ΠΕΡΙΕΧΟΜΕΝΑ:

Πέτρος Γ.
Σμαραγδένιος Χορηγός Μαθήματος

Στράτος Κ.
Ασημένιος Χορηγός Μαθήματος

Μία **κλάση (class)**:

- Μοντελοποιεί μια ιδέα (ένα πρότυπο, μια κατηγορία οντοτήτων)
  - Ορίζουμε **"τι είναι"** αυτή η κατηγορία οντοτήτων
  - Προσοχή, όμως ότι αυτή η ιδέα **δεν "υπάρχει"** στον πραγματικό κόσμο.
- **Αποτελείται από δεδομένα και συναρτήσεις:**
  - Τα δεδομένα καλούνται **μέλη της κλάσης (class members)**
    - Ορίζουν χαρακτηριστικά της κλάσης.
  - Οι συναρτήσεις λέγονται και **μέθοδοι της κλάσης (class methods)**
    - Ορίζουν συμπεριφορά της κλάσης
  - Ενώ το "δέσιμο" (binding) μελών και μεθόδων σε μία κλάση αναφέρεται ως **ενθυλάκωση (encapsulation)**

Ένα **Αντικείμενο (object)**:

- Είναι ένα **στιγμιότυπο (instance)** της κλάσης
- Μπορούν να οριστούν οσαδήποτε στιγμιότυπα (instances) της κλάσης
- Είναι κάτι που "υπάρχει" στον πραγματικό κόσμο

Παραδείγματα:

- Αντικείμενα της κλάσης "Σκύλος" είναι ο σκύλος μου ο Πίκο, η Λάσι και ο Χάτσικο
- Αντικείμενο της κλάσης "Τρίγωνο" είναι ένα τρίγωνο που έχω ζωγραφίσει αυτή τη στιγμή στο χαρτί μου
- Αντικείμενο της κλάσης "Ακέραιος" είναι το 3, το 11 και το 99.

Ο **αντικειμενοστρεφής προγραμματισμός (object-oriented programming - OOP)** εδραιώθηκε με τη C++ στις αρχές των 80s.

- Σκεφτόμαστε, όχι πια τι συναρτήσεις θέλουμε για να κατασκευάσουμε το πρόγραμμα μας, αλλά ποια αντικείμενα χρειαζόμαστε για να διεκπεραιώσουν τις ενέργειες και τις οντότητες που θα αλληλεπιδράσουν.
- Αποδείχθηκε πολύ καλή ιδέα και γλώσσες όπως η Java παγίωσαν τον αντικειμενοστρεφή τρόπο σκέψης για μεγάλα προγράμματα. Σήμερα χρησιμοποιείται από (σχεδόν) όλες τις δημοφιλείς γλώσσες προγραμματισμού.

Ορισμός κλάσης:

- Ορισμός της κλάσης: **class Class_name:**
  - Το όνομα της κλάσης κατά σύμβαση ξεκινά με κεφαλαίο γράμμα.
  - Στο σώμα μπορούμε να ορίσουμε μέλη και μεθόδους
- Ορισμός αντικειμένου της κλάσης: obj_name = Class_name()
  - Ορίζει το αντικείμενο obj_name

Παράδειγμα 1: empty.class.py

```
class Empty:
    pass

e = Empty()
print(type(e))
```

Μέθοδος __init__ και μέλη κλάσης (class members ή attributes):

- Τα μέλη είναι τα δεδομένα της κλάσης. Η κλάση μπορεί να περιέχει οσεσδήποτε μεταβλητές οποιουδήποτε τύπου.
- Στο παράδειγμα, ορίζουμε το αρχέτυπο της αγελάδας και μία συγκεκριμένη αγελάδα (την molly):

```
# class.members.py
class Cow:
    def __init__(self, weight, hunger):
        self.weight = weight
        self.hunger = hunger

molly = Cow(500, 10)
print(type(molly))
print(molly.hunger)
```

- Η μέθοδος **__init__**:
  - Είναι μια ειδική μέθοδος, που τρέχει αυτόματα όταν κατασκευάζεται το αντικείμενο (κατασκευαστής – constructor)
    - Στην init, συνηθίζεται να αρχικοποιούμε τα μέλη μέσω παραμέτρων.
  - **self**: παράμετρος που διοχετεύεται αυτόματα, είναι μία αναφορά στο ίδιο το αντικείμενο. Πρέπει να υπάρχει (πρώτη) σε κάθε μέθοδο της κλάσης.
    - **self.member_name** (με καταχώρηση) ορίζει ότι η member_name θα είναι μέλος της κλάσης.

Άσκηση 1: Το αρχέτυπο του σκύλου

Ένας σκύλος χαρακτηρίζεται από:

- Το όνομά του
- Το βάρος του
- Το είδος του

Κατασκευάστε την κλάση σκύλος και ορίστε δύο στιγμιότυπα:

- Piko (10 κιλά – Terrier)
- Lassie (30 κιλά – Colley)

Άσκηση 2: Το αρχέτυπο του φιλοσόφου

Ένας φιλόσοφος χαρακτηρίζεται από:

- Το ονοματεπώνυμό του
- Την εποχή του (συμβολοσειρά, π.χ Ancient Greece)
- Τα βιβλία του (Λίστα π.χ. [“The Republic”, “Phaedon”])
- Την σχολή του (συμβολοσειρά π.χ. “Idealism”)
- Τα αμφισβητούμενα έργα του (να αρχικοποιείται με την κενή λίστα)

Κατασκευάστε την κλάση φιλόσοφος και ορίστε δύο στιγμιότυπα:

- Plato (Ancient Greek – [The Republic, Phaedon], “Platonism”)
- Baruch Spinoza (Modern Netherlands – [Ethics, Political Treatise], “Rationalism”)

Στην άσκηση αυτή, προσέξτε και από το βίντεο την αρχικοποίηση μέλους με κάποια default τιμή.

Μέθοδοι Κλάσης (class methods):

- Ορίζουν τη συμπεριφορά της κλάσης.
- Πρακτικά είναι συναρτήσεις μέσα στις κλάσεις, οι οποίες καθορίζουν τις ενέργειες που μπορεί να κάνει ένα αντικείμενο.
- Συνεχίζοντας το αρχέτυπο της αγελάδας:

```
# class.methods.py
class Cow:
    def __init__(self, weight, hunger):
        self.weight = weight
        self.hunger = hunger

    def express(self):
        if self.hunger > 5:
            print("Moooooowwwwwwwwww")
        else:
            print("Mowww")

molly = Cow(500, 10)
molly.express()
```

- Το 1ο όρισμα σε κάθε μέθοδο είναι υποχρεωτικά το self (αναφορά στο αντικείμενο)
- Κατά τ' άλλα ο ορισμός της μεθόδου έχει όλα τα χαρακτηριστικά μιας συνάρτησης (π.χ. μπορούμε να χρησιμοποιήσουμε νέες μεταβλητές, εσωτερικές συναρτήσεις, προαιρετικές παραμέτρους κ.λπ.)

Άσκηση 3: Επεκτείνοντας την κλάση με μεθόδους:

Επεκτείνετε την κλάση σκύλος της άσκησης 1 με τα εξής:

- Το μέλος mood (ακέραιος μεταξύ 5 και 10). Να αρχικοποιείται σε 5.
- Τη μέθοδο eat (αυξάνει την διάθεση κατά 1)
- Τη μέθοδο bark:
  - Αν η διάθεση είναι πάνω από 5, να τυπώνεται "Woof Woof Woof"
  - Αλλιώς να τυπώνεται μόνο "Woof"
- Τη μέθοδο walk:
  - Αυξάνει τη διάθεση κατά 1

Έπειτα υλοποιήστε το σενάριο:

- Ορίστε έναν σκύλο
- Ο σκύλος γαβγίζει
- Πάει βόλτα
- Γαβγίζει
- Πάει βόλτα
- Γαβγίζει
- Τρώει
- Γαβγίζει

Δημόσια Μέλη (και μέθοδοι):

- Όλα τα μέλη (και οι μέθοδοι εκτός της init) είναι **δημόσια (public)**, δηλαδή είναι προσβάσιμα και εκτός της κλάσης, π.χ.:

```
# public.py
class Cow:
   # as in previous example

molly = Cow(100, 5)
print(molly.hunger)
print(molly.weight)
molly.express()
```

Γιατί να περιορίσουμε την πρόσβαση στα μέλη (ενθυλάκωση);

- Για να επιτύχουμε την απόκρυψη πληροφορίας (information hiding). Η κλάση να δίνει έξω μόνο την απαραίτητη λειτουργικότητα και να μη γίνονται λάθη (π.χ. ο προγραμματιστής να αλλάξει κάτι που δεν έπρεπε)
- Άλλες γλώσσες (π.χ. C++ ή Java) είναι πολύ αυστηρές σε αυτό το θέμα:
  - Περίπλοκα προγραμματιστικά στοιχεία κατασκευάζονται μόνο για την διαχείριση της πρόσβασης.
- Η Python είναι πιο "χαλαρή":
  - Guido: Ο προγραμματιστής να είναι προσεκτικός.
  - **Θα χρησιμοποιούμε ιδιωτικά μέλη (και μεθόδους) μόνο αν έχουμε πραγματικά ισχυρό λόγο.**

