Python Hashing και Αμετάβλητα Δεδομένα (Python Hashing and Immutable Data)

1. Τι είναι το Hashing; (What is Hashing?)

Το hashing είναι η διαδικασία μετατροπής δεδομένων οποιουδήποτε μεγέθους σε μια σταθερού μεγέθους τιμή (hash value). (Hashing is the process of converting data of any size into a fixed-size value (hash value).)

Βασικά Χαρακτηριστικά (Basic Characteristics):

- Αντιστρεψιμότητα: Μη αντιστρέψιμη διαδικασία (Reversibility: Non-reversible process)
- Ντετερμινισμός: Ίδια είσοδος = Ίδια έξοδος (Determinism: Same input = Same output)
- Κατανομή: Ομοιόμορφη κατανομή των τιμών (Distribution: Uniform distribution of values)

2. Η Συνάρτηση hash() στην Python

```
# Παραδείγματα χρήσης της hash()
# Examples of using hash()

# Για αριθμούς (For numbers)
print(hash(42))  # Επιστρέφει το ίδιο hash κάθε φορά
print(hash(3.14))

# Για strings
print(hash("Hello"))

# Για tuples με immutable στοιχεία
print(hash((1, 2, "three")))
```

3. Γιατί Μόνο Αμετάβλητα (Immutable) Δεδομένα; (Why Only Immutable Data?)

Θεωρητική Εξήγηση (Theoretical Explanation)

Μόνο αμετάβλητα δεδομένα επιτρέπονται ως κλειδιά σε λεξικά και σαν στοιχεία στα σύνολα επειδή: (Only immutable data are valid as keys in dictionaries and sets because:)

1. **Συνέπεια Hash** (Hash Consistency)
    - Η τιμή hash πρέπει να παραμένει σταθερή (κάτι σαν υπογραφή)
    - (Hash value must remain constant)
    - Αν άλλαζε το αντικείμενο, θα άλλαζε και το hash
    - (If the object changed, the hash would change too)
2. **Αξιοπιστία Αναζήτησης** (Search Reliability)
    - Η αναζήτηση βασίζεται στην τιμή hash
    - (Searching is based on hash value)
    - Μεταβαλλόμενα κλειδιά θα "χάνονταν" στη δομή
    - (Mutable keys would get "lost" in the structure)

Παράδειγμα Προβλήματος με Μεταβλητά Δεδομένα (Example Problem with Mutable Data)

```
# Αυτό θα προκαλούσε πρόβλημα αν επιτρεπόταν
# This would cause problems if allowed
mutable_key = [1, 2, 3]
dictionary = {}

# Υποθετικό σενάριο (Hypothetical scenario)
dictionary[mutable_key] = "value"
mutable_key.append(4)  # Αλλάζει το hash! (Changes the hash!)
# Το κλειδί δεν θα μπορούσε πλέον να βρεθεί
# The key could no longer be found
```

In [ ]:

Python Sets (Σύνολα)

Εισαγωγή

Τα sets (σύνολα) είναι μια συλλογή από μοναδικά στοιχεία στην Python. Λειτουργούν όπως τα μαθηματικά σύνολα - κάθε στοιχείο εμφανίζεται μόνο μία φορά και μπορούμε να κάνουμε πράξεις όπως ένωση και τομή.

Βασικά Χαρακτηριστικά

- Τα sets δημιουργούνται με άγκιστρα `{}` ή με τη συνάρτηση `set()`
- Τα στοιχεία είναι μοναδικά (unique)
- Τα στοιχεία πρέπει να είναι αμετάβλητα (immutable)
- Τα sets δεν έχουν συγκεκριμένη σειρά (unordered)

Δημιουργία Set (Creating a Set)

```
# Empty set (must use set() constructor)
empty_set = set()

# Set with elements
numbers = {1, 2, 3, 4, 5}

# Creating set from list
fruits = set(["apple", "banana", "orange"])

# Duplicates are automatically removed
numbers = {1, 2, 2, 3, 3, 4}  # Result: {1, 2, 3, 4}
```

Βασικές Λειτουργίες (Basic Operations)

1. Προσθήκη Στοιχείων (Adding Elements)

```
# Add single element
fruits.add("grape")
```

```
# Add multiple elements
fruits.update(["mango", "pear"])
```

