Week-1, Graded Assignment (theory)

(15 marks)

Problem-1

(1 mark)

Question

What is the type of the following expression?

```
1 + 4 / 2
```

(a) `int`

(b) `float`

(c) `str`

(d) `bool`

Answer

(b)

Solution

In the expression, the order of precedence is `/` > `+`. In Python The `/` operator always returns a `float` value and `+` operation with `float` and `int` value also returns a `float` value. So, the above expression will return a `float` value. Hence, Option (b) is correct.

Problem-2

(1 mark)

Question

What is the type of the following expression?

```
1 > 0 and -1 < 0 and 1 == 1
```

(a) `str`

(b) `bool`

(c) `True`

(d) `False`

Answer

(b)

Solution

In the above expression, multiple conditions are connected using a logical operator `and`. Each condition evaluates to `True`. So, the expression always returns `True` which is of a `bool` type. Hence, Option (b) is correct.

Problem-3

(2 marks)

Question

How does the Python interpreter parenthesize the following expression?

```
1 + 3 / 4 ** 2 * 0
```

(a) `1 + (((3 / 4) ** 2) * 0)`

(b) `1 + ((3 / (4 ** 2)) * 0)`

(c) `(1 + 3 / 4) ** (2 * 0)`

(d) All of the above

Answer

(b)

Solution

In the expression, the order of precedence is `**` > `/` , `*` > `+`. According to the precedence order, `(4 ** 2)` will be evaluated first. Operators `/` and `*` have the same precedence and left to right associativity. So, `(3 / (4 ** 2))` will be evaluated first. Thereafter `((3 / (4 ** 2)) * 0)` will be evaluated and finally `1 + ((3 / (4 ** 2)) * 0)` will be evaluated. Hence, Option (b) is correct.

Problem-4

(2 marks)

Question

Convert the following mathematical statement into a Python expression. It is a Multiple Select Question (MSQ).

$$10^3 + 9^3 = 12^3 + 1^3 = 1729$$

(a) `10^3 + 9^3 == 12^3 + 1^3 == 1729`

(b) `10 ** 3 + 9 ** 3 = 12 ** 3 + 1 ** 3 = 1729`

(c) `10 ** 3 + 9 ** 3 == 12 ** 3 + 1 ** 3 == 1729`

(d) `(10 ** 3) + (9 ** 3) == (12 ** 3) + (1 ** 3) == 1729`

Answer

(c), (d)

Solution

- In Python,
  - 10^3 + 9^3 will be equivalent to `10 ** 3 + 9 ** 3` or `(10 ** 3) + (9 ** 3)`
  - 12^3 + 1^3 will be equivalent to `12 ** 3 + 1 ** 3` or `(12 ** 3) + (1 ** 3)`
  - Equal sign used in mathematics `=` is equivalent to `==` operator.

Hence, option (c) and (d) both are correct.

Problem-5

(2 marks)

Question

`E_1` and `E_2` are boolean expressions. Consider the following expression.

```
E_3 = not (E_1 or E_2)
E_4 = (not E_1) and (not E_2)
print(E_3 == E_4)
```

What can you say about the value of the expression given above?

(a) It is `True` if and only if both `E_1` and `E_2` have the same value.

(b) It is `False` if and only if both `E_1` and `E_2` have the same value.

(c) It is always `True`.

(d) It is always `False`.

Answer

(c)

Solution

E_1	E_2	not (E_1 or E_2)	(not (E_1) and not(E_2))	not (E_1 or E_2) == ((not E_1) and (not E_2))
False	False	True	True	True
False	True	False	False	True
True	False	False	False	True
True	True	False	False	True

So, we can see that for all possible values of `E_1` and `E_2`, expression returns `True`. Hence, option (c) is correct.

Problem-6

(2 marks)

Question

`E` is a boolean variable. Consider the following sequence of expressions:

```
not E
not not E
not not not E
not not not not E
.
.
.
```

This pattern keeps repeating for a thousand lines. If line number 500 evaluates to `False`, what is the value of `E`?

(a) `True`

(b) `False`

(c) Cannot be determined

Answer

(b)

Solution

This pattern evaluates `True` and `False` for the alternate line because one `not` operator is added in expression each time. So, if line number `500` evaluates to `False` that means the even-number line evaluates `False` and the odd-number line evaluates `True`. This means the line number `1` which is `not E` will be evaluated as `True`. So, the value of `E` is `False`. Hence, option (b) is correct.

Problem-7

(3 marks)

Question

`E_1` and `E_2` are two boolean variables. Consider the following code.

```
1 E_1 and E_2 and 1 / 0
2 print(E_2)
```

Which of the following scenarios are possible when the code given above is executed? Assume that all scenarios are independent of each other. It is a Multiple Select Question (MSQ).

(a) The code throws an error.

(b) `True` is printed after line-2 is executed.

(c) `False` is printed after line-2 is executed.

(d) None of the above.

Answer

(a), (b), (c)

Solution

E_1	**E_2**	**E_1 and E_2 and 1 / 0**	**print(E_2)**	**Possible scenario for (E_1 and E_2 and 1 / 0)**
False	False	False	False	`E_1` is `False`. Hence, the remaining part will not be evaluated.
False	True	False	True	`E_1` is `False`. So, the remaining part will not be evaluated
True	False	False	False	`E_1` is `True`. Therefore, `E_2` will be checked. Since, `E_2` is `False` so the remaining part will not be evaluated
True	True	Error	No Execution	`E_1` is `True`. Therefore, `E_2` will be checked. Now, since `E_2` is `True`, the remaining part `1/0` will be evaluated which results in error.

As we can see that there are three possible scenarios for execution. So, option (a), (b), and (c) are correct.

Problem-8

(2 marks)

Question

Consider the following string:

```
word = '138412345678901938'
```

For what values of `a` and `b` does the following expression evaluate to `True`? Assume that `a` and `b` are both positive integers.

```
word[a : b] == '123456789'
```

Answer

`a = 4, b = 13`

Solution

For `word = '138412345678901938'` index representation are given below:-

word=	1	3	8	4	1	2	3	4	5	6	7	8	9	0	1	9	3	8
index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

For the condition `word[a : b] == '123456789'` evaluated to be `True`, the start index `a` should be `4` and stop index `b` should be `13` (1 more than the index of last character, because stop index not included in slicing range). Hence, a = 4, b = 13.