Περιεχόμενο του εργαστηρίου:

- Βασικοί λογικοί τελεστές `||`, `&&`, `~=`, `==`, `<`, `<=`, `>`, `>=`.
- Βρόγχοι ελέγχου `if`.

Βασικοί λογικοί τελεστές:

Οι διάφοροι βρόγχοι επανάληψης ή ελέγχου έχουν τη βασική δομή:

```
Εάν [μία λογική συνθήκη] είναι αληθής:
     [Κάνε αυτές τις εντολές]
Τέλος εντολών.
```

ή:

```
Όσο [μία λογική συνθήκη] είναι αληθής:
     [Κάνε αυτές τις εντολές]
Τέλος εντολών.
```

Είναι λοιπόν αναγκαίο να μπορούμε να εκφράσουμε στη γλώσσα του υπολογιστη λογικές συνθήκες. Στη συνέχεια θα αναφερθούμε σε λογικούς τελεστές με τους οποίους μπορούμε να δομούμε λογικές συνθήκες.

- Είναι ίσο: Χρησιμοποιείται το `==`

  ```
  1==1
  ans = 1
  2==1
  ans = 0
  ```

  (Τις γραμμές 1 και 3 γράφουμε εμείς, ενώ τις 2 και 4 εμφανίζει το Matlab).

  Προσέξτε εδώ ότι δεν χρησιμοποιείται το =, καθώς έχει δεσμευτεί για την εκχώρηση τιμής. Χρησιμοποιείται το ==. Τον συγκεκριμένο τελεστή μπορούμε να τον φανταζόμαστε ως ερώτηση προς τον υπολογιστή: Ρωτάμε αν δύο αντικείμενα είναι ίδια, και ο υπολογιστής απαντάει `1` εάν είναι ίδια και `0` εάν δεν είναι. Το `1` μπορεί να αντικατασταθεί από το `true` και το `0` από το `false`.
- Άρνηση: Χρησιμοποιείται το `~`

  ```
  ~(1==1)
  ans = 0
  ~(2==1)
  ```

Κουκουδάκης Νίκος (nicolaskoukoudakis@gmail.com)
Φράγκος Τάσος (afragos@post.com)

```
ans = 1
```

(Τις γραμμές 1 και 3 γράφουμε εμείς, ενώ τις 2 και 4 εμφανίζει το Matlab).

Στην ουσία η ~ εναλλάσσει τις λογικές τιμές μίας λογικής συνθήκης. Όταν `P==true` (δηλαδή `P==1`) έχουμε `~P==false` (δηλαδή `~P==0`), ενώ όταν `P==false` (δηλαδή `P==0`) έχουμε `~P==true` (δηλαδή `~P==1`).

- Δεν είναι ίσο: Χρησιμοποιείται το ~=

```
1~=1
ans = 0
2~=1
ans = 1
```

(Τις γραμμές 1 και 3 γράφουμε εμείς, ενώ τις 2 και 4 εμφανίζει το Matlab).

Παρατηρήστε ότι `~(1==1)==(1~=1)` και `~(2==1)==(2~=1)`.

- Ή το ένα ή το άλλο ή και τα δύο: Χρησιμοποιείται το ||

```
(1==2)||(12==4)
ans = 0
(1==1)||(12==4)
ans = 1
(1==2)||(12==12)
ans = 1
(1==1)||(12==12)
ans = 1
```

(Τις γραμμές 1, 3, 5 και 7 γράφουμε εμείς, ενώ τις 2, 4, 6 και 8 εμφανίζει το Matlab).

Εάν `P` και `Q` είναι δύο λογικές συνθήκες, η `P||Q` αληθεύει εάν και μόνο αν κάποια από τις `P`, `Q` αληθεύει.

- Και: Χρησιμοποιείται το &&

```
(1==2)&&(12=4)
ans = 0
(1==1)&&(12=4)
ans = 0
(1==2)&&(12==12)
ans = 0
```

Κουκουδάκης Νίκος (nicolaskoukoudakis@gmail.com)
Φράγκος Τάσος (afragos@post.com)

```
(1==1)&&(12==12)
ans = 1
```

(Τις γραμμές 1, 3, 5 και 7 γράφουμε εμείς, ενώ τις 2, 4, 6 και 8 εμφανίζει το Matlab).

Εάν `P` και `Q` είναι δύο λογικές συνθήκες, η `P&&Q` αληθεύει μόνο αν και οι δύο `P`, `Q` αληθεύουν.

Επίσης, θα αναφέρουμε χωρίς παραδείγματα τις σχέσεις διάταξης.

- Είναι μικρότερο: Χρησιμοποιείται το <
- Είναι μεγαλύτερο: Χρησιμοποιείται το >
- Είναι μικρότερο ή ίσο: Χρησιμοποιείται το <=
- Είναι μεγαλύτερο ή ίσο: Χρησιμοποιείται το >=

Άσκηση 1: Ελέγξτε με το Matlab τις τιμές των λογικών συνθηκών:

1) `(1~=1)&&((2>-3)||((5<3)&&(1==1)))`
2) `(1~=1)||(5==5)-1` (παρατηρήστε ότι μπορείτε να κάνετε πράξεις με λογικές τιμές, αφού αντιμετωπίζονται ως `0` και `1`).

Άσκηση 2: Φτιάξτε ένα πρόγραμμα στο Matlab που θα δέχεται δύο αριθμούς στο σύνολο `{0,1}`, και θα υπολογίζει μία λογική πρόταση με τιμές αληθείας:

Είσοδος 1	Είσοδος 2	Έξοδος
0	0	0
0	1	1
1	0	1
1	1	0

[Σημείωση: Θα πρέπει να κατασκευάσετε τη λογική πρόταση, χρησιμοποιώντας τα ==, ,~, ~=, `&&`, `||`].

[Υπόδειξη: Μελετήστε τον πίνακα αληθείας του `||`].

Άσκηση 3: Φτιάξτε ένα πρόγραμμα στο Matlab που θα δέχεται δύο αριθμούς στο σύνολο `{0,1}`, και θα υπολογίζει μία λογική πρόταση με τιμές αληθείας:

Είσοδος 1	Είσοδος 2	Έξοