PYTHON INTERVIEW GUIDE

Python Interview-এর 100 Core Concepts

Python, OOP ও Problem Solving—সহজ বাংলায়

লেখক

Md. Aminul Haque Palash

সূচিপত্র

ভূমিকা 2
Part I — Python Core Fundamentals 3
1. Python কীভাবে Code Execute করে 3
2. Dynamic vs Static Typing 3
3. Variables, Objects এবং References 3
4. Mutable vs Immutable Objects 4
5. Identity vs Equality—`is` vs `==` 4
6. Python Built-in Data Types 4
7. List vs Tuple 5
8. Set এবং Frozenset 5
9. Dictionary, Keys এবং Hashing 5
10. Shallow Copy vs Deep Copy 5
Part II — Syntax, Scope & Data Handling 7
11. Indexing এবং Slicing 7
12. Packing এবং Unpacking 7
13. Truthy এবং Falsy Values 7
14. Conditional Expression 8
15. Loop এবং Loop Control 8
16. Scope এবং LEGB Rule 8
17. `global` এবং `nonlocal` 9
18. Comprehensions 9
19. Sorting এবং Custom Keys 9
20. Type Hints 9
Final Revision Framework 10
শেষ কথা 11

ভূমিকা

Python interview-এ শুধু syntax জানা যথেষ্ট নয়। Interviewer দেখতে চান—আপনি Python-এর object model বোঝেন কি না, সঠিক data structure বেছে নিতে পারেন কি না, OOP design ব্যাখ্যা করতে পারেন কি না এবং brute-force solution-কে কীভাবে improve করবেন তা জানেন কি না।
এই বইয়ের প্রতিটি concept চারটি প্রশ্নের উত্তর দেবে: concept-টি কী, Python-এ কীভাবে কাজ করে, বাস্তবে কোথায় লাগে এবং interview-এ কীভাবে সংক্ষেপে explain করবেন। লক্ষ্য মুখস্থ করা নয়; নিজের ভাষায় পরিষ্কারভাবে বোঝাতে শেখা।

Part I — Python Core Fundamentals

1. Python কীভাবে Code Execute করে

Python source code সাধারণত প্রথমে bytecode-এ compile হয়, এরপর Python Virtual Machine সেই bytecode execute করে। তাই Python-কে শুধু "interpreted" বলা convenient হলেও পুরো execution modelটি compile এবং interpret—দুটোর সমন্বয়।

Python Example:

```
import dis
dis.dis(lambda x: x + 1)
```

Common Trap: "Python কখনো compile হয় না"—এটি সঠিক নয়।

> **Interview-ready Answer**
> Python source থেকে bytecode তৈরি করে এবং virtual machine-এ execute করে; implementation অনুযায়ী details বদলাতে পারে।

2. Dynamic vs Static Typing

Python dynamically typed: variable-এর fixed type নেই; runtime-এ object-এর type থাকে। Static language-এ variable-এর type সাধারণত compile time-এ check হয়।

Python Example:

```
x = 10
x = "ten"
```

Common Trap: Dynamic typing মানে weak typing নয়; Python incompatible types স্বয়ংক্রিয়ভাবে সবসময় convert করে না।

> **Interview-ready Answer**
> Python-এ names runtime objects-কে reference করে, তাই একই name ভিন্ন সময়ে ভিন্ন type-এর object point করতে পারে।

3. Variables, Objects এবং References

Python variable আসলে একটি name, যা object-কে reference করে। Assignment সাধারণত object copy করে না; আরেকটি name একই object-কে point করতে পারে।

Python Example:

```
a = [1, 2]
b = a
b.append(3)
print(a)  # [1, 2, 3]
```

Interview-ready Answer
Python assignment name-কে object-এর সঙ্গে bind করে; mutable object share করলে এক reference-এর change অন্য reference থেকেও দেখা যায়।

4. Mutable vs Immutable Objects

Mutable object তৈরি হওয়ার পর পরিবর্তন করা যায়—যেমন list, dict, set। Immutable object পরিবর্তন করা যায় না—যেমন int, str, tuple; "পরিবর্তন" করলে নতুন object তৈরি হয়।

Python Example:

```
items = [1, 2]
items.append(3)      # same object changes
name = "Py"
name += "thon"      # new string object
```

Interview-ready Answer
Mutability বলে object in-place বদলানো যাবে কি না; এটি sharing, copying এবং function arguments বোঝার জন্য গুরুত্বপূর্ণ।

5. Identity vs Equality—`is` vs `==`

`==` value equality check করে, আর `is` দুই reference একই object কি না check করে। `None` comparison-এ `is None` ব্যবহার করা উচিত।

Python Example:

```
a = [1, 2]
b = [1, 2]
print(a == b)  # True
print(a is b)  # False
```

Common Trap: Small integer/string interning-এর কারণে `is` কখনো unexpected True হতে পারে; value comparison-এ এর ওপর নির্ভর করবেন না।

Interview-ready Answer
`==` content বা logical equality যাচাই করে; `is` object identity যাচাই করে। Value compare করতে সাধারণত `==` ব্যবহার করি।