2. Αφαίρεση Στοιχείων (Removing Elements)

```
# Remove element (raises error if not found)
fruits.remove("apple")

# Discard element (no error if not found)
fruits.discard("melon")

# Remove and return arbitrary element
item = fruits.pop()

# Remove all elements
fruits.clear()
```

Πράξεις Συνόλων (Set Operations)

```
A = {1, 2, 3, 4}
B = {3, 4, 5, 6}

# Ένωση (Union)
union = A | B  # or A.union(B)
# Result: {1, 2, 3, 4, 5, 6}

# Τομή (Intersection)
intersection = A & B  # or A.intersection(B)
# Result: {3, 4}

# Διαφορά (Difference)
difference = A - B  # or A.difference(B)
# Result: {1, 2}

# Συμμετρική Διαφορά (Symmetric Difference)
sym_diff = A ^ B  # or A.symmetric_difference(B)
# Result: {1, 2, 5, 6}
```

Έλεγχοι σε Sets (Set Checks)

```
# Check if element exists
if "apple" in fruits:
    print("Found apple!")

# Check if one set is subset of another
A = {1, 2}
B = {1, 2, 3, 4}
is_subset = A <= B  # True

# Check if one set is superset of another
is_superset = B >= A  # True

# Check if sets are disjoint (no common elements)
C = {5, 6}
are_disjoint = A.isdisjoint(C)  # True
```

Παράδειγμα Χρήσης (Usage Example)

```python
# Student courses registration system
math_students = {"John", "Maria", "George"}
physics_students = {"Maria", "Helen", "Peter"}
chemistry_students = {"George", "Peter", "Anna"}

# Students taking both math and physics
math_and_physics = math_students & physics_students
print("Students in both Math and Physics:", math_and_physics)

# Students taking any science course
all_science_students = math_students | physics_students | chemistry_students
print("Total unique students:", all_science_students)

# Students taking only math (not physics or chemistry)
only_math = math_students - (physics_students | chemistry_students)
print("Students taking only Math:", only_math)
```

Πρακτικές Χρήσεις (Practical Uses)

1. Αφαίρεση διπλότυπων από λίστα (Removing duplicates from a list)

```python
numbers_list = [1, 2, 2, 3, 3, 3, 4]
unique_numbers = list(set(numbers_list))
```

2. Εύρεση κοινών στοιχείων (Finding common elements)

```python
list1 = [1, 2, 3, 4]
list2 = [3, 4, 5, 6]
common_elements = set(list1) & set(list2)
```

Συμβουλές (Tips)

1. Χρησιμοποιήστε sets όταν χρειάζεστε μοναδικά στοιχεία (Use sets when you need unique elements)
2. Τα sets είναι πιο αποδοτικά από τις λίστες για έλεγχο ύπαρξης στοιχείων (Sets are more efficient than lists for membership testing)
3. Χρησιμοποιήστε sets για μαθηματικές πράξεις συνόλων (Use sets for mathematical set operations)
4. Θυμηθείτε ότι τα sets δεν διατηρούν σειρά (Remember that sets don't maintain order)
5. Μόνο αμετάβλητα (immutable) στοιχεία μπορούν να μπουν σε set (Only immutable elements can be added to a set)

In [ ]:

Μεταβλητές Περιβάλλοντος στην Python (Environment Variables in Python)

Εισαγωγή (Introduction)

Οι μεταβλητές περιβάλλοντος είναι δυναμικές τιμές που μπορούν να επηρεάσουν τη συμπεριφορά των εκτελούμενων διεργασιών σε ένα λειτουργικό σύστημα. (Environment variables are dynamic values that can affect the behavior of running processes in an operating system.)

Το PATH στα Windows (PATH in Windows)

Το PATH είναι μια ειδική μεταβλητή περιβάλλοντος που καθορίζει τις διαδρομές όπου το λειτουργικό σύστημα αναζητά εκτελέσιμα προγράμματα. (PATH is a special environment variable that defines where the operating system looks for executable programs.)

