Primitive Types

Primitive Types

Variable

---

variables can store a some data in the memory..so we can use them later.

Types of variables

- Numbers
- Boolians
- Strings

ex :- student_count = **1000** (is call **inteager**)
rating = **4.99** (is call a **float**)
is_published = **False** (is call **Boolian**)
course_name = "**Python_Basic**" (is call **string**)

---

len( ) = for use counting length of charanceters in the string
course = "Hello Python"
print(len(course))

[] = bracet notation to acces specific element
ex:-
course = "Python Programming"
print(course[0]) = result is P
print(course[-1]) = result is g
print(course[0:3]) = result is Pyt // **end index not included**
print(course[0:]) = result is Python Programming
print(course[:3]) = result is Pyt
print(course[:]) = result is Python Programming

Escape Sequences
if we need put " or ' or \ to our programme we can use escape sequences..
Ex :-
course = "Python Programming "
print (course) = result is Python Programming

course = "Python "Programming "
print (course) = result is Syntax error. **How to fix this**

course = "Python "Programming "

print (course) = result is Python "Programming

course = "Python 'Programming "

print (course) = result is Python 'Programming

course = "Python \Programming "

print (course) = result is Python \Programming

course = "Python \nProgramming "

print (course) = result is

Python

Programming

Formatted Strings

like we can use code into code using this ***f"{}"***

`Must use f`

course = "Python "

name = "Mosh"

full = f"{course}{name}"

print(full) = result is Python Mosh

course = "Python "

name = "Mosh"

full = f"{course} {name}"

print(full) = result is Python Mosh

course = "Python "

name = "Mosh"

full = f"{len(course)} {name}"

print(full) = result is 7 Mosh

course = "Python "

name = "Mosh"

full = f"{len(course)} {2+2}"

print(full) = result is 7 4

String Methods

course = "python programming"

course.

`in this course call object and after the . then we use functions but we call term methods because this term come from object oriented` course = "Python programming"

print(course.upper()) = result is PYTHON PROGRAMMING

print(course.lower()) = result is python programming

print(course.title()) = result is Python Programming

course = " Python programming"

print(course.upper())

print(course.lower())

print(course.title())

print(course.strip())

`rstrip = can use drop white space in the right`

`lstrip = can use drop white space in the left`

```
result is =
                PYTHON PROGRAMMING
                python programming
                Python Programming
Python programming
```

course = "Python programming"

print(course.find("pro"))

print(course.replace("p","j"))

print("pro" in course)

print("pr" not in course)

result is =

7

Python jrogramming

True

False

Numbers

x = 1 : Integers

x = 1.1 : Floats

x = 1 + 2j : complex number (#a + bj)

print(10 + 3) = 13

print(10 - 3) = 7

print(10 * 3) = 30

print(10 / 3) = 3.3333333333333335

print(10 // 3) = 3

print(10 % 3) = 1
print(10 ** 3) = 1000

`x = x +3 and x += 3 is exactly same`

Working with Numbers

In mathematics, the term "ceiling" typically refers to the ceiling function, denoted as ⌈x⌉, which rounds a number up to the nearest integer. More formally, the ceiling of a real number x, denoted as ⌈x⌉, is the **smallest integer greater than or equal to x.**

For example:

- ⌈3.2⌉ = 4
- ⌈5.8⌉ = 6
- ⌈-2.5⌉ = -2 (because -2 is the smallest integer greater than or equal to -2.5)

The ceiling function is often used in various mathematical contexts, such as in algorithms, computer science, and discrete mathematics, where you need to ensure that a value is rounded up to the next whole number.

```
import math
print(round(2.9))
print(abs(-5.6))
print(math.ceil(3.1))
```

answers =

- 3
- 5.6
- 4

Type Conversion

Using Codes for conversion

```
int (x)
float(x)
bool(x)
str(x)
```

```
x = input( "x: ")
print(type(x))
```

#now we can see what type input as we inputted.

Some Example:-

```
x = input("x: ")
y = int(x) + 1
print(f"x: {x}, y: {y}")
```

Answer =

```
x: 3
x: 3, y: 4
```

In the boolian
#falsy value is a (" " , 0 , None) other all is a Trualy value

---

What are the primitive types in Python ?

- Strings , Numbers and boolians
  -numbers can be inteager floats and complex numbers