6. Python Built-in Data Types

Common built-ins হলো numeric types, `str`, `list`, `tuple`, `range`, `dict`, `set`, `frozenset`, `bool`, `bytes` এবং `NoneType`। Type নির্বাচন data-এর nature ও operation অনুযায়ী করা উচিত।

Python Example:

```
count = 3
tags = {"python", "oop"}
user = {"id": 7, "active": True}
```

Interview-ready Answer
Built-in type বেছে নেওয়ার সময় ordering, mutability, uniqueness, lookup এবং memory requirement বিবেচনা করি।

7. List vs Tuple

List mutable, tuple immutable। List dynamic collection-এর জন্য; tuple fixed record বা immutable group-এর জন্য useful। Hashable elements থাকলে tuple dictionary key হতে পারে।

Real-world Use: Coordinates `(lat, lon)` tuple হতে পারে, কিন্তু shopping cart items list হওয়া natural।

Interview-ready Answer
List পরিবর্তনযোগ্য sequence, tuple অপরিবর্তনীয় sequence; intent ও mutability requirement অনুযায়ী নির্বাচন করি।

8. Set এবং Frozenset

Set unique hashable values রাখে এবং fast membership test দেয়। `frozenset` immutable, তাই dictionary key বা অন্য set-এর element হতে পারে।

Python Example:

```python
seen = {1, 2, 2}
print(seen)          # {1, 2}
permissions = frozenset({"read", "write"})
```

Interview-ready Answer
Set uniqueness ও average O(1) membership-এর জন্য; frozenset একই ধারণার immutable ও hashable form।

9. Dictionary, Keys এবং Hashing

Dictionary key-এর hash ব্যবহার করে value locate করে। Key hashable হতে হবে এবং equality/hash contract consistent হতে হবে। Modern Python dict insertion order preserve করে।

Python Example:

```python
users = {101: "Rahim", 102: "Karim"}
print(users.get(103, "Unknown"))
```

Interview-ready Answer
Dictionary hash table-ভিত্তিক mapping; average lookup O(1), তবে key immutable/hashable এবং consistent equality semantics থাকা দরকার।

10. Shallow Copy vs Deep Copy

Shallow copy outer container copy করে কিন্তু nested objects share করে। Deep copy recursively nested objects-ও copy করার চেষ্টা করে।

Python Example:

```python
import copy
a = [[1], [2]]
b = copy.copy(a)
c = copy.deepcopy(a)
```

Common Trap: Deep copy সবসময় দরকার নেই; এটি expensive হতে পারে এবং shared-resource object-এর ক্ষেত্রে inappropriate হতে পারে।

Interview-ready Answer

Shallow copy নতুন outer object বানায় কিন্তু child references share করে; deep copy nested object graph-ও duplicate করে।

Part II — Syntax, Scope & Data Handling

11. Indexing এবং Slicing

Index একটি element নেয়; slicing `[start:stop:step]` নতুন sequence তৈরি করে। Negative index end থেকে count করে।

Python Example:

```
data = [0, 1, 2, 3, 4]
print(data[1:4])   # [1, 2, 3]
print(data[::-1]) # reverse
```

Interview-ready Answer

Slicing sequence-এর selected range নেয়; stop exclusive এবং omitted values-এর sensible defaults থাকে।

12. Packing এবং Unpacking

Multiple values tuple হিসেবে pack করা যায় এবং pattern অনুযায়ী unpack করা যায়। Extended unpacking-এ `*` remaining items collect করে।

Python Example:

```
first, *middle, last = [1, 2, 3, 4]
a, b = b, a
```

Interview-ready Answer

Packing multiple values-কে collection বানায়; unpacking structure থেকে values আলাদা names-এ bind করে।

13. Truthy এবং Falsy Values

`False`, `None`, numeric zero এবং empty collections falsy; অন্য অধিকাংশ object truthy। Custom class `__bool__` বা `__len__` দিয়ে truth value নির্ধারণ করতে পারে।

Python Example:

```
items = []
if not items:
    print("No items")
```

Common Trap: `if not value` zero, empty এবং None-কে একসঙ্গে ধরে; distinction দরকার হলে `is None` ব্যবহার করুন।

Interview-ready Answer
Python condition explicit boolean ছাড়াও object-এর truth value evaluate করে; empty collection ও zero সাধারণত falsy।

14. Conditional Expression

Python-এর ternary form হলো `x if condition else y`। ছোট value selection-এ readable, কিন্তু complex nested expression readability কমায়।

Python Example:

```
status = "adult" if age >= 18 else "minor"
```

Interview-ready Answer
Conditional expression একটি condition অনুযায়ী দুই value-এর একটি return করে; simple case-এ ব্যবহার করি।

15. Loop এবং Loop Control

`for` iterable consume করে; `while` condition true থাকা পর্যন্ত চলে। `break` loop থামায়, `continue` current iteration skip করে, loop-এর `else` normal completion-এ চলে।

Python Example:

```
for n in numbers:
    if n < 0:
        break
else:
    print("No negative number")
```