Ιδιωτικά Μέλη (και μέθοδοι):

- Μπορούμε να ορίσουμε ιδιωτικά μέλη και μεθόδους, στα οποία έχουν πρόσβαση μόνο οι μέθοδοι της κλάσης.
- Π.χ. εδώ αλλάζουμε την κλάση ώστε η πείνα να είναι ιδιωτική μεταβλητή της αγελάδας:

```
# private.py
class Cow:
   def __init__(self, weight, hunger):
      self.weight = weight
      self.__hunger = hunger

   def express(self):
      if self.__hunger > 5:
         print("Mooooooowwwwwwwww")
      else:
         print("Mowww")


molly = Cow(100, 5)
print(molly.__hunger)
print(molly.weight)
molly.express()
```

- Ορίζουμε ότι ένα μέλος (ή μέθοδος) είναι ιδιωτικό, απλά θέτοντας δύο underscores μπροστά από το όνομά του

Άσκηση 4: Σημεία στο Επίπεδο

Η κλάση σημείο (του επιπέδου):

- Έχει ως ιδιωτικά μέλη τις συντεταγμένες του x,y
- Έχει ως μέθοδο τη set_x και τη set_y που παίρνουν ως παράμετρο την αντίστοιχη συντεταγμένη και θέτουν την τιμή της ίση με το όρισμα.
- Έχει ως μέθοδο τη get_x και τη get_y που επιστρέφουν τις αντίστοιχες συντεταγμένες

Ορίστε μία λίστα από 10 σημεία στο επίπεδο, έτσι ώστε αυτά να ορίζονται από τη συνάρτηση $y=x^2$ και έπειτα τυπώστε τα.
(οι τιμές του x να είναι στο εύρος 1…10)

Στις γλώσσες με αυστηρή απόκρυψη πληροφορίας, οι μέθοδοι:

- getter: Επιστρέφει την τιμή ενός μέλους
- setter: Θέτει την τιμή ενός μέλους

Είναι πολύ συνηθισμένες και λέγονται accessors. Έτσι το "στήσιμο" της παραπάνω άσκησης είναι λιγότερο Python-ικό και περισσότερο Java-ικό (και πιθανότατα θα το ξανασυναντήσουμε διαβάζοντας κώδικα άλλων προγραμματιστών)

Άσκηση 5: Συνάρτηση που παίρνει όρισμα αντικείμενο

Κατασκευάστε την κλάση φοιτητής

- Μέλη: Ονοματεπώνυμο και Βαθμός

Κατασκευάστε μία συνάρτηση με όνομα grade_student. Να παίρνει ως παράμετρο έναν φοιτητή. Να τον βαθμολογεί με έναν τυχαίο αριθμό από το 0 έως 10.

Κατασκευάστε μία συνάρτηση με όνομα average. Να παίρνει ως όρισμα μία λίστα φοιτητών και να υπολογίζει το μέσο όρο των βαθμολογιών τους και να τον τυπώνει.

Στο κυρίως πρόγραμμα: Κατασκευάστε μία λίστα με 10 τυχαίους φοιτητές (αρχικοποιήστε το όνομα και το επώνυμο με τυχαίο τρόπο και το βαθμό με -1). Έπειτα βαθμολογήστε τους και στη συνέχεια εξάγετε το μέσο όρο των βαθμολογιών τους.

Σχέση "έχει":

- Όπως είδαμε μια κλάση ορίζει τι "είναι" κάποια έννοια του κόσμου.
- και για να το ορίσουμε αυτό, περιγράφουμε **τί "έχει" (περιέχει, has-a)** μία κλάση: Θα περιέχει μεταβλητές ως μέλη.
- Και πλέον γνωρίζοντας ότι όλες οι μεταβλητές είναι στην πραγματικότητα αντικείμενα, μπορούμε να ορίσουμε ότι μια κλάση περιέχει αντικείμενα άλλων κλάσεων.

