INEURON ASSIGNMENT 1

Python

Kaushik Dey

1. In the below elements which of them are values or an expression? eg:- values can be integer or string and expressions will be mathematical operators.

Ans :

* Expression (Multiplication is a mathematical operator, it's an expression)

'hello' (treated as string so, it's a value)

-87.8 (treated as Numeric: decimal so, it's a value)

- (subtraction is a mathematical operator, it's an expression)

/ (Division is a mathematical operator, it's an expression)

+ (Addition is a mathematical operator, it's an expression)

6 (treated as Numeric: Int, it's a value)

2. What is the difference between string and variable?

Ans: String is a datatype & variable is a placeholder or symbols where we can store data. Data can be anything (string, numeric, Boolean, binary types, sequence types and many more) Variable is a cell reference or its an empty box that we fill with data. In python data type by default auto declared, so no need to mentioned data type in other programming languages like java, .net. and we can't start with numbers or special charecters.

The examples are below.

# can't start with number while assigning any value to variable

```
90p=90

90p

Cell In [22], line 2

   90p=90

    ^

SyntaxError: invalid decimal literal
```

```
@pokl89 = "hello new course"
print(@pokl89)
 Cell In [24], line 1
    @pokl89 = "hello new course"
    ^
SyntaxError: invalid syntax. Maybe you meant '==' or ':=' instead of '='?
```

3. Describe three different data types.

Ans: In python , many data types are there which is built in by default and we can categorised in below tabular format.

Text Type:	Str
Numeric Types:	int, float, complex
Sequence Types:	list, tuple, range
Mapping Type:	Dict
Set Types:	set, frozenset
Boolean Type:	Bool
Binary Types:	bytes, bytearray, memoryview
None Type:	NoneType

but we can discuss three different data types are in follows with some discussion.

1) **Numeric Data Type**: it is used to hold numeric values like int, long, float & complex. here, int holds signed integers of non-limited length. Long holds long integers where float holds floating precision numbers and its accurate upto 15 decimal places. Complex holds complex numbers.
   Some examples are given below.

```
#variable with integer value.
a=10
print("The type of variable having value", a, " is ", type(a))
The type of variable having value 10  is  <class 'int'>
```

```
# variable with float value.
```

```
b=11.2389
print("The type of variable having value", b, " is ", type(b))
The type of variable having value 11.2389  is  <class 'float'>
```

```
# variable with complex value.

 c=2000+4j

 print("The type of variable having value", c, " is ", type(c))

The type of variable having value (2000+4j)  is  <class 'complex'>
```

2) **String Data Type:** we can say that strings are sequences of character data. it is implemented as an array data structure of bytes that stores a sequence of elements, its called characters. The string type in python is called str. Generally, strings are represented by single or double quotes. We can concat, separate the multiple strings in string operation. There are many in built method or function is there. Some examples are given below.

```
# Assigning a string to a variable is done with the variable name followed by an equal
sign and the string:
# single string
a = "kaushik-dey"
print("The value of a is ", a, "and it is", type(a) )
The value of a is  kaushik-dey and it is <class 'str'>
```

```
# You can assign a multiline string to a variable by using three quotes:
# Multi line string
multi_string = """in ineuron learning full stack,
data science course with hands on examples,
4 live classes are done with starting python module
5 assignments given to us to check our knowledge."""
print("The value of a is ", multi_string, "and it is", type(multi_string) )

The value of a is  in ineuron learning full stack,
data science course with hands on examples,
4 live classes are done with starting python module
5 assignments given to us to check our knowledge. and it is <class 'str'>
```

```
# You can assign a multiline string to a variable by using three quotes:
```

```
# New line string
new_line_str = "I'm learning Python.\nI refer to ineuron.\nIt is the most popular
course for Python programmers.\n"
print("The value of a is ", new_line_str, "and it is", type(new_line_str) )

The value of a is  I'm learning Python.
I refer to ineuron.
It is the most popular course for Python programmers.
 and it is <class 'str'>
```

```
# slicing string
slicing_string = "Hello, Kaushik!"
print(slicing_string[2:5])
# slicing from the start
slice_from_start = "Hello, Kaushik!"
print(slice_from_start[:5])
# slice to end
slice_to_end = "Hello, Kaushik!"
print(slice_to_end[2:])
# negative indexing
negative_indexing = "Hello, Kaushik!"
print(negative_indexing[-5:-2])

llo
Hello
llo, Kaushik!
shi
```

```
string_split
# Modifying strings
# string upper
string_upper = "Hello, Kaushik!"
print(string_upper.upper())
# string lower
string_lower = "Hello, Kaushik!"
print(string_lower.lower())
# remove whitespace
string_strip = " Hello, Kaushik! "
print(string_strip.strip())
# replacing strings
string_replace = "Helcome, Ineuron!"
print(string_replace.replace("H", "W"))
# spliting strings
```

```
string_split = "Data Science course, Learning Data Science"
print(string_split.split(','))


HELLO, KAUSHIK!
hello, kaushik!
Hello, Kaushik!
Welcome, Ineuron!
['Data Science course', ' Learning Data Science']
```

```
### Concat strings
string_a = "Welcome to "
string_b = "Ineuron"
final_result = string_a + string_b
print(final_result)
Welcome to Ineuron
```

```
### String Format
course_fees = 20000
course_name = "Full Stack Data Science Course, and the course fees are Rs. {}"
print(course_name.format(course_fees))
Full Stack Data Science Course, and the course fees are Rs. 20000
```

