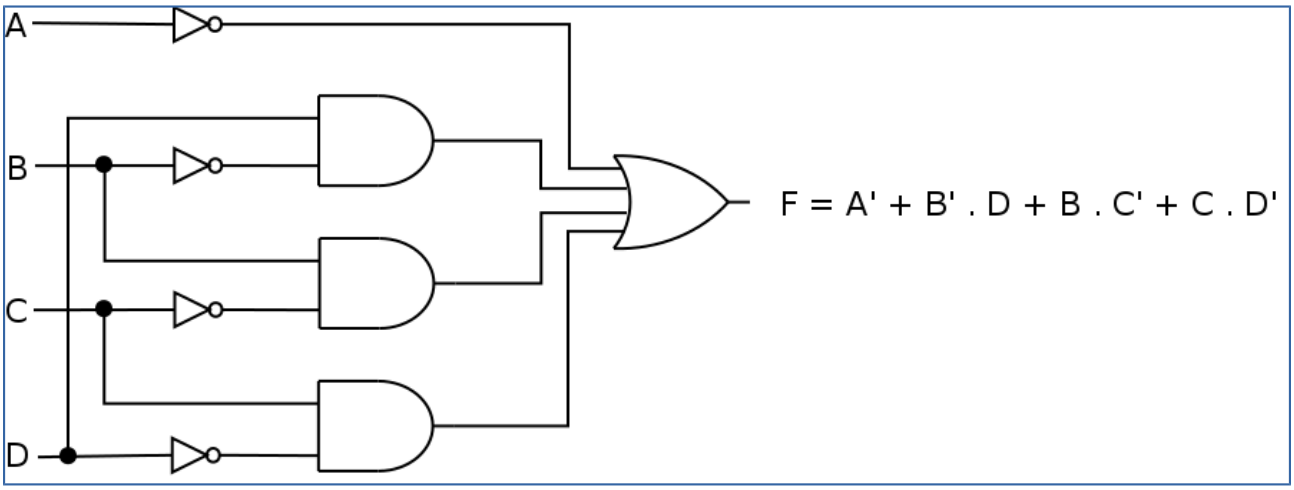
ΘΕΜΑΤΑ ΚΑΙ ΛΥΣΕΙΣ

1) Έστω η λογική συνάρτηση F = A' + B' ˙ D + B ˙ C' + C ˙ D' (όπου οι τόνοι δηλώνουν συμπληρώματα).

i) Σχεδιάστε το αντίστοιχο κύκλωμα.

ii) Δημιουργήστε τον πίνακα αλήθειας της συνάρτησης F.

ΛΥΣΗ



A	B	C	D	A'	B'	C'	D'	B' and D	B and C'	C and D'	F
0	0	0	0	1	1	1	1	0	0	0	1
0	0	0	1	1	1	1	0	1	0	0	1
0	0	1	0	1	1	0	1	0	0	1	1
0	0	1	1	1	1	0	0	1	0	0	1
0	1	0	0	1	0	1	1	0	1	0	1
0	1	0	1	1	0	1	0	0	1	0	1
0	1	1	0	1	0	0	1	0	0	1	1
0	1	1	1	1	0	0	0	0	0	0	1
1	0	0	0	0	1	1	1	0	0	0	0
1	0	0	1	0	1	1	0	1	0	0	1
1	0	1	0	0	1	0	1	0	0	1	1
1	0	1	1	0	1	0	0	1	0	0	1
1	1	0	0	0	0	1	1	0	1	0	1
1	1	0	1	0	0	1	0	0	1	0	1
1	1	1	0	0	0	0	1	0	0	1	1
1	1	1	1	0	0	0	0	0	0	0	0

2α) Να εκτελέσετε την αφαίρεση $(33)_{<10>}$ - $(73)_{<10>}$ στο δυαδικό σύστημα αρίθμησης, χρησιμοποιώντας τη μέθοδο του συμπληρώματος ως προς 2. Θεωρήστε ότι οι δυαδικοί αριθμοί αναπαριστώνται με 8 δυαδικά ψηφία (bits).

ΛΥΣΗ

Βάση		**Πηλίκο**		**Υπόλοιπο**	
33:2	=	16	+	1	
16:2	=	8	+	0	
8:2	=	4	+	0	
4:2	=	2	+	0	
2:2	=	1	+	0	
1:2	=	0	+	1	
					$1\,0\,0\,0\,0\,1_{<2>}$

Βάση		**Πηλίκο**		**Υπόλοιπο**	
73:2	=	36	+	1	
36:2	=	18	+	0	
18:2	=	9	+	0	
9:2	=	4	+	1	
4:2	=	2	+	0	
2:2	=	1	+	0	
1:2	=	0	+	1	
					$1\,0\,0\,1\,0\,0\,1_{<2>}$

Συμπλήρωμα ως προς 2 του $1001001_{<2>}$:

0$1001001_{<2>}$ → Συμπλήρωμα ως προς 1: $10110110_{<2>}$ → Συμπλήρωμα ως προς 2: $10110111_{<2>}$

Αρκεί να προσθέσουμε τον μειωτέο $00100001_{<2>}$ στο συμπλήρωμα ως προς 2 του αφαιρετέου.

$$\begin{array}{r} 00100001 \\ +\ \ \underline{10110111} \\ 11011000 \end{array}$$

Το αποτέλεσμα της αφαίρεσης είναι $11011000_{<2>}$.

2β) Να εκτελέσετε την πρόσθεση των αριθμών $(3F3)_{<16>}$ + $(7C)_{<16>}$ στο δεκαεξαδικό σύστημα αρίθμησης.