Παράδειγμα 1: Η πολυκατοικία και οι όροφοί της (building.py)

```
class Storey:
    def __init__(self):
        self.people = 0

class Building:
    def __init__(self, number_storeys):
        self.storeys = [Storey() for _ in range(number_storeys)]

    def add_people(self, storey, people):
        self.storeys[storey].people += people

    def print_people(self):
        for i in range(len(self.storeys)):
            print(f"Storey {i}: {self.storeys[i].people} people")

b = Building(3)
b.add_people(0,2)
b.print_people()
```

Άσκηση 6: Τα διαμερίσματα στους ορόφους

Επεκτείνετε το πρόγραμμα του παραδείγματος 1:

- Κατασκευάστε μία κλάση με όνομα flat, με μέλος πόσα άτομα περιέχει (people). Να αρχικοποιείται σε 0.
- Τροποποιήστε την storey: Να περιέχει ως μέλος μία λίστα με διαμερίσματα. Η init να αρχικοποιεί τη λίστα με number_flats διαμερίσματα σε κάθε όροφο (number_flats, δηλαδή να είναι παράμετρος).
- Τροποποιήστε την building: Η init να δέχεται ως έξτρα παράμετρο και το πλήθος των διαμερισμάτων σε κάθε όροφο
- Τροποποιήστε κατάλληλα τις μεθόδους όλων των κλάσεων ώστε το πλήθος των ατόμων να αποθηκεύονται στο διαμέρισμα (και όχι πλέον στον όροφο).

Άσκηση 7: Κλάση Γραμμή

Επεκτείνοντας την άσκηση 4, ορίστε την κλάση γραμμή:

- Ορίζεται από δύο σημεία point_A και point_B (απλοποιήστε την κλάση Point ώστε να είναι πιο pythonian, χωρίς ιδιωτικές μεταβλητές και επεκτείνετέ την με μία μέθοδο που να τυπώνει μορφοποιημένα τις συντεταγμένες του σημείου (print))
- Ο κατασκευαστής να δέχεται ορίσματα τα δύο σημεία. Αν δεν διοχετεύονται ορίσματα, να τα αρχικοποιεί στο (0,0).
- Ορίστε setters για τα δύο σημεία.
- Ορίστε τη μέθοδο length, η οποία να υπολογίζει το μήκος της ευθείας από τον τύπο: $\sqrt{(A.x - B.x)^2 + (A.y - B.y)^2}$.
  Σημείωση: Η τετραγωνική ρίζα υπολογίζεται με χρήση της συνάρτησης sqrt που έχει οριστεί στο module math.

Στο κυρίως πρόγραμμα, πρώτα ορίστε μια γραμμή χωρίς αρχικοποίηση σημείων και έπειτα υπολογίστε το μήκος του ευθυγράμμου τμήματος με άκρα τα σημεία (1,1) και (4,5)

Άσκηση 8: Time

Ορίστε την κλάση time:

- Μέλη θα είναι τα hour, minute, second (κατασκευάστε init με default τιμές παραμέτρων σε 0)
- Θα υπάρχουν setters που θα ελέγχουν αν οι τιμές είναι έγκυρες. 0..23 για την ώρα και 0..59 για λεπτά και τα δευτερόλεπτα. Ο έλεγχος να γίνεται και για τις τιμές που τίθενται στην init.
- Τη μέθοδο total_seconds η οποία επιστρέφει το συνολικό πλήθος δευτερολέπτων της χρονικής στιγμής που απεικονίζεται στο αντικείμενο
- Τη μέθοδο print που τυπώνει μορφοποιημένα την χρονική στιγμή.
- Τη μέθοδο next_second, η οποία κατασκευάζει ένα αντικείμενο της κλάσης time που απεικονίζει την χρονική στιγμή ένα δευτερόλεπτα μετά από αυτό που απεικονίζει το αντικείμενο και το επιστρέφει.

Ελέγξτε την ορθότητα της κατασκευής σας, με κατάλληλα στιγμιότυπα στον κώδικα.

Άσκηση 9: object reference

Κατασκευάστε μία άδεια κλάση

- Τυπώστε την self στην __init__
- Δηλώστε ένα αντικείμενο και τυπώστε το

Παρατηρήστε τις εκτυπώσεις όσον αφορά τη διεύθυνση του αντικειμένου.

Χαρακτηριστικά κλάσης

- Μπορούμε να ορίσουμε μεταβλητές που είναι κοινές για όλα τα αντικείμενα τις κλάσεις.
- Είναι γνωστές σαν **χαρακτηριστικά κλάσης (class attribute)**

```
# class.attribute.py
class C:
    counter = 0

    def __init__(self):
        C.counter += 1

o1 = C()
o2 = C()
print(C.counter, o1.counter, o2.counter)
```

Η λέξη κλειδί del

- Μπορούμε να διαγράψουμε οποιοδήποτε αντικείμενο με τη λέξη – κλειδί del ως: **del object**

(βλ. και del.py)

Όλοι οι τύποι δεδομένων είναι κλάσεις και όλες οι μεταβλητές είναι αντικείμενα.