3. **Sequence Data Type : List**

List Data Type **:** List are used to store multiple elements in single variable. In Javascript its called array and in python it's called List. And it is built in and sequence data type in python world. List items are ordered, changeable and allow duplicate values. With the help of len() function we can get its size or length. We can also append or add new items in the existing list. List items can be any data types we can add . we can also access, remove and change list items. We can also sort the list. We can also join the list.

```
# Create a list
custom_list = ["FrontEnd", "Backend", "DataScience"]
print(custom_list)
['FrontEnd', 'Backend', 'DataScience']
```

```
# Allow duplicate
duplicate_list = ["apple", "banana", "cherry", "apple", "cherry"]
print(duplicate_list)
['apple', 'banana', 'cherry', 'apple', 'cherry']
```

```
# get the size of list
length_list = ["FrontEnd", "Backend", "DataScience"]
print(len(length_list))
# Original list items
original_list = ["FrontEnd", "Backend", "DataScience"]
print(original_list)
# Append new items in that list
original_list.append("BlockChain")
print(original_list)
['FrontEnd', 'Backend', 'DataScience']
['FrontEnd', 'Backend', 'DataScience', 'BlockChain']
```

```
# it can contain different data type
list1 = ["apple", "banana", "cherry"]
list2 = [1, 5, 7, 9, 3]
list3 = [True, False, False]
list4 = list1 + list2 + list3
print(list4)
['apple', 'banana', 'cherry', 1, 5, 7, 9, 3, True, False, False]
```

```
# we can also for loop the list
for_loop_items = ["apple", "banana", "cherry"]
for i in range(len(for_loop_items)):
    print(for_loop_items[i])

apple
banana
cherry
```

```
# From Python's perspective, lists are defined as objects with the data type 'list':
mylist_type = ["apple", "banana", "cherry"]
print(type(mylist_type))
<class 'list'>
```

```
# access the list items
thislist = ["FrontEnd", "Backend", "DataScience"]
# list index are starting from 0
print(thislist[1])
Backend
```

```
# remove list items

thislist = ["FrontEnd", "Backend", "DataScience"]

thislist.remove("Backend")

print(thislist)

['FrontEnd', 'DataScience']
```

```
# print the last items

thislist = ["FrontEnd", "Backend", "DataScience"]

print(thislist[-1])

DataScience
```

```
 # pop methond removes specific index

thislist = ["FrontEnd", "Backend", "DataScience"]

thislist.pop(2)

print(thislist)

['FrontEnd', 'Backend']
```

```
# sort the list items

sort_list = ["orange", "mango", "kiwi", "pineapple", "banana"]

sort_list.sort()

print(sort_list)

['banana', 'kiwi', 'mango', 'orange', 'pineapple']
```

```
course_list = ["FrontEnd", "Backend" , "DataScience","BlockChain"]

list2 = [1, 2, 3]

for x in list2:

  course_list.append(x)

print(course_list)

['FrontEnd', 'Backend', 'DataScience', 'BlockChain', 1, 2, 3]
```

```
#we can also extend the list

list1 = ["a", "b" , "c"]

list2 = [1, 2, 3]

list1.extend(list2)

print(list1)

['a', 'b', 'c', 1, 2, 3]
```

4. What is an expression made up of? What do all expressions do?

Ans : An combination of operands and operators is called expression. In any programming language an expression is evaluated as per the precedence of its operators. Or we can say with the help of expression a certain task is performed, so it's a kind of template and with the help of proper expression we can get the value from it. The expression in Python produces some value or result after being interpreted by the Python interpreter.

