PYTHON INTERVIEW QUESTIONS WITH ANSWER DOC2

1. Logical Operators in Python

Definition: Logical operators are used to combine multiple conditional statements and evaluate them to return a

Boolean result (True or False).

Types:

1. AND (and) - Returns True only if all conditions are True

2. OR (or) - Returns True if at least one condition is True

3. NOT (not) - Reverses the logical state (True becomes False and vice versa)

Examples:

print(5 > 3 and 10 > 8) *)*

print(5 > 3 and 2 > 8)

print(5 > 3 or 2 > 8)

print(1 > 3 or 2 > 8)

print(not(5 == 5))

print(not(5 == 3))

2. Difference Between AND and OR Operators

AND requires all conditions to be True, while OR only needs one True condition.

Practical Example:

has_account = True

correct_password = False

print(has_account and correct_password)

print(has_account or correct_password)

3. Membership Operators

Definition: Operators that test if a value is present in a sequence (string, list, tuple, etc.).

Operators:

1. in - Returns True if value exists

2. not in - Returns True if value doesn't exist

Examples:

print('a' in 'apple')

print('z' in 'apple')

fruits = ['apple', 'banana', 'mango']

print('banana' in fruits)

print('grape' not in fruits)

4. Difference Between IN and NOT IN

in checks for presence, while not in checks for absence.

Practical Example:

allowed_users = ['admin', 'editor', 'viewer'] user = 'guest'

if user in allowed_users:

print("Access granted")

else:

print("Access denied")

print('admin' not in allowed_users)

5. Comparison Operators (== vs !=)

== checks equality, != checks inequality.

Examples:

print(5 == 5.0)

print(5 != '5')

name1 = "Alice"

name2 = "alice"

print(name1 == name2)

print(name1.lower() == name2.lower())

6. Conditional Statements

Definition: Control structures that execute different code blocks based on conditions.

Syntax:

if condition1:

elif condition2:

else:

Practical Example (Age Verification):

age = 20

if age < 13:

print("Child")

elif age < 20:

print("Teenager")

else:

print("Adult")

7. if-else Example (Pass/Fail System)

marks = 78

if marks >= 40:

print("Congratulations! You passed.")

if marks > 90:

print("With distinction!") else:

print("Sorry, you failed. Try again next time.")

8. if-elif-else Ladder (Grading System)

score = 87

if score >= 90:

grade = 'A'

elif score >= 80:

grade = 'B'

elif score >= 70:

grade = 'C'

elif score >= 60:

```
grade = 'D'

else:

grade = 'F'

print(f"Your grade is {grade}")
```

9. Nested Conditions (Voting Eligibility)

```
age = 22

is_citizen = True

has_voter_id = False

if age >= 18:
        if is_citizen:
                if has_voter_id:
                        print("You can vote!")
                else:
                        print("Please get your voter ID first")
        else:
                print("Citizenship required")
else:
        print("You're too young to vote")
```

10. Indentation in Python

Definition: The spaces/tabs at the beginning of code lines that define code blocks (instead of curly braces {} like

other languages).

Correct:

```
if True:
        print("This is properly indented")
                print("This is part of the same block") Incorrect (Error):

python

if True:
```

print("This will cause IndentationError")

11. Common Python Errors

1. **SyntaxError** - Invalid syntax

python

if True print("Hello") # Missing colon

2. **NameError** - Undefined variable

python

print(undefined_var) # Variable not declared

3. **TypeError** - Wrong operation on type

python

"5" + 3 # Can't add string and integer

4. **IndexError** - Invalid list index

python

lst = [1,2,3]

print(lst[5]) # Only has 3 elements

5. **KeyError** - Missing dictionary key

python

person = {'name': 'John'}

print(person['age']) # No 'age' key

12. Examples of Common Python Errors

SyntaxError Example:

Occurs when Python code structure is invalid.

Missing parentheses in print function (Python 3+)

if True

print("Hi there!")

NameError Example:

Trying to use a variable that hasn’t been declared.

'number' is not defined anywhere

print(number)

KeyError Example:

Accessing a dictionary key that doesn't exist.

student = {"id": 101, "name": "Ravi"}

print(student["grade"]) # 'grade' key is not in the dictionary

13. What is a Loop and How Many Types Are There in Python? A loop is a control structure that allows repeating a block of code multiple times based on a condition or sequence.

Python supports two main types of loops:

1. for loop – Used to iterate through items in a sequence like a string, list, or dictionary.

2. while loop – Executes a block of code as long as a given condition remains True.

14. for Loop Example Using a List (Different Items)

```
cities = ["Delhi", "Mumbai", "Chennai", "Kolkata"]

for city in cities:
    print("City:", city)
```

15. for Loop Examples Using Different Data Types

Using a String:

```
greeting = "Hello"

for char in greeting:
    print("Letter:", char)
```

Using a Dictionary:

```
book = {"title": "Python Basics", "pages": 250}

for key in book:
    print(f"{key} => {book[key]}")
```

Using a Tuple:

```
numbers = (10, 20, 30, 40)

for num in numbers:
    print("Value:", num)
```