- Ελέγχουμε αν δύο αντικείμενα είναι τα ίδια (ίδια διεύθυνση) με τον τελεστή is.

```
print([1,2] is [1,2])
```

- Ελέγχουμε αν δύο αντικείμενα έχουν ίδιες τιμές με τον τελεστή ==:

```
print([1,2] == [1,2])
```

- Ελέγχουμε αν ένα αντικείμενο είναι μίας συγκεκριμένης κλάσης με τη συνάρτηση isinstance

```
print(isinstance([1,2],list))
```

- (βλ. πρόγραμμα is.py για περισσότερες λεπτομέρειες)

Συχνές ερωτήσεις:

Που κατασκευάζουμε μια κλάση

- Θεωρείται καλή πολιτική, να έχουμε κάθε κλάση σε ένα ξεχωριστό module.

Πότε κατασκευάζουμε μία κλάση;

- Είναι το πιο συνηθισμένο ερώτημα που θα ρωτάμε τον εαυτό μας.
- Το σύνηθες είναι ότι κάτι εννοιολογικά αυτόνομο, πρέπει να είναι μια κλάση.
- Αλλά η εμπειρία – με τη λύση πολλών ασκήσεων – θα βοηθήσει να παίρνουμε μια πιο σωστή απόφαση.

Άσκηση 10: Τραπεζικοί Λογαριασμοί

(Α) Ένας τραπεζικός λογαριασμός ορίζεται από:

- τον αριθμό του (15-ψήφια συμβολοσειρά)
- τον κάτοχό του (ονοματεπώνυμο)
- το ποσό που περιέχει (πραγματικός)

Ορίστε δύο τραπεζικούς λογαριασμούς (με στοιχεία της αρεσκείας σας).

Επεκτείνετε το λογαριασμό με μία μέθοδο (transfer) η οποία θα παίρνει σαν όρισμα έναν άλλο τραπεζικό λογαριασμό και ένα ποσό και θα μεταφέρει το ποσό σε αυτόν (εφόσον υπάρχει επαρκές υπόλοιπο) εκτυπώνοντας ένα περιεκτικό μήνυμα που να περιέχει τα ονόματα των κατόχων των λογαριασμών.

Επαληθεύστε τη σωστή λειτουργία της μεθόδου transfer.

(Β) Κατασκευάστε μία κλάση με όνομα Person η οποία περιέχει το ονοματεπώνυμο, την ηλικία και τον αριθμό ταυτότητας του ατόμου. Ορίστε δύο άτομα στο κυρίως πρόγραμμα.

Τροποποιήστε τον ορισμό του λογαριασμού, ώστε να περιέχει ένα αντικείμενο τύπου Person, αντί να περιέχει απλά το ονοματεπώνυμο του κατόχου.

Άσκηση 11: Κλάση που περιέχει λίστα αντικειμένων

Ένα βιβλίο (σε ένα βιβλιοπωλείο) ορίζεται από:

- το τίτλο του
- τη λίστα των συγγραφέων του
- την τιμή του
- το πλήθος των αντιτύπων του

Επίσης ορίζονται οι μέθοδοι:

- print_full_title: Εκτυπωνει τον τίτλο, ακολουθούμενο από τους συγγραφείς χωρισμένους με κόμματα.
- add_author: προσθέτει ένα συγγραφέα στη λίστα συγγραφέων

Ένας συγγραφέας ορίζεται από:

- το ονοματεπώνυμό του
- το email του

Στο κυρίως πρόγραμμα:

- Ορίστε το βιβλίο “The Ultimate Python Cookbook” των Bob Patterson και James McConan (με αντίστοιχα e-mail bob@pcookbook.com και james@pcookbook.com) με τιμή 18.25 το οποίο το έχουμε σε 43 αντίγραφα.
- Τυπώστε τον τίτλο.
- Προσθέστε το συγγραφέα Tom Rossbow (tom@pcookbook.com)
- Τυπώστε τον τίτλο.

