ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΜΕ PYTHON (ΟΜΑΔΑ Β)

ΑΣΚΗΣΗ 1 [1 μονάδα]

α) Γράψτε έναν βρόχο while που κάνει ακριβώς το ίδιο με τον εξής βρόχο for:

```
word = "cinematography"
for i in range(len(word)):
        print 'i =', i, word[i]
```

β) Ποιο θα είναι το αποτέλεσμα της εκτέλεσης των παρακάτω γραμμών κώδικα;

```
x = 0
while x <= 3:
    if x%2 == 1:
        print 'Patra'
    elif x – 1 > 0:
        print 'Athens'
    else:
        print 'Volos'
    x = x + 1
```

ΛΥΣΗ

α)

```
word = "cinematography"

i=0
while i < len(word):
        print 'i =', i, word[i]
        i += 1
```

β)

Volos
Patra
Athens
Patra

ΑΣΚΗΣΗ 2 [1 μονάδα]

α) Μετατρέψτε τον δεκαεξαδικό αριθμό $(1D2.B2)_{<16>}$ σε οκταδικό μέσω του δυαδικού συστήματος, δείχνοντας όλα τα βήματα της μετατροπής.

β) Εκτελέστε τις ακόλουθες πράξεις στα συστήματα που αναφέρονται για κάθε πράξη:

$(11.101)_{<2>} + (10.101)_{<2>}$

$(6F.E2)_{<16>} + (F1.12)_{<16>}$

ΛΥΣΗ

α) $(1D2.B2)_{<16>} = 0001\ 1101\ 0010\ .\ 1011\ 0010_{<2>} =$

$000\ 111\ 010\ 010\ .\ 101\ 100\ 100_{<2>} = (722.544)_{<8>}$

β)

Η εκτέλεση των προσθέσεων γίνεται ως ακολούθως:

μεταφορά:		111 1
		11.101
	+	10.101
		110.010

μεταφορά: 11

6F.E2

\+ F1.12

160.F4

ΑΣΚΗΣΗ 3 [1 μονάδα]
Δίνεται η λογική συνάρτηση:

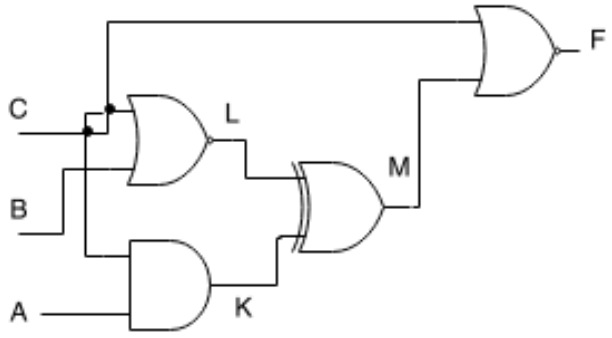
F = ((A AND C) XOR (B OR C)) OR C

i) Σχεδιάστε το λογικό κύκλωμα που υλοποιεί την F.

ii) Σχηματίστε τον πίνακα αληθείας της F.

ΛΥΣΗ

i) Το λογικό κύκλωμα έχει ως εξής:



ii) Ο πίνακας αληθείας είναι:

A	B	C	K	L	M	F
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	0	1	1	1
0	1	1	0	1	1	1
1	0	0	0	0	0	0
1	0	1	1	1	0	1
1	1	0	0	1	1	1
1	1	1	1	1	0	1

ΑΣΚΗΣΗ 4 [1 μονάδα]
Βρείτε τι εκτελεί η παρακάτω αναδρομική συνάρτηση foobar με παράμετρο arg μια λίστα:

```
def foobar(arg):
    if arg == []:
        return arg
    else:
        return foobar(arg[1:]) + [arg[0]]
```

Χρησιμοποιήστε ως περιπτώσεις εισόδου τις εξής 3 λίστες:

```
arg1 = []
arg2 = [4,10,8]
arg3 = [2,4,4,2]
```

ΛΥΣΗ

Αν η arg είναι κενή λίστα, η foobar επιστρέφει την ίδια (κενή) λίστα. Αν η arg δεν είναι κενή λίστα, π.χ., arg = [4,10,8], τότε foobar([4,10,8]) = foobar([10,8]) + [4], αλλά foobar([10,8]) = foobar([8]) + [10] και foobar([8]) = foobar([]) + [8] = [] + [8] = [8]. Επομένως, foobar([4,10,8]) = [8] + [10] + [4] = [8,10,4], δηλαδή, η συνάρτηση foobar αντιστρέφει τα στοιχεία μιας λίστας.