Πώς να δείτε το PATH (How to view PATH):

```
cmd
echo %PATH%
```

Πώς να προσθέσετε στο PATH στα Windows (How to add to PATH in Windows):

1. Πίνακας Ελέγχου → Σύστημα → Ρυθμίσεις συστήματος για προχωρημένους → Μεταβλητές Περιβάλλοντος (Control Panel → System → Advanced system settings → Environment Variables)
2. Βρείτε το "Path" στις μεταβλητές συστήματος (Find "Path" in system variables)
3. Επεξεργασία → Νέο → Προσθέστε τη διαδρομή (Edit → New → Add your path)

Χρήση Μεταβλητών Περιβάλλοντος στην Python (Using Environment Variables in Python)

Πρόσβαση σε Μεταβλητές Περιβάλλοντος (Accessing Environment Variables)

```python
import os

# Λήψη τιμής μεταβλητής περιβάλλοντος (Get environment variable)
path = os.environ.get('PATH')
print(path)

# Λίστα όλων των μεταβλητών περιβάλλοντος (List all environment variables)
all_env = os.environ
for key, value in all_env.items():
    print(f"{key}: {value}")

# Έλεγχος αν υπάρχει μεταβλητή (Check if variable exists)
if 'MY_VAR' in os.environ:
    print("Η μεταβλητή υπάρχει! / Variable exists!")
```

Ορισμός Μεταβλητών Περιβάλλοντος (Setting Environment Variables)

Μέσω Python (Through Python):

```python
import os

# Ορισμός νέας μεταβλητής (Set new variable)
os.environ['MY_VAR'] = 'my_value'

# Προσωρινή αλλαγή PATH (Temporary PATH modification)
os.environ['PATH'] = os.environ['PATH'] + ';C:\\my_new_path'
```

Μέσω Command Line στα Windows (Through Windows Command Line):

```
cmd
set MY_VAR=my_value
```

Μέσω PowerShell:

```
$env:MY_VAR = "my_value"
```

Εκτέλεση Python Script με Μεταβλητές Περιβάλλοντος (Running Python Script with Environment Variables)

Windows CMD:

```
cmd
set ENV_VAR=1234
python script.py
```

Windows PowerShell:

```
$env:ENV_VAR = "1234"; python script.py
```

Μονογραμμική Εντολή (One-liner):

```
cmd
set ENV_VAR=1234 && python script.py
```

Παράδειγμα Script (Example Script)

```python
# test.py
import os

# Λήψη μεταβλητής περιβάλλοντος με προεπιλεγμένη τιμή
# (Get environment variable with default value)
env_value = os.environ.get('ENV_VAR', 'default_value')
print(f"Η τιμή της ENV_VAR είναι: {env_value}")
print(f"The value of ENV_VAR is: {env_value}")

# Παράδειγμα χρήσης σε πραγματική εφαρμογή
# (Example usage in real application)
debug_mode = os.environ.get('DEBUG', 'False').lower() == 'true'
if debug_mode:
    print("Εκτέλεση σε debug mode / Running in debug mode")
else:
    print("Εκτέλεση σε κανονική λειτουργία / Running in normal mode")
```

Συνήθεις Χρήσεις (Common Uses)

1. **Ρυθμίσεις Ασφαλείας (Security Settings)**
   - Κλειδιά API (API keys)
   - Κωδικοί πρόσβασης (Passwords)
   - Μυστικά tokens (Secret tokens)
2. **Ρυθμίσεις Εφαρμογής (Application Settings)**
   - Περιβάλλον εκτέλεσης (Runtime environment)
   - Παραμετροποίηση (Configuration)
   - Διαδρομές αρχείων (File paths)

Καλές Πρακτικές (Best Practices)

1. **Ασφάλεια (Security)**
   - Μην αποθηκεύετε ευαίσθητες πληροφορίες στον κώδικα
   - (Don't store sensitive information in code)
   - Χρησιμοποιείτε αρχεία .env για τοπική ανάπτυξη
   - (Use .env files for local development)
2. **Διαχείριση (Management)**
   - Διατηρείτε τεκμηρίωση για όλες τις μεταβλητές
   - (Maintain documentation for all variables)
   - Χρησιμοποιείτε προεπιλεγμένες τιμές όπου είναι δυνατόν
   - (Use default values where possible)

In [ ]: