Εισαγωγή στον Προγ/μό Υπολογιστών

Διάλεξη 6

Αφαιρέσεις δεδομένων

Περιεχόμενα

1. Δεδομένα
2. Σύνθετα δεδομένα
3. Αφαίρεση δεδομένων

Δεδομένα

Δεδομένα

- ***Δεδομένα (data)***: *τιμές* που αναπαριστούν πληροφορία

- Αριθμητική πληροφορία:

  23°C

  12,5 χμ/ω

  01101101

  $2 + 3j$

  – Ενσωματωμένοι αριθμητικοί τύποι: `int, float, complex`

```
23
12.5
0b1101101
2 + 3j
```

Δεδομένα

- ***Δεδομένα (data)***: *τιμές* που αναπαριστούν πληροφορία

- Κείμενο:

  «*ἄνδρα μοι ἔννεπε, μοῦσα, πολύτροπον, ὃς μάλα πολλὰ…*»

  «MMXVIV»

  «105.37»

  «`from math import sqrt`»

  – Ενσωματωμένος αλφαριθμητικός τύπος: `str`

  `'ἄνδρα μοι ἔννεπε, μοῦσα, πολύτροπον, ὃς μάλα πολλὰ...'`

  `'MMXVIV'`

  `'105.37'`

  `'from math import sqrt'`

Δεδομένα

- ***Δεδομένα (data)***: *τιμές* που αναπαριστούν πληροφορία

- Κ

  - Ισοδύναμα με κωδικοποίηση χαρακτήρων *Unicode Transformation Format (UTF)* :

    `'\u1f04\u03bd\u03b4\u03c1\u03b1\u0020\u03bc\u03bf\u03b9\u0020\u1f14\u03bd\u03bd\u03b5\u03c0\u03b5\u002c\u0020\u03bc\u03bf\u1fe6\u03c3\u03b1\u002c\u0020\u03c0\u03bf\u03bb\u03cd\u03c4\u03c1\u03bf\u03c0\u03bf\u03bd\u002c\u0020\u1f43\u03c2\u0020\u03bc\u03ac\u03bb\u03b1\u0020\u03c0\u03bf\u03bb\u03bb\u1f70\u2026'`

  – Ενσωματωμένος αλφαριθμητικός τύπος: `str`

    `'ἄνδρα μοι ἔννεπε, μοῦσα, πολύτροπον, ὃς μάλα πολλὰ...'`

    `'MMXVIV'`

    `'105.37'`

    `'from math import sqrt'`

Τύποι δεδομένων

- Ο τύπος μιας τιμής εμφανίζεται με τη συνάρτηση `type`

```
>>> type(12)
<class 'int'>

>>> type(3.14)
<class 'float'>

>>> type('hello')
<class 'str'>

>>> type(True)
<class 'bool'>
```

Σύνθετα δεδομένα

Σύνθετα δεδομένα

- Τα περισσότερα πραγματικά δεδομένα είναι *σύνθετα*
  - *Ρητός αριθμός: αριθμητής, παρονομαστής*
  - *Ημερομηνία*: ημερά, μήνας, έτος
  - *Τοποθεσία*: γεωγραφικό μήκος, πλάτος
  - *Άτομο*:
    όνομα: Thomas
    επίθετο: Anderson
    προσωνύμιο: Neo
    φύλο: άρρεν
    ημ/νία γέννησης: 3/11/1962
    τόπος γέννησης: Capital City, ΗΠΑ
    επάγγελμα: προγραμματιστής
  - *Εικόνα*: χρώμα, φωτεινότητα κάθε σημείου (pixel)

Αφαίρεση δεδομένων

- ***Αφαίρεση δεδομένων (data abstraction):*** *σύνθετα δεδομένα* αναπαρίστανται ως μια ενιαία τιμή αποκρύπτοντας τις πολλαπλές τιμές από τις οποίες αποτελείται
  - *Η χρήση ενός δεδομένου γίνεται ανεξάρτητη από την υλοποίηση του*

  Κατασκευαστής **(*constructor*):** επιστρέφει σύνθετο δεδομένο ως μια ενιαία τιμή και το αρχικοποιεί

  Συναρτήσεις επιλογής (*selectors*): επιστρέφουν κομμάτια του σύνθετου δεδομένου

- Πχ, αναπαράσταση ρητών αριθμών

  Κατασκευαστής: `x = ritos(a, p)`: επιστρέφει τον *ρητό* «`a` προς `p`»

  Selectors: `arith(x)`: επιστρέφει τον *αριθμητή* του ρητού `x`

  `paron(x)`: επιστρέφει τον *παρονομαστή* του ρητού `x`

Αφαίρεση δεδομένων

- Συναρτήσεις εκτέλεσης υπολογισμών με ρητούς:

  `mul_r(x, y)`: πολλαπλασιασμός ρητών `x, y`

  `add_r(x, y)`: άθροισμα

  `eq_r(x, y)`: έλεγχος ισότητας

  `print_r(x)`: εμφάνιση

Αφαίρεση δεδομένων

```
def mul_r(x, y):
    return ritos(arith(x)*arith(y), paron(x)*paron(y))

def add_r(x, y):
    a = arith(x)*paron(y) + arith(y)*paron(x)
    p = paron(x)*paron(y)
    return ritos(a, p)

def eq_r(x, y):
    return arith(x)*paron(y) == arith(y)*paron(x)

def print_r(x):
    print(arith(x), '/', paron(x))
```

Αφαίρεση δεδομένων

- Οι συναρτήσεις `mul_r, add_r, eq_r, print_r` δεν γνωρίζουν τον τρόπο αναπαράστασης των ρητών
  - Χρησιμοποιούν μόνο τον κατασκευαστή και επιλογείς
- Ο κατασκευαστής και οι επιλογείς είναι αρμόδιοι για την αναπαράσταση

`mul_r, add_r, eq_r, print_r`

Αναπαράσταση ρητών ως ενιαία τιμή

---------- `ritos arith paron` ----------

Θα αναπαραστήσουμε ζεύγη τιμών `a, p` με ***λίστες*** `[a, p]`

Παράδειγμα: ζεύγη αριθμών

- Βασικός τύπος `list` (λίστα) της Python

```
>>> [1, 2]
[1, 2]
>>> type([1, 2])
<class 'list'>
>>> x = [1, 2]
>>> x
[1, 2]
>>> a, p = [1, 2]
>>> print(a, '/', p)
1 / 2
```

Παράδειγμα: ζεύγη αριθμών

- Επιλογή μεμονωμένου στοιχείου ενός ζεύγους με βάση τη θέση του (σύμβαση: πρώτη θεση = `0`, δεύτερη = `1`, κτλ.)

```
>>> x = [3, 9]
>>> x[0]
3
>>> x[1]
9
```

- Ισοδύναμα, η επιλογή μπορεί να γίνει με κλήση στη `getitem`

```
>>> from operator import getitem
>>> getitem(x, 0)
3
>>> getitem(x, 1)
9
```

Αφαίρεση δεδομένων

- ***Αφαίρεση δεδομένων (data abstraction):*** *σύνθετα δεδομένα* αναπαρίστανται ως μια ενιαία τιμή αποκρύπτοντας τις πολλαπλές τιμές από τις οποίες αποτελείται
  - *Η χρήση ενός δεδομένου γίνεται ανεξάρτητη από την υλοποίηση του*

  Κατασκευαστής **(*constructor*):** επιστρέφει σύνθετο δεδομένο ως μια ενιαία τιμή και το αρχικοποιεί

  Συναρτήσεις επιλογής (*selectors*): επιστρέφουν κομμάτια του σύνθετου δεδομένου

- Πχ, αναπαράσταση ρητών αριθμών

  Κατασκευαστής: `x = ritos(a, p)`: επιστρέφει τον *ρητό* «`a` προς `p`»

  Selectors: `arith(x)`: επιστρέφει τον *αριθμητή* του ρητού `x`

  `paron(x)`: επιστρέφει τον *παρονομαστή* του ρητού `x`

Αφαίρεση δεδομένων

- Κατασκευή αναπαράστασης ρητών στις `ritos`, `arith`, `paron`

```
def ritos(a, p):
    return [a, p]

def arith(x):
    return x[0]

def paron(x):
    return x[1]
```

Αφαίρεση δεδομένων

```
def mul_r(x, y):
    return ritos(arith(x)*arith(y), paron(x)*paron(y))

def add_r(x, y):
    a = arith(x)*paron(y) + arith(y)*paron(x)
    p = paron(x)*paron(y)
    return ritos(a, p)

def eq_r(x, y):
    return arith(x)*paron(y) == arith(y)*paron(x)

def print_r(x):
    print(arith(x), '/', paron(x))
```

Αφαίρεση δεδομένων

- Οι συναρτήσεις `mul_r, add_r, eq_r, print_r` δεν γνωρίζουν τον τρόπο αναπαράστασης των ρητών
  - Χρησιμοποιούν μόνο τον κατασκευαστή και επιλογείς
- Ο κατασκευαστής και οι επιλογείς είναι αρμόδιοι για την αναπαράσταση

`mul_r, add_r, eq_r, print_r`

Αναπαράσταση ρητών ως ενιαία τιμή

---------- `ritos arith paron` ----------

Αναπαράσταση με ***λίστες*** `[a, p]`

Αφαίρεση δεδομένων

- Αλλαγή της `ritos` (κατασκευαστή) ώστε οι αριθμητές και παρονομαστές να είναι απλοποιημένοι:

```
>>> def ritos(a, p):
        from math import gcd
        g = gcd(a, p)
        return [a // g, p // g]
```

