Εισαγωγή στον Προγραμματισμό Η/Υ

Διδάσκοντες: Ι. Πολενάκης, Κ. Σκιαδόπουλος, Δ. Μουρατίδου

Τμήμα Πληροφορικής, Ιόνιο Πανεπιστήμιο

6. Συλλογές στην Python:

- **Συλλογές στην Python**

… θα δούμε αναλυτικά τις τέσσερις βασικές δομές δεδομένων της Python:

- **list (λίστα)**
- **tuple (πλειάδα)**
- **set (σύνολο)**
- **dictionary (λεξικό)**

6. Συλλογές στην Python:

Συλλογές στην Python

Λίστα - list	Πλειάδα - tuple	Σύνολο - set	Λεξικό - dictionary
Δομή αποθήκευσης ομοιογενών αλλά και ετερογενών δεδομένων			
Επιτρέπει μια ή περισσότερες γραμμές και μια ή περισσότερες στήλες		Επιτρέπει μόνο μια γραμμή	Ζευγάρια κλειδιού - τιμής
Αναπαράσταση με []	Αναπαράσταση με ()	Αναπαράσταση με {}	Αναπαράσταση με {}
Επιτρέπονται διπλότυπα στοιχεία	Επιτρέπονται διπλότυπα στοιχεία	Δεν επιτρέπονται διπλότυπα στοιχεία	Δεν επιτρέπονται διπλότυπα κλειδιά
Π.χ. [1, 2, 3, 4, 5]	Π.χ. (1, 2, 3, 4, 5)	Π.χ. {1, 2, 3, 4, 5}	Π.χ. {'el' : 'Greece', 'fr' : 'France'}
Δημιουργία με τη list()	Δημιουργία με τη tuple()	Περιέχουν μόνο αμετάβλητα στοιχεία: string, int, float & () με μόνο αμετάβλητα στοιχεία	Κλειδιά: (α) μόνο αμετάβλητα στοιχεία, λ.χ: string, int, float, & (), (β) μοναδικά
Υποστηρίζουν δεικτοδότηση & τεμαχισμό με []		Δεν υποστηρίζουν δεικτοδότηση & τεμαχισμό με []	Δεν υποστηρίζουν τεμαχισμό με []
Επιτρέπονται αλλαγές σε λίστα και στοιχεία της	Δεν επιτρέπονται αλλαγές στην πλειάδα ή τα στοιχεία της	Επιτρέπονται αλλαγές στο σύνολο και τα στοιχεία του	Επιτρέπονται αλλαγές στο λεξικό και τα στοιχεία του
Διατεταγμένο	Διατεταγμένο	Μη διατεταγμένο	Μη Διατεταγμένο
Κενή λίστα l=[]	Κενή πλειάδα t=()	Κενό σύνολο s=set()	Κενό λεξικό d={}

6. Συλλογές στην Python:

- **List (Λίστα)**

Τι είναι

Η λίστα είναι μια μεταβλητή, σειριακή, ταξινομημένη συλλογή στοιχείων. Μπορεί να αλλάξει μετά τη δημιουργία της.

Χαρακτηριστικά

- Μεταβάλλεται (mutable)
- Διατηρεί σειρά
- Επιτρέπει διπλότυπα στοιχεία
- Υποστηρίζει προσθήκη, αφαίρεση, αλλαγή

6. Συλλογές στην Python:

List (Λίστα)

- **Δημιουργία λίστας**

```
a = [1, 2, 3]
b = ["μήλο", "πορτοκάλι", "μπανάνα"]
c = [1, "text", True, 3.14]
```

- **Πρόσβαση στοιχείων**

```
fruits = ["μήλο", "πορτοκάλι", "αχλάδι"]
print(fruits[0]) # μήλο
print(fruits[-1]) # αχλάδι
```

- **Αλλαγή στοιχείων**

```
nums = [10, 20, 30]
nums[1] = 200
print(nums) # [10, 200, 30]
```

- **Προσθήκη στοιχείων**

```
nums = [1, 2, 3]
nums.append(4) # Προσθήκη στο τέλος
nums.insert(1, 100) #Προσθήκη στη θέση 1
print(nums)
```