ΛΥΣΗ

σε μεταφορά: 11

$$\begin{array}{r} 3F3_{<16>} \\ +\ \ \underline{7C_{<16>}} \\ 46F_{<16>} \end{array}$$

3α) Ποιο θα είναι το αποτέλεσμα της εκτέλεσης των παρακάτω τριών γραμμών κώδικα;

```
st='abcdefghiklmn'
for i in range(len(st)):
   print 'char number %i is %s' %(i,st[i])
```

Γράψτε έναν βρόχο while που κάνει ακριβώς το ίδιο με τον παραπάνω βρόχο for.

ΛΥΣΗ

```
char number 0 is a
char number 1 is b
char number 2 is c
char number 3 is d
char number 4 is e
char number 5 is f
char number 6 is g
char number 7 is h
char number 8 is i
char number 9 is k
char number 10 is l
char number 11 is m
char number 12 is n

st='abcdefghiklmn'
i=0
while i<len(st):
   print 'char number %i is %s' %(i,st[i])
   i+=1
```

3β)Ποιο θα είναι το αποτέλεσμα της εκτέλεσης των παρακάτω γραμμών κώδικα;

```
u=0
while True:
   if u%3==0:
      print 'India'
   elif u-1<=0:
      print 'Brazil'
   elif u-2<=0:
      print 'China'
   elif u>=3:
      break
   u+=1
```

ΛΥΣΗ

```
India
Brazil
China
India
```

3c)def fn(n):

```
        if n==1: return 1
        elif (n%2==0): return fn(n/2)+2
        else: return fn(n-1)+3
```

Ποια θα είναι η τιμή της fn(5); Τεκμηριώστε την απάντησή σας

ΛΥΣΗ

fn(5)=fn(4)+3
fn(4)=fn(2)+2
fn(2)=fn(1)+1
fn(1)=1
άρα fn(5)=8

4α)Γράψτε πρόγραμμα σε γλώσσα προγραμματισμού python, το οποίο θα διαβάζει 20 ακέραιες τιμές, και θα εμφανίζει στην οθόνη το μεγαλύτερο αριθμό, τον μικρότερο και την μέση τιμή των αριθμών.

ΛΥΣΗ

```
import numpy
a=[]
for i in range(0,20,1):
        a.append(int(raw_input("Type value: ")))
print numpy.min(a),numpy.max(a),numpy.mean(a)
```

4b)Γράψτε πρόγραμμα σε γλώσσα python που θα σχεδιάζει τη γραφική παράστασης της συνάρτησης: f(x)= x*sin(x) στο διάστημα [0,2π].

ΛΥΣΗ

```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(0,2*np.pi,100)
y = np.sin(x)*x
plt.plot(x,y)
plt.show()
```

5α)Γράψτε μια συνάρτηση recEq(n) με παράμετρο έναν ακέραιο αριθμό n, η οποία (συνάρτηση) υπολογίζει (επιστρέφει) τη λίστα (ακολουθία) a_{k+1}= 2sin((2k+1)π/2)a_k, a_0 = 1, για k = 0, 1, …, n.

```
def recEq(n):
   import math
   a = []
   k = 0
   while k <= n:
      if k == 0:
         a.append(1)
      else:
         a.append(2*math.sin((2*k+1)*math.pi/2.) * a[-1])
      k += 1
   return a
```

5β)Γράψτε μια συνάρτηση charge που έχει τρεις παραμέτρους: value (τιμή χωρίς ΦΠΑ), fpa (ΦΠΑ), discount (έκπτωση) και επιστρέφει το ποσό πληρωμής μετά την έκπτωση. Η έκπτωση καθορίζεται ως κάποιο ποσοστό επί τοις εκατό, και εφαρμόζεται μόνο εφόσον η τιμή χωρίς ΦΠΑ είναι μεγαλύτερη από 10.

ΛΥΣΗ

```
def charge(amount, fpa, discount):
    if amount < 10:
        return amount* (1+fpa)
    else:
        return amount* (1+fpa)*(1-discount)
```

6)Γράψτε μια συνάρτηση three_ranking με δυο παραμέτρους: (1) candidates, που είναι μια λίστα υποψηφίων να εκλεγούν, και (2) voters, που είναι ένα λεξικό με κλειδιά κάποιους ψηφοφόρους (voters) και τιμές, για τον κάθε ψηφοφόρο, μια πλειάδα (tuple) αποτελούμενη από 3 υποψήφιους κατά σειρά προτίμησης. Π.χ., μια λίστα υποψήφιων μπορεί να είναι η candidates=['cand1','cand2','cand3','cand4','cand5','cand6'] και ένα λεξικό ψηφοφόρων μπορεί να είναι το voters={'voter1':('cand1','cand2','cand5'),'voter2':('cand3','cand4','cand6'),'voter3':('cand2','cand3','cand4')}. Η συνάρτηση three_ranking, για κάθε ψηφοφόρο, θα δίνει 3 ψήφους στον υποψήφιο που ο ψηφοφόρος αυτός τον επιλέγει πρώτο, 2 ψήφους στον υποψήφιο που τον επιλέγει δεύτερο και ψήφο στον υποψήφιο που τον επιλέγει τρίτο. Στο τέλος, η συνάρτηση three_ranking εκτυπώνει υποψήφιους μαζί με τους συνολικούς ψήφους που πήραν από την ψηφοφορία σε φθίνουσα σειρά συνολικών ψήφων.

ΛΥΣΗ

```
def three_ranking(candidates,voters):
    results={}
    for key,val in voters.items():
        u=3
        for candi in val:
            if candi not in results:
                results[candi]=u
            else:
                results[candi]+=u
            u-=1
    result={}
    for i,v in results.items():
        if v not in result:
            result[v]=[i]
        else:
            result[v].append(i)
    for i in sorted(result.keys(),reverse=True):
        for j in result[i]:
            print 'O %s phre %i psifous' %(j,results[j])
```