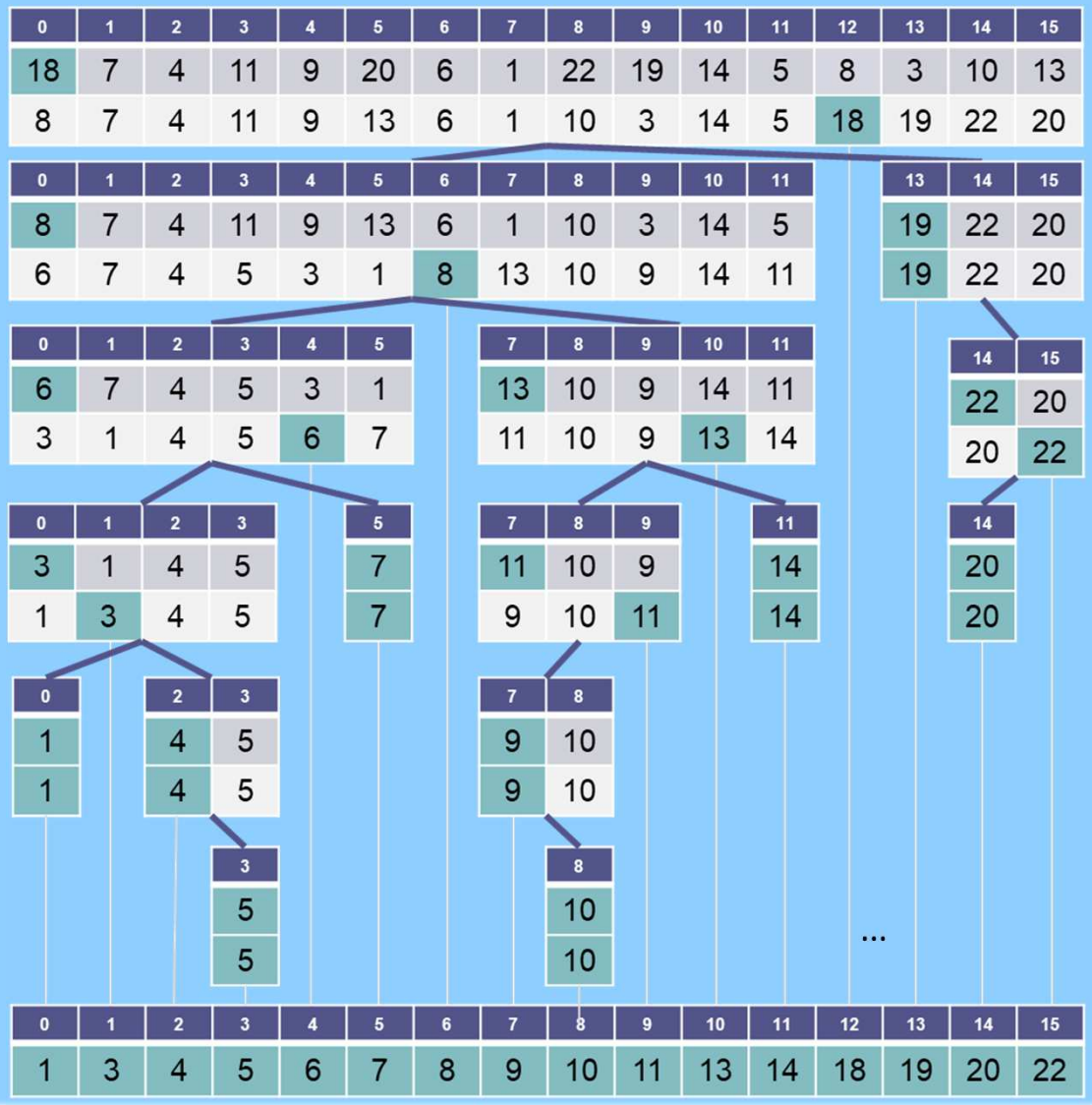
Algorithm: Quick Sort (Γρήγορη Ταξινόμηση)

Συνεχίζουμε με το πρόβλημα της ταξινόμησης, με έναν ακόμη αλγόριθμο που το επιλύει.

Το **σκεπτικό της γρήγορης ταξινόμησης (quick sort)** είναι:

- Επιλογή ενός στοιχείου ως “οδηγό στοιχείο”. Χωρίζει τον πίνακα σε τρία μέρη: Τα μικρότερα στοιχεία από το οδηγό στοιχείο, το οδηγό στοιχείο και τα μεγαλύτερα στοιχεία από το οδηγό στοιχείο
- (Αναδρομικά) Ταξινόμησε τον υποπίνακα αριστερά του οδηγού στοιχείου
- (Αναδρομικά) Ταξινόμησε τον υποπίνακα δεξιά του οδηγού στοιχείου