- **Αφαίρεση στοιχείων**

```
nums = [10, 20, 30]
nums.remove(20) # Με βάση την τιμή
nums.pop() # Αφαίρεση τελευταίου
nums.pop(0) # Αφαίρεση με βάση θέση
```

- **Χρήσιμες μέθοδοι λίστας**

```
numbers = [1, 5, 3, 2]
numbers.sort()
numbers.reverse()
print(len(numbers))
```

6. Συλλογές στην Python:

List (Λίστα) - Παραδείγματα

1. Άθροισμα τιμών μιας λίστας : Δώσε το άθροισμα όλων των αριθμών σε μια λίστα.

```
nums = [3, 7, 2, 9]
print(sum(nums))
```

2. Εισαγωγή νέου στοιχείου: Πρόσθεσε τον αριθμό 10 στη λίστα.

```
nums = [1, 2, 3]
nums.append(10)
print(nums)
```

3. Ταξινόμηση λίστας: Ταξινόμησε τη λίστα σε αύξουσα σειρά.

```
nums = [5, 1, 9, 3]
nums.sort()
print(nums)
```

4. Εύρεση μέγιστου στοιχείου: Βρες το μεγαλύτερο στοιχείο της λίστας.

```
nums = [4, 7, 1, 6]
print(max(nums))
```

6. Συλλογές στην Python:

List (Λίστα) - Παραδείγματα

5. Αντιστροφή λίστας: Αντιστροφή της λίστας.

```
nums = [1, 2, 3]
nums.reverse()
print(nums)
```

6. Μετρήστε πόσες φορές εμφανίζεται ένα στοιχείο: Πόσες φορές εμφανίζεται το 2;

```
nums = [1, 2, 2, 3, 2]
print(nums.count(2))
```

7. Αφαίρεση στοιχείου με τιμή: Αφαίρεσε το στοιχείο 3 από τη λίστα.

```
nums = [1, 3, 5]
nums.remove(3)
print(nums)
```

8. Συγχώνευση δύο λιστών: Συνένωσε δύο λίστες σε μία.

```
a = [1, 2]
b = [3, 4]
c = a + b
print(c)
```

6. Συλλογές στην Python:

- **Tuple (Πλειάδα)**

Τι είναι

Μια tuple είναι μια συλλογή παρόμοια με λίστα, αλλά αμετάβλητη.

Χαρακτηριστικά

- Αμετάβλητη (immutable)
- Ταχύτερη από λίστα
- Διατηρεί σειρά
- Επιτρέπει διπλότυπα στοιχεία

6. Συλλογές στην Python:

- **Tuple (Πλειάδα)**

  - **Δημιουργία tuple**

```
t = (1, 2, 3)
t2 = ("a", "b", "c")
t3 = (1, "hello", True)
```

  - **Πρόσβαση στοιχείων**

```
t = (10, 20, 30)
print(t[0])
print(t[-1])
```

  - **Γιατί να χρησιμοποιήσω tuple;**
    - Όταν δεν θέλουμε να αλλάξει μια δομή
    - Κατάλληλη για σταθερές τιμές
    - Χρησιμοποιείται σε επιστροφή πολλών τιμών από συναρτήσεις

Παράδειγμα χρήσης

```
def get_point():
    return (10, 20)


x, y = get_point()
print(x, y)
```

6. Συλλογές στην Python:

Tuple (Πλειάδα) - Παραδείγματα

1. Πρόσβαση σε στοιχείο: Εκτύπωσε το δεύτερο στοιχείο του tuple.

```
t = (10, 20, 30)
print(t[1])
```

2. Μήκος tuple: Βρες πόσα στοιχεία έχει το tuple.

```
t = (1, 2, 3, 4)
print(len(t))
```

3. Μετατροπή tuple σε list: Μετέτρεψε το tuple σε λίστα.

```
t = (1, 2, 3)
lst = list(t)
print(lst)
```

4. Έλεγχος αν στοιχείο υπάρχει: Υπάρχει το 5 στο tuple;

```
t = (3, 5, 7)
print(5 in t)
```