- Δεν απαιτείται αλλαγή καμιάς άλλης συνάρτησης

```
>>> result = add_r(ritos(1, 3), ritos(1, 3))
>>> print_r(result)
2 / 3
```

Αφαίρεση δεδομένων

- Υλοποίηση νέων υπολογισμών; πχ. `square_r`, `inverse_r`
- ➢ Υπολογισμοί με σύνθετα δεδομένα πρέπει να χρησιμοποιούν το σύνορο αναπαράστασης *χωρίς να το παρακάμπτουν*

`mul_r, add_r, eq_r, print_r`

Αναπαράσταση ρητών ως ενιαία τιμή

---------- `ritos arith paron` ----------

Αναπαράσταση με λίστες

Αφαίρεση δεδομένων

```
>>> def inverse_r(x):
        return ritos(x[1], x[0])

>>> def inverse_r(x):
        return [paron(x), arith(x)]

>>> def inverse_r(x):
        return [x[1], x[0]]
```

Παράκαμψη συνόρου αφαίρεσης: χρησιμοποιείται αναπαράσταση χαμηλότερου επιπέδου

- Ορισμός χωρίς παράκαμψη συνόρου:

```
>>> def inverse_r(x):
        return ritos(paron(x), arith(x))
```

Αφαίρεση δεδομένων

- Κάθε τύπος δεδομένων δημιουργεί ένα **σύνορο αναπαραστάσεων ή επίπεδο αφαίρεσης (*abstraction layer*)**

`mul_r, add_r, eq_r, print_r`

Αναπαράσταση ρητών ως ενιαία τιμή

`ritos arith paron`

Αναπαράσταση με λίστες

`[a, p] x[0] x[1]`

Αφαίρεση δεδομένων

- Κάθε τύπος δεδομένων δημιουργεί ένα **σύνορο αναπαραστάσεων ή επίπεδο αφαίρεσης (*abstraction layer*)**

`mul_r, add_r, eq_r, print_r`

Αναπαράσταση ρητών ως ενιαία τιμή

`ritos arith paron`

Αναπαράσταση με ~~λίστες~~ συναρτήσεις

`pair select`

Αναπαράσταση με closures

Αφαίρεση δεδομένων

- Κατασκευή αφηρημένου τύπου *ζευγαριού (pair)* χωρίς λίστα, με διαφορετικό τρόπο
  - Κατασκευαστής: `pair`
  - Επιλογέας: `select`

```
>>> def pair(a, b):
        def asfunc(i):
            if i == 0:
                return a
            elif i == 1:
                return b
        return asfunc

>>> def select(x, i):
        return x(i)
```

Αφαίρεση δεδομένων

```
def pair(a, b):
    def asfunc(i):
        if i == 0:
            return a
        elif i == 1:
            return b
    return asfunc

def select(x, i):
    return x(i)

x = pair(2, 3)

a = select(x, 0)
p = select(x, 1)
```

μετάβαση στο Pythontutor

Αφαίρεση δεδομένων

- Η αλλαγή αναπαράστασης απαιτεί αλλαγές μόνο των `ritos, arith, paron`

```
>>> def ritos(a, p):
        return pair(a, p)

>>> def arith(x):
        return select(x, 0) # ή x(0)

>>> def paron(x):
        return select(x, 1) # ή x(1)
```

Αφαίρεση δεδομένων

- Οι συναρτήσεις των πάνω επιπέδων δεν επηρεάζονται από την αλλαγή και λειτουργούν με τον ίδιο τρόπο

```
>>> res = add_r(ritos(1,2), ritos(2, 3))
>>> print_r(res)
7 / 6
```

Αφαίρεση δεδομένων

- Όπως και με την υλοποίηση συνάρτησης, <u>οποιαδήποτε</u> υλοποίηση του κατασκευαστή και επιλογέων είναι έγκυρη εάν έχει την επιθυμητή συμπεριφορά

- Κριτήριο ορθής υλοποίησης ρητών:
  - Εάν `x=ritos(a, p)` τότε `arith(x)==a, paron(x) == p`
  - Επίσης, `gcd(arith(x), paron(x)) == 1`

- Κριτήριο ορθής υλοποίησης ζευγών:
  - Εάν `x = pair(a, b)` τότε `select(x, 0) == a` και `select(x, 1) == b`

Αφαίρεση δεδομένων

- Με την *αφαίρεση δεδομένων* επιτυγχάνουμε

  1. *Απλοποίηση προγραμμάτων* (γιατί δεν χρειάζεται να χειριζόμαστε πολλαπλές τιμές)

  2. *Γενικό κώδικα (*αφού αλλαγές σε μια αναπαράσταση δεν επιφέρουν αλλαγές σε κώδικα που χρησιμοποιεί διαφορετικά επίπεδα αφαίρεσης)

- Επόμενη διάλεξη: ακολουθίες