In python there are many types of expression, so we can discuss it below.

Expression Name	**How it works**

Constant Expressions , This expression holds constant values only	# constant expression
	
	x = 20 + 2.3
	
	print(x)
	
	22.3

Arithmetic Expressions :
 	
An arithmetic expression is a combination of numeric values, operators, and sometimes parenthesis.
	### airthmatic expression
	
	x = 50
	
	y = 20
	
	add = x + y
	
	sub = x - y
	
	pro = x * y
	
	div = x / y
	
	print(add)
	
	print(sub)
	
	print(pro)
	
	print(div)
	
	70
	
	30
	
	1000
	
	2.5

Integral Expressions: it produce only integer results after all computations and type conversions.	# Integral Expressions
	
	a = 13
	
	b = 12.0
	
	c = a + int(b)
	
	print("Integral Expresion",c)
	
	Integral Expresion 25

Floating Expressions: it produce floating point numbers as result after all computations and type conversions.	# Floating Expressions
	
	a = 13
	
	b = 5
	
	c = a / b

	print(c) 2.6

Relational Expressions : arithmetic expressions are written on both sides of relational operator (> , < , >= , <=).	# Relational Expressions
	
	a = 21
	
	b = 13
	
	c = 40
	
	d = 37
	
	p = (a + b) >= (c - d)
	
	print(p)
	
	True

Logical Expressions : These are kinds of expressions that result in either True or False. It basically specifies one or more conditions. It has and, or, not operators to get the logical expressions.	P = (10 == 9)
	
	Q = (7 > 5)
	
	# Logical Expressions
	
	R = P and Q
	
	S = P or Q
	
	T = not P
	
	print(R)
	
	print(S)
	
	print(T)
	
	False
	
	True
	
	True

Combinational Expressions : To use different types of expressions in a single expression, and that will be termed as combinational expressions.	# Combinational Expressions
	
	a = 16
	
	b = 12
	
	c = a + (b >> 1)
	
	print(c)

	22

5. This assignment statements, like spam = 10. What is the difference between an expression and a statement?

Ans : Expression is one side or it has execution only where statement is two sided execution. So in real world example if house is statement or entire structure then rooms are expression , the building blocks of the house. A line or block of code is a statement . like spam=10 means it's a statement where 10 can be divided in 5+5, 2+2+2+2+2, 11-1 all are expression.

Spam =10 => it's a statement

Spam => it's a variable

10=> it's a numeric value but we can provide multiple expression to achieve that.

6. After running the following code, what does the variable bacon contain?

bacon = 22

bacon + 1

Ans :

```
# After running the following code, what does the variable bacon contain?

bacon = 22

bacon + 1

print(bacon)

22
```

7. What should the values of the following two terms be?

'spam' + 'spamspam'

'spam' * 3

Ans :

'spam' + 'spamspam'

'spam' * 3

'spamspamspam'

8. Why is eggs a valid variable name while 100 is invalid?

Ans : any variable name can start with string but in python any variable not starting with number or numeric value because there are lots of problems while **parsing/compiling code** with such variables due to the rule of language design. And also the compiler could confuse it with a number literal. So, its better to start variable name with string. A variable name must start with a **letter or underscore charecters**.

9. What three functions can be used to get the integer, floating-point number, or string version of a value?

Ans : The three in-built functions are given below:

int () => get the integer value.

float() => get the floating point number.

str() => get the string value

10. Why does this expression cause an error? How can you fix it?

'I have eaten ' + 99 + ' burritos.'

Ans : when add this expression in notebook , get the following error.

```
# typeError in python

vv = 'I have eaten ' + 99 + ' burritos.'

vv

---------------------------------------------------------------------------

TypeError                                 Traceback (most recent call last)

Cell In [96], line 1

----> 1 vv = 'I have eaten ' + 99 + ' burritos.'

      2 vv

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

To fix the error can use the following expression.

```
# to fix the error
```

```
vv = 'I have eaten ' + str(99) + ' burritos.'

vv

'I have eaten 99 burritos.'
```