6. Συλλογές στην Python:

Tuple (Πλειάδα) - Παραδείγματα

5. Συγχώνευση δύο tuples: Συνένωσε δύο tuples.

```
a = (1, 2)
b = (3, 4)
c = a + b
print(c)
```

6. Επανάληψη tuple: Δημιούργησε tuple που επαναλαμβάνει 3 φορές.

```
t = (1, 2)
print(t * 3)
```

7. Εύρεση θέσης στοιχείου: Βρες το index του 7.

```
t = (4, 7, 9)
print(t.index(7))
```

8. Καταμέτρηση στοιχείου: Πόσες φορές εμφανίζεται το 2;

```
t = (2, 4, 2, 9)
print(t.count(2))
```

6. Συλλογές στην Python:

- **Set (Σύνολο)**

Τι είναι

Το set είναι μια μη ταξινομημένη συλλογή μοναδικών στοιχείων.

Χαρακτηριστικά

- Δεν διατηρεί σειρά
- Δεν επιτρέπει διπλότυπα
- Πολύ γρήγορο για αναζητήσεις

6. Συλλογές στην Python:

Set (Σύνολο)

- **Δημιουργία set**

```
s = {1, 2, 3}
s2 = set([1, 2, 2, 3, 4])
print(s2) # {1,2,3,4}
```

- **Προσθήκη / αφαίρεση στοιχείων**

```
s = {1, 2, 3}
s.add(4)
s.remove(2)
s.discard(10) # Δεν δημιουργεί σφάλμα αν δεν υπάρχει
```

- **Πράξεις Συνόλων**

```
a = {1, 2, 3}
b = {3, 4, 5}
print(a | b) # Ένωση
print(a & b) # Τομή
print(a - b) # Διαφορά
print(a ^ b) # Συμμετρική διαφορά
```

6. Συλλογές στην Python:

Set (Σύνολο)- Παραδείγματα

1. Αφαίρεση διπλοτύπων από λίστα: Μετέτρεψε τη λίστα σε set για να φύγουν τα διπλότυπα.

```
nums = [1, 2, 2, 3]
s = set(nums)
print(s)
```

2. Προσθήκη στοιχείου: Πρόσθεσε το 10 στο set.

```
s = {1, 2, 3}
s.add(10)
print(s)
```

3. Αφαίρεση στοιχείου: Αφαίρεσε το 2 από το set.

```
s = {1, 2, 3}
s.remove(2)
print(s)
```

4. Ένωση δύο sets: Κάνε union των δύο sets.

```
a = {1, 2}
b = {2, 3}
print(a | b)
```

6. Συλλογές στην Python:

Set (Σύνολο)- Παραδείγματα

5. Τομή sets: Βρες την τομή των sets.

```
a = {1, 2, 3}
b = {2, 4}
print(a & b)
```

6. Διαφορά sets: Βρες ποια στοιχεία έχει το a που δεν έχει το b.

```
a = {1, 2, 3}
b = {2}
print(a - b)
```

7. Συμμετρική διαφορά: Στοιχεία που δεν είναι κοινά.

```
a = {1, 2, 3}
b = {3, 4}
print(a ^ b)
```

8. Έλεγχος υποσυνόλου: Είναι το {1,2} υποσύνολο του {1,2,3}?

```
a = {1, 2}
b = {1, 2, 3}
print(a.issubset(b))
```

6. Συλλογές στην Python:

- **Dictionary (Λεξικό)**

Τι είναι

Το λεξικό είναι μια συλλογή από ζεύγη κλειδιού – τιμής.