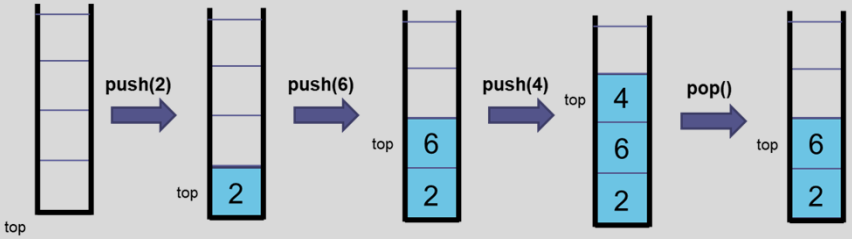
Παράδειγμα Εκτέλεσης:



<u>Άσκηση 12:</u> Υλοποιήστε τον αλγόριθμο quick sort

Άσκηση 13.1: Στοίβα

Η στοίβα είναι μια δομή δεδομένων με γραμμική διάταξη, στην οποία η ώθηση (push) και η εξαγωγή (pop) γίνονται από την κεφαλή της στοίβας (LIFO: Last In, First Out)



Ορίστε την Κλάση Στοίβα (Stack):

- Περιέχει μία λίστα (με όνομα array)
- Αρχικοποιείται στην κενή λίστα
- Μέθοδος push: Παίρνει όρισμα κάποιο στοιχείο και το βάζει στο τέλος του array
- Μέθοδος pop: Εξάγει το στοιχείο στο τέλος του array και το επιστρέφει. Επιστρέφει None αν η λίστα είναι άδεια
- Μέθοδος length: Επιστρέφει το πλήθος των στοιχείων της στοίβας

Ελέγξτε ότι η υλοποίησή σας λειτουργεί σωστά.
Η στοίβα να οριστεί σε module με όνομα stack.py.

Άσκηση 13.2: Δυαδική Αναπαράσταση Αριθμού

Για τον υπολογισμό της δυαδικής αναπαράστασης ενός ακέραιου αριθμού κάνουμε διαδοχικές διαιρέσεις με το 2.

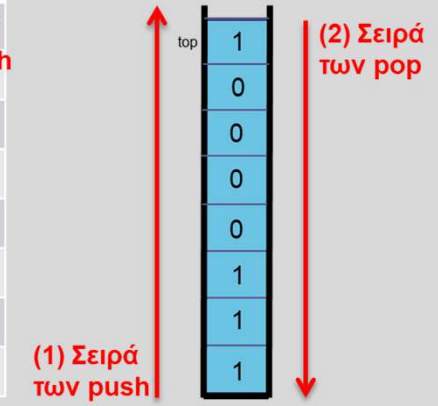
- Εισάγουμε τα διαδοχικά υπόλοιπα σε μία στοίβα
- και έπειτα εξάγοντας τα έχουμε τη δυαδική του αναπαράσταση (βλ. παράδειγμα)

ΥΠΟΛΟΓΙΣΜΟΙ

Αριθμός /2	Πηλίκο	Υπόλοιπο
135/2	67	1
67/2	33	1
33/2	16	1
16/2	8	0
8/2	4	0
4/2	2	0
2/2	1	0
1/2	0	1

(1) Σειρά των push

ΣΤΟΙΒΑ



ΔΥΑΔΙΚΟΣ ΑΡΙΘΜΟΣ

(2) Σειρά των pop

1 0 0 0 0 1 1 1

Άρα $(135)_{10}=(10000111)_2$

Κατασκευάστε συνάρτηση η οποία με όρισμα n να επιστρέφει τη δυαδική του αναπαράσταση (ως συμβολοσειρά).

Παρατηρήσεις

- Στις Δομές Δεδομένων, θεωρείται σημαντικό ότι όλες οι ενέργειες στο κυρίως πρόγραμμα, πρέπει να γίνονται μέσω των βασικών μεθόδων (εδώ η push και η pop) στο κυρίως πρόγραμμά μας.

Άσκηση 14.1: Κλάση Character

Ορίστε την κλάση Character (Χαρακτήρας του παιχνιδιού) με μέλη:

- όνομα
- δείκτης υγείας health (να αρχικοποιείται σε 100)
- δείκτης ταχύτητας attack_speed (να έχει default αρχικοποίση σε 2 – είναι από 1-10, όσο πιο μεγάλος τόσο πιο γρήγορα χτυπάει)
- δείκτης καθυστέρησης delay (να έχει default αρχικοποιηση σε 0 – απεικονίζει πόσο πρέπει να περιμένει για να ξαναχτυπήσει)

και την εξής λειτουργικότητα:

- attack(): Επιστρέφει έναν τυχαίο ακέραιο στο εύρος 3-10. Θέτει το delay = 10-attack_speed
- is_dead(): Επιστρέφει T/F ανάλογα με τον είναι νεκρός (υγεία = 0)
- end_round() – Αυξάνει το δείκτη υγείας κατά 1, μειώνει το delay κατά 1
- print(): Εκτυπώνει τα στοιχεία του χαρακτήρα

Άσκηση 14.2: Κλάση Arena

Ορίστε την κλάση Arena με μέλη:

- δύο teams (A και B) που είναι λίστες χαρακτήρων

και την εξής λειτουργικότητα:

- print_state(): Εκτυπώνει την κατάσταση των ομάδων
- play(): Οργανώνει το παιχνίδι σε χρονικές στιγμές (ατέρμων βρόχος). Σε κάθε χρονική στιγμή:
  - Κατασκευάζει μία λίστα με τους χαρακτήρες που μπορούν να παίξουν (delay = 0)
  - Κάθε χαρακτήρας που μπορεί να παίξει επιλέγει έναν αντίπαλο χαρακτήρα και τον χτυπάει. Από τον αντίπαλο αφαιρείται τόση υγεία, όσο και το χτύπημα
  - Γίνεται έλεγχος νεκρών χαρακτήρων. Οι νεκροί αφαιρούνται από την ομάδα.
  - Γίνεται έλεγχος αν υπάρχει νικήτρια ομάδα. Αν ναι, το παιχνίδι τερματίζει με την ανακοίνωση του νικητή.
  - καλείται η end_round για κάθε ζωντανό παίκτη

Άσκηση 14.3: main

Έπειτα βάλτε το παιχνίδι σε εφαρμογή (main):

- Η 1η ομάδα να αποτελείται από 5 orcs (attack_speed = 2 και με τυχαίες αρχικοποίησεις στο delay από 0 έως 3)
- Η 2η ομάδα να αποτελείται από 3 night elfs (attack_speed = 3, με τυχαίες αρχικοποιήσεις στο delay από 0 έως 2)

Μεριμνήστε στο πρόγραμμα για ποιοτικές εκτυπώσεις!

Refactoring:

- Ο όρος refactoring αναφέρεται στον επανασχεδιασμό του κώδικα, χωρίς να αλλάξει η εξωτερική συμπεριφορά του (π.χ. οι εκτυπώσεις που έκανε)
- Είναι συχνή διαδικασία, ιδίως όταν το project μας μεγαλώνει σε μέγεθος, να ανασχεδιάζεται ώστε να διορθώνονται προβλήματα που βρέθηκαν στον σχεδιασμό του κατά τη διάρκεια της επέκτασής του.
- Θα κάνουμε refactoring στο project μας, μετατρέποντας το σε αντικειμενοστρεφές.

Άσκηση 15.1: Κλάση Teacher (teacher.py)

Μέλη:

- Ίδια με αυτά του λεξικού. Κάντε μετονομασία σε first_name, last_name και teacher_id στα κλειδιά (και στο JSON αρχείο)

Μέθοδοι:

- __init__ : Να αρχικοποιεί μέσω ορισμάτων ένα αντικείμενο. Να έχει και default τιμές ορισμάτων.
- from_dict: Παίρνει όρισμα ένα λεξικό (όπως στο JSON) και αρχικοποιεί το αντικείμενο
- to_dict: Επιστρέφει το λεξικό που απεικονίζει το αντικείμενο.
- print_teacher(): Εκτυπώνει μορφοποιημένα έναν καθηγητή.

Άσκηση 15.2: Κλάση Teachers (teachers.py)

Μέλη:

- Περιέχει λίστα με όνομα teachers (αντικείμενα τύπου Teacher)

Μέθοδοι:

- __init__: Διαβάζει από το αρχείο JSON και αποθηκεύει στη λίστα τους καθηγητές
- save_teachers_data(): Σώζει στο αρχείο JSON τους καθηγητές (ως λεξικά)
- next_id(): Βρίσκει το επόμενο id
- create_teacher(first_name, surname): Δημιουργεί νέο καθηγητή (εφόσον δεν υπάρχει ήδη στη λίστα)
- read_teacher(teacher_id): Επιστρέφει τον teacher με το συγκεκριμένο id.
- update_teacher(teacher_id): Θέτει στο χρήστη να επιλέξει ποιο χαρακτηριστικό θέλει να διορθώσει (εκτός του id)
- delete_teacher(): Ζητάει το id και έπειτα να διαγράφει τον καθηγητή (αν υπάρχει)

Κάντε τις απαραίτητες διορθώσεις και στη main()

"Readability counts."
Zen of Python #7