Interview-ready Answer
`for` iteration-driven এবং `while` condition-driven; break/continue flow control করে, আর loop-else break না হলে execute হয়।

16. Scope এবং LEGB Rule

Name resolution order: Local, Enclosing, Global, Built-in। Function-এর ভিতরে assignment করলে name defaultভাবে local হয়।

Python Example:

```
x = "global"
def outer():
    x = "enclosing"
    def inner():
        print(x)
```

Interview-ready Answer
Python name lookup LEGB order অনুসরণ করে—Local, Enclosing, Global, তারপর Built-in scope।

17. `global` এবং `nonlocal`

`global` module-level binding modify করে; `nonlocal` nearest enclosing function scope-এর binding modify করে। অতিরিক্ত ব্যবহার hidden state তৈরি করে।

Common Trap: Cleaner design-এর জন্য return value বা object state অনেক সময় ভালো।

Interview-ready Answer

`global` global scope এবং `nonlocal` enclosing function scope-এর existing name rebind করতে ব্যবহৃত হয়।

18. Comprehensions

List, set ও dict comprehensions transformation এবং filtering compactভাবে প্রকাশ করে। Simple হলে Pythonic; nested logic হলে normal loop বেশি readable।

Python Example:

```
squares = [x * x for x in range(10) if x % 2 == 0]
lookup = {u.id: u.name for u in users}
```

Interview-ready Answer

Comprehension iterable থেকে নতুন collection declaratively তৈরি করে, optional filteringসহ।

19. Sorting এবং Custom Keys

`sorted()` নতুন list দেয়; `list.sort()` in-place sort করে। `key` function প্রতিটি item-এর comparison value নির্ধারণ করে। Python sort stable।

Python Example:

```
users.sort(key=lambda u: (u.age, u.name))
latest = sorted(orders, key=lambda o: o.created_at, reverse=True)
```

Interview-ready Answer

`sorted` নতুন result দেয়, `sort` list modify করে; custom ordering-এর জন্য key function ব্যবহার করি।

20. Type Hints

Type hints documentation ও static analysis improve করে, কিন্তু defaultভাবে runtime enforcement করে না। `list[str]`, union এবং protocol expressive API তৈরি করে।

Python Example:

```
def total(prices: list[float]) -> float:
    return sum(prices)
```

Interview-ready Answer

Type hints developer ও type checker-কে expected types জানায়; Python runtime সাধারণত এগুলো enforce করে না।

Final Revision Framework

প্রতিটি concept revise করার সময় পাঁচটি প্রশ্ন করুন:

1. এক বাক্যে definition দিতে পারি?
2. ছোট Python example লিখতে পারি?
3. কোন real-world problem-এ লাগে জানি?
4. common mistake বা trade-off বলতে পারি?
5. brute-force solution-এর complexity explain করে better approach দিতে পারি?

> **Interview-ready Answer**
> Interview-এ প্রথমে direct answer দিন, তারপর ছোট example এবং শেষে limitation বা trade-off বলুন। অপ্রয়োজনীয় theory দিয়ে মূল উত্তর ঢেকে ফেলবেন না।

শেষ কথা

Python interview preparation মানে শত syntax মুখস্থ করা নয়। মূল লক্ষ্য হলো Python-এর object/reference model বোঝা, readable code লেখা, সঠিক abstraction ও data structure নির্বাচন করা এবং নিজের সিদ্ধান্তের trade-off পরিষ্কারভাবে ব্যাখ্যা করা।
একটি concept সত্যিই শেখা হয়েছে তখনই, যখন আপনি code না দেখেও সেটি সহজ ভাষায় explain করতে পারেন এবং ছোট একটি example লিখতে পারেন।

ABOUT THE AUTHOR

লেখক পরিচিতি

Md. Aminul Haque Palash

Software Engineer • Machine Learning Practitioner

Md. Aminul Haque Palash একজন Software Engineer এবং Machine Learning practitioner। তিনি Chittagong University of Engineering and Technology (CUET) থেকে Computer Science and Engineering (CSE) বিষয়ে B.Sc. ডিগ্রি অর্জন করেছেন এবং Software Engineering ও AI/ML নিয়ে কাজ করে আসছেন।

তাঁর অভিজ্ঞতায়, একজন engineer-এর জন্য শুধু Python syntax জানা যথেষ্ট নয়; Python-এর core concepts, Object-Oriented Programming, data structures এবং problem-solving-এর উপর পরিষ্কার ধারণা থাকা জরুরি—বিশেষ করে technical interview এবং বাস্তব software development-এর জন্য।

বাংলাভাষী engineers ও শিক্ষার্থীদের জন্য এসব বিষয় সহজ, structured এবং interview-oriented উপায়ে তুলে ধরার লক্ষ্যেই এই বইটি লেখা। এখানে concept মুখস্থ করার পরিবর্তে **কেন প্রয়োজন, কীভাবে কাজ করে এবং interview ও বাস্তব কাজে কীভাবে প্রয়োগ করা যায়**—সেদিকে গুরুত্ব দেওয়া হয়েছে।