ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΟΙΚΟΝΟΜΙΑΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ

ΜΑΘΗΜΑ: ΕΠΙΣΤΗΜΟΝΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΕ ΓΛΩΣΣΑ PYTHON

ΔΙΔΑΣΚΩΝ: Μ. ΖΑΜΠΟΓΛΟΥ

ΘΕΜΑΤΑ ΕΞΕΤΑΣΕΩΝ

ΙΟΥΝΙΟΣ 2025

Διάρκεια εξέτασης: 2 ώρες

Ερώτημα 1 (1.5 μονάδες)

Τι θα εμφανίσουν τα παρακάτω τμήματα κώδικα:

1.1)

```
x = "10"
y = 3
z = float(x) + y
print(z)
```

A) `13` B) `'103'`
C) `13.0` D) `<class 'float'>`

1.2)

```
a = "13"
b = 1
c = (int(a[-1]) + b) ** 2
print(c)
```

A) `16` B) `169`
C) `4` D) `TypeError`

1.3)

```
a = 10
b = "3"
c = a / int(b) > a / int(b) - 1
print(type(c))
```

A) `True` B) `False`
C) `<class 'bool'>` D) `TypeError`

1.4)

```
a = "12,34"
b = a.split(',')
print(len(a) < len(b))
```

A) `True` B) `False`
C) `None` D) `TypeError`

1.5)

```
a = 'hello'
def x(a,b):
    z = a[0]+b
    return z
print(x(a))
```

A) `hello` B) `h`
C) `ello` D) `TypeError`

1.6)

```
file = open('data.txt')
a = file.read()
file.close()
print(a[-1])
```

Περιεχόμενα του data.txt:

```
Python is wonderful.
I wish I could code all day
```

A) `I wish I could code all day`
B) `y`
C) `.` D) `TypeError`

Ερώτημα 2 (1.5 μονάδες)

Τι θα εμφανίσει το καθένα από τα παρακάτω τμήματα κώδικα; Δικαιολογήστε την απάντησή σας **εν συντομία**:

2.1)
```
i = 10
while i > 0:
      i -= 3
      print("*")
      if i <= 3:
            break
```

2.2)
```
a = set([3, 6, 4, 3, 6])
for i in a:
      print("*")
```

2.3)
```
d = {"x": (2, 4), "z": (4, 5)}
for k in d.keys():
            d[k].append(0)
            print(d[k])
```

2.4)
```
a="Python 2025"
print(a[-8:5])
```

2.5)
```
a = [1, 2, 3, 4, 0]
b = a
a.sort()
print(a[0]+b[0])
```

2.6)
```
def my_fun(a,b):
    if a<b:
        if b>4:
            return(a*b)
    else:
        return(a+b)
    return(a)
print(my_fun(7,3))
```

Ερώτημα 3 (1 μονάδα)

Δίνεται μια λίστα **A** που αποτελείται από λίστες των τριών αντικειμένων. Να γραφεί μια γραμμή κώδικα list comprehension ("υπολογιζόμενες λίστες") που να δημιουργεί μια νέα λίστα **B** που θα περιέχει το τρίτο αντικείμενο κάθε λίστας. Π.χ. αν
A=[[1,5,1], [4,6,2], [2,5,6], [4,6,1]],
η **B** θα είναι η **[1,2,6,1]**. *Δε χρειάζεται εντολή εισόδου, θεωρήστε ότι η A υπάρχει ήδη.*

Ερώτημα 4 (2 μονάδες)

Να γραφεί κώδικας σε Python που

1) θα ζητάει από το χρήστη μια πρόταση και θα την αποθηκεύει σε ένα string **S**
2) θα αφαιρεί από την πρόταση τον τελευταίο χαρακτήρα
3) θα χωρίζει την πρόταση σε λέξεις (υποθέτοντας ότι χωρίζονται με κενά)
4) θα δημιουργεί ένα λεξικό **D** που θα έχει ως κλειδιά τις λέξεις της πρότασης, και οι τιμές του θα είναι Boolean που θα είναι true αν η αντίστοιχη λέξη έχει πάνω από 3 χαρακτήρες.

