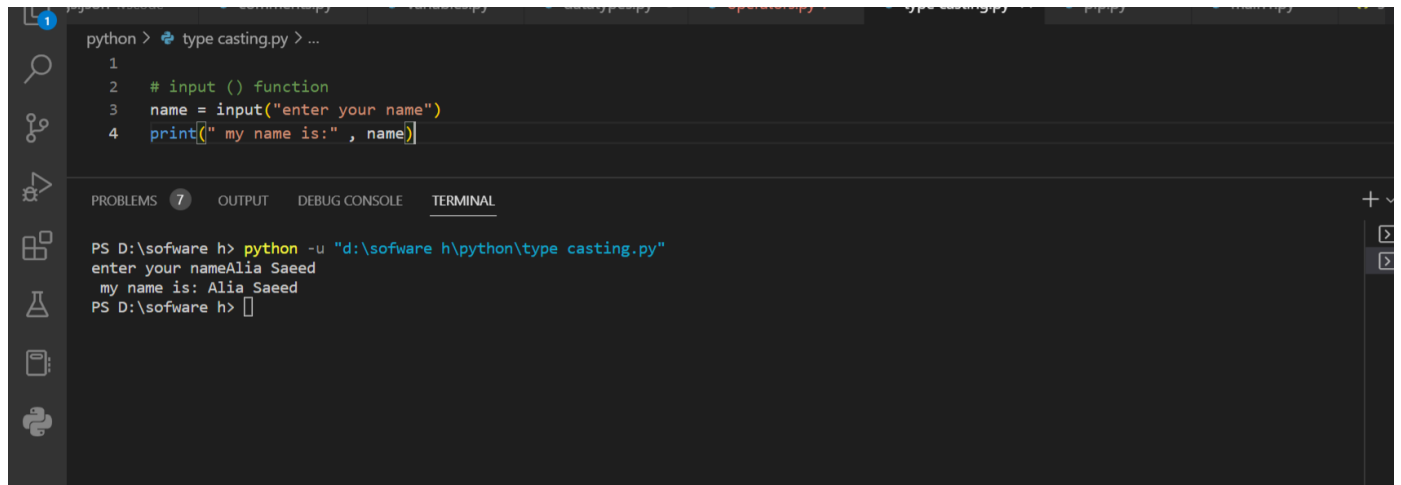


The screenshot shows the Visual Studio Code editor with a file named 'type casting.py'. The code in the editor is:

```
File Edit Selection View Go Run Terminal Help • type casting.py - software h - Visual Studio Code
js.json .vscode comments.py variables.py datatypes.py operators.py 7 type casting.py pip.py main1.py
python > type casting.py > ...
1 #convert typecasting
2 x="3"
3 print(int("3"))
```

The terminal output at the bottom shows the command being executed and the result:

```
PS D:\software h> python -u "d:\software h\python\type casting.py"
3
PS D:\software h> []
```



The image shows a Visual Studio Code editor window with a Python script named `casting.py` and its execution output in the terminal.

**Python Script (`casting.py`):**

```
1  
2 # input () function  
3 name = input("enter your name")  
4 print(" my name is:" , name)
```

**Terminal Output:**

```
PS D:\software h> python -u "d:\software h\python\type casting.py"  
enter your nameAlia Saeed  
my name is: Alia Saeed  
PS D:\software h>
```

The terminal output shows the script being executed from the directory `D:\software h`. The prompt `enter your name` is displayed, followed by the user input `Alia Saeed`. The script then prints `my name is: Alia Saeed`.