Χαρακτηριστικά

- Προσπέλαση με κλειδί (key)
- Κάθε key είναι μοναδικό
- Mutable
- Πολύ γρήγορο σε αναζητήσεις

6. Συλλογές στην Python:

Dictionary (Λεξικό)

- **Δημιουργία dictionary**

```
person = {
    "name": "Nikos",
    "age": 25,
    "city": "Athens"
```

- **Πρόσβαση τιμών**

```
print(person["name"])
print(person.get("age"))
```

- **Προσθήκη / αλλαγή**

```
person["job"] = "Developer"
person["age"] = 26
```

- **Αφαίρεση**

```
person.pop("city")
del person["job"]
```

- **Επανάληψη σε dictionary**

```
for key in person:
    print(key, person[key])

for key, value in person.items():
    print(key, value)
```

6. Συλλογές στην Python:

- **Dictionary (Λεξικό) - Παραδείγματα**

1. Πρόσβαση σε τιμή: Εκτύπωσε την ηλικία.

```
person = {"name": "Maria", "age": 25}
print(person["age"])
```

2. Προσθήκη ζεύγους κλειδί–τιμή: Πρόσθεσε το "city": "Athens".

```
person = {"name": "Nikos"}
person["city"] = "Athens"
print(person)
```

3. Τροποποίηση τιμής: Άλλαξε την ηλικία σε 30.

```
person = {"age": 20}
person["age"] = 30
print(person)
```

4. Διαγραφή κλειδιού: Διέγραψε το κλειδί "name".

```
person = {"name": "Anna", "age": 22}
del person["name"]
print(person)
```

6. Συλλογές στην Python:

Dictionary (Λεξικό) - Παραδείγματα

5. Λίστα με όλα τα κλειδιά: Εκτύπωσε όλα τα keys.

```
d = {"a": 1, "b": 2}
print(d.keys())
```

6. Λίστα με όλες τις τιμές: Εκτύπωσε όλες τις τιμές.

```
d = {"x": 10, "y": 20}
print(d.values())
```

7. Έλεγχος ύπαρξης κλειδιού: Υπάρχει το "age"?

```
person = {"name": "Anna", "age": 22}
print("age" in person)
```

8. Iteration σε dictionary: Εκτύπωσε όλα τα κλειδιά και τιμές.

```
d = {"a": 1, "b": 2}
for k, v in d.items():
    print(k, v)
```

6. Συλλογές στην Python:

- **Σύγκριση των τεσσάρων δομών**

Δομή	Μεταβλητή;	Επιτρέπει διπλότυπα;	Διατηρεί σειρά;	Χρήση
List	✓	✓	✓	Γενική χρήση
Tuple	✗	✓	✓	Σταθερές τιμές
Set	✓	✗	✗	Μοναδικές τιμές, πράξεις συνόλων
Dictionary	✓	Κλειδιά ✗	✓ (Python 3.7+)	Αναζήτηση με κλειδιά

Μπορείς να χρησιμοποιείς κάθε δομή ανάλογα με την ανάγκη:

- list για συλλογές που αλλάζουν,
- tuple για σταθερά δεδομένα,
- set για μοναδικές τιμές και γρήγορο έλεγχο, και
- dictionary για οργανωμένα δεδομένα με κλειδιά.

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

Α. Συνδυασμός List + Tuple

1. Λίστα από tuples: ταξινόμηση βάσει δεύτερου στοιχείου

Δίνεται λίστα με tuples της μορφής (όνομα, ηλικία). Ταξινόμησε τη λίστα με βάση την ηλικία.

```
people = [("Anna", 25), ("Nikos", 19), ("Maria", 30)]
people_sorted = sorted(people, key=lambda x: x[1])
print(people_sorted)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

A. Συνδυασμός List + Tuple

2. Επεξεργασία λίστας με tuples – αύξηση τιμής στοιχείου

Έχεις λίστα από tuples (product, price). Αύξησε όλες τις τιμές κατά 10% και φτιάξε νέα λίστα.

```
items = [("Laptop", 800), ("Mouse", 20), ("Keyboard", 40)]
updated = [(name, price * 1.10) for name, price in items]
print(updated)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

Α. Συνδυασμός List + Tuple

3. Μετατροπή λίστας tuples σε dictionary

Δώσε dictionary όπου key = όνομα, value = ηλικία.

```
people = [("Anna", 25), ("Nikos", 19), ("Maria", 30)]
d = dict(people)
print(d)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

Β. Συνδυασμός List + Set

1. Αφαίρεση διπλότυπων από λίστα λιστών

Δίνεται λίστα αριθμών με διπλότυπα. Χρησιμοποίησε set μέσα σε list comprehension για να κρατήσεις μόνο μοναδικά.