(***Συνεχίζεται στην επόμενη σελίδα***)

Για παράδειγμα, αν δώσω το string:
"Η Python είναι μια καταπληκτική γλώσσα προγραμματισμού!", θα πρέπει να δημιουργήσει το
D={"Η":False, "Python":True, "είναι": True, "μία": False, "καταπληκτική": True, "γλώσσα": True, "προγραμματισμού": True}

Ερώτημα 5 (1 μονάδα)

Δίνεται το παρακάτω τμήμα κώδικα:

```
b=5
def my_fun(a,b):
      a=a+2
      b=b+2
      d=a*b
      return d
a = 3
c = my_fun(a=a, b=2)
print(a)
print(b)
print(c)
print(d)
```

Τι θα εμφανίσει κάθε μία από τις τέσσερις τελευταίες εντολές;

Ερώτημα 6 (2 μονάδες)

Ένα σύστημα Smart Home έχει όλες τις πιθανές συνδεδεμένες συσκευές οργανωμένες σε μια αντικειμενοστραφή ιεραρχία.

Α) Δημιουργήστε μια κλάση **`SmartDevice`** με μια ιδιότητα **`name`** και μια ιδιότητα **`is_on`**. Κατά την αρχικοποίηση, το **`name`** θα δίνεται ως παράμετρος ενώ η **`is_on`** θα ξεκινάει πάντα **`False`**. Θα υπάρχει επίσης μια μέθοδος **`toggle_power`** χωρίς παραμέτρους, που θα τροποποιεί την **`is_on`** (αν είναι **`True`** θα την κάνει **`False`** και αντίστροφα). (1 μονάδα)

Β) Δημιουργήστε μια υποκλάση της **`SmartDevice`** με όνομα **`Thermostat`** και μια επιπλέον ιδιότητα, την **`target_temp`**, που θα αρχικοποιείται ως παράμετρος αλλά θα έχει προεπιλεγμένη (default) τιμή το **`22`**. Η **`Thermostat`** θα έχει επίσης μια μέθοδο **`check_temp(current_temp)`** η οποία θα δέχεται την τρέχουσα θερμοκρασία. Αν η τρέχουσα θερμοκρασία είναι ίση με την **`target_temp`** και η **`is_on`** είναι **`True`**, θα εκτελεί την **`toggle_power()`** του.

Ερώτημα 7 (1.5 μονάδες)

Δίνεται ο παρακάτω κώδικας σε Python:

```
import numpy as np
n=50
matrix = np.random.rand(n,1000)
```

α) Να συνεχιστεί ο κώδικας ώστε να υπολογίζει και να εμφανίζει το πλήθος των γραμμών του **matrix** που έχουν μέση τιμή μεγαλύτερη του **0.6**. (1 μονάδα)

Hint: θυμηθείτε ότι στην Python οι τιμές *True* και *False* είναι ταυτόχρονα οι αριθμοί 0 και 1

β) Αν μεγαλώσω το n σε 100, μετά σε 1.000, μετά σε 10.000 και για κάθε νέα τιμή του n επανεκτελέσω τον κώδικα, θα παρατηρήσω ότι όσο μεγαλώνει το n, οι μέσες τιμές των γραμμών θα τείνουν σταδιακά σε έναν αριθμό. Σε ποιον; Γιατί; (0.5 μονάδες)

Χρήσιμες μέθοδοι της Python (**δεν είναι απαραίτητο να τις χρησιμοποιήσετε όλες**):

list.append(*x*): επέκταση λίστας με νέο στοιχείο

list.sort(*, *key=None*, *reverse=False*): ταξινόμηση λίστας in-place

str.split(*sep=None*, *maxsplit=-1*): διαχωρισμός string

str.replace(*old*, *new*[, *count*]): αντικατάσταση περιεχομένου string

numpy.mean(*a*, *axis=None*, *dtype=None*, *out=None*, *keepdims=<no value>*, *, *where=<no value>*): Μέση τιμή / μέσες τιμές των στοιχείων του πινακα NumPy **a**. Συνολική ή κατά μήκος ενός άξονα.

numpy.sum(*a*, *axis=None*, *dtype=None*, *out=None*, *keepdims=<no value>*, *initial=<no value>*, *where=<no value>*): Άθροισμα/αθροίσματα των στοιχείων του πινακα NumPy **a**. Συνολικό ή κατά μήκος ενός άξονα.