ΑΣΚΗΣΗ 5 [1 μονάδα]

$$A = \begin{bmatrix} 4 & 2 & 1 \\ 7 & 10 & 1 \\ 2 & 1 & 4 \end{bmatrix},$$

Χρησιμοποιείστε την python ή το matlab για να βρείτε:

- Τον αντίστροφο του πίνακα Α, τις ιδιοτιμές και τα ιδιοδιανύσματα του πίνακα Α
- Την γραφική παράσταση της f(x)= 3*x+2*sin(x)+5*log(x+1)+1 στο διάστημα [0,2π]

ΛΥΣΗ

```
A=[4 2 1; 7 10 1; 2 1 4];
inv(A)
[p d]=eig(A)
x = 0:pi/100:2*pi;
y=3*x+2*sin(x)+5*log(x+1)+1;
plot(x,y)
```

ΑΣΚΗΣΗ 6 [1 μονάδα]

Γράψτε μια συνάρτηση recEq που παίρνει ως παράμετρο εισόδου ένα θετικό ακέραιο αριθμό n και επιστρέφει σε λίστα τους όρους της ακολουθίας:

$a_k = 4(-1)^k a_{k-1}$, $a_1 = 3$, για k = 2, …, n.

ΛΥΣΗ

```
def recEq(n):
    a = []
    k = 1
    while k <= n:
        if k == 1:
            a.append(3)
        else:
            a.append((-1)**k * 4 * a[-1])
        k += 1
    return a
```

ΑΣΚΗΣΗ 7 [1 μονάδα]

Γράψτε μια συνάρτηση ConvertEurosToDollars που παίρνει τρεις παραμέτρους εισόδου: amount (που εκφράζει το ποσό σε ευρώ), rate (που εκφράζει την ισοτιμία ευρώ-δολάριου), charge (που εκφράζει ένα πόσο χρέωσης της συναλλαγής). Η συνάρτηση ConvertEurosToDollars πρέπει να επιστρέφει το καθαρό ποσό σε δολάρια ενός ποσού ευρώ μετά την αφαίρεση της χρέωσης της συναλλαγής. Η χρέωση της συναλλαγής είναι charge, αν το amount (ποσό σε ευρώ) είναι μικρότερο του 1000, και διπλασιάζεται (charge*2) όταν το amount είναι μεγαλύτερο του 1000.

ΛΥΣΗ

```
def ConvertEurosToDollars(amount, rate, charge):
        total = amount * rate
        if amount < 1000:
                fee = charge
        else:
                fee = 2*charge
        return total – fee
```

ΑΣΚΗΣΗ 8 [1 μονάδα]

Γράψτε μια συνάρτηση hotelStay με τέσσερις παραμέτρους εισόδου: price (που εκφράζει την τιμή του δωματίου του ξενοδοχείου ανά διανυχτέρευση), days (που εκφράζει τις διανυχτερεύσεις στο ξενοδοχείο),

tax (που εκφράζει τον ΦΠΑ επί τοις εκατό στη συνολική τιμή) και fee (που εκφράζει μια επιπρόσθετη χρέωση ανά μέρα παραμονής αφού συμπεριληφθεί στο ποσό συνολικής πληρωμής και ο ΦΠΑ). Η συνάρτηση hotelStay πρέπει να επιστρέφει το τελικό ποσό που στοιχίζει η παραμονή στο ξενοδοχείο. Διευκρινίζεται ότι η τιμή του fee δεν προστίθεται στο τελικό ποσό αν οι μέρες διαμονής είναι περισσότερες των 10, ενώ λαμβάνεται υποψή υποδιπλασιασμένη (δηλ. fee/2) αν οι μέρες διαμονής είναι από 6 έως και 10.

ΛΥΣΗ

```
def hotelStay(price, days, tax, fee):
        if days > 10:
                fee = 0
        elif days > 5:
                fee = fee/2.
        total = price * days
        totalPlusTax = total * (1 + tax/100.)
        return totalPlusTax + fee * days
```

ΑΣΚΗΣΗ 9 [2 μονάδες]

Έστω ότι τα αποτελέσματα κάποιων αγώνων μπάσκετ δίνονται με μια λίστα της μορφής
agwnes = ['aa-bb: 73-61', 'aa-cc: 80-81', 'bb-cc: 70-62', 'cc-dd: 72-82', 'dd-aa: 62-60'],
όπου aa, bb, cc, dd είναι ομάδες (παρατηρείστε ότι αμέσως μετά το ':' υπάρχει κενό).
Γράψτε μια συνάρτηση pontoi με παράμετρο εισόδου τη λίστα agwnes, η οποία να επιστρέφει ένα λεξικό με κλειδιά τις ομάδες της λίστας agwnes και τιμές τους συνολικούς πόντους που αντίστοιχα πέτυχαν οι ομάδες στους αγώνες.

ΛΥΣΗ

```
def kalathia(agwnes):
    points = {}
    for i in agwnes:
        tl=i.split(": ")
        tl1=tl[0].split("-")
        tl2=tl[1].split("-")
        for i in range(2):
            if tl1[i] not in points:
                points[tl1[i]]=int(tl2[i])
            else:
                points[tl1[i]]+=int(tl2[i])
    return points
```