```
nums = [1, 2, 2, 3, 3, 3, 4]
unique = list(set(nums))
print(unique)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

Β. Συνδυασμός List + Set

2. Λίστα sets: ένωση όλων των sets

Δίνεται λίστα από πολλά sets. Βρες το ενωμένο set όλων των στοιχείων.

```
mysets = [{1, 2}, {2, 3, 4}, {4, 5}]
union_all = set().union(*mysets)
print(union_all)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

Β. Συνδυασμός List + Set

3. Σύγκριση δύο λιστών με χρήση sets

Δώσε τα κοινά στοιχεία δύο λιστών χρησιμοποιώντας set.

```
a = [1, 2, 3, 4]
b = [3, 4, 5, 6]
common = list(set(a) & set(b))
print(common)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

C. Συνδυασμός Dictionary + List

1. Ταξινόμηση dictionary με βάση value χρησιμοποιώντας λίστα

Δίνεται dictionary με μαθητές και βαθμούς. Ταξινόμησε τους μαθητές κατά βαθμό.

```
grades = {"Anna": 17, "Nikos": 12, "Maria": 19}
sorted_items = sorted(grades.items(), key=lambda x: x[1], reverse=True)
print(sorted_items)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

C. Συνδυασμός Dictionary + List

2. Λίστα dictionaries: βρες το άτομο με τη μέγιστη ηλικία

Δίνεται λίστα από dictionaries:{"name": ..., "age": ...}Βρες το άτομο με τη μεγαλύτερη ηλικία.

```
people = [
    {"name": "Anna", "age": 25},
    {"name": "Nikos", "age": 19},
    {"name": "Maria", "age": 30}
]

oldest = max(people, key=lambda x: x["age"])
print(oldest)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

C. Συνδυασμός Dictionary + List

3. Μετατροπή dictionary σε λίστα από tuples

Δώσε λίστα της μορφής [(key, value), ...].

```
d = {"a": 1, "b": 2, "c": 3}

tlist = list(d.items())
print(tlist)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

D. Σύνθετες Φωλιασμένες Δομές

1. Λίστα από dictionaries που περιέχουν sets

Δίνεται λίστα χρηστών με σετ από φίλους. Βρες εκείνους που έχουν πάνω από 2 φίλους.

```
users = [
    {"name": "Anna", "friends": {"Nikos", "Maria"}},
    {"name": "Nikos", "friends": {"Anna", "Giorgos", "Eleni"}},
    {"name": "Maria", "friends": set()}
]

result = [u["name"] for u in users if len(u["friends"]) > 2]
print(result)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

D. Σύνθετες Φωλιασμένες Δομές

2. Dictionary που περιέχει lists από tuples

Δίνεται dictionary μαθημάτων → λίστα από tuples (student, grade). Βρες τον καλύτερο βαθμό σε κάθε μάθημα.

```
courses = {
    "Math": [("Anna", 18), ("Nikos", 15)],
    "Physics": [("Anna", 17), ("Maria", 19)]
}

best = {
    course: max(students, key=lambda x: x[1])
    for course, students in courses.items()
}

print(best)
```

6. Συλλογές στην Python:

Ασκήσεις Συνδυασμού Δομών Python

D. Σύνθετες Φωλιασμένες Δομές

3. Λίστα με dictionaries που περιέχουν tuples και sets

Κάθε προϊόν έχει: όνομα; tuple τιμών (παλιά, νέα); set με tags; Βρες όλα τα προϊόντα που έχουν tag "sale" και νέα τιμή < 50.

```
products = [
    {"name": "Mouse", "price": (20, 15), "tags": {"electronics", "sale"}},
    {"name": "Keyboard", "price": (50, 45), "tags": {"electronics"}},
    {"name": "Headphones", "price": (80, 55), "tags": {"sale"}}
]

filtered = [
    p["name"]
    for p in products
    if "sale" in p["tags"] and p["price"][1] < 50
]

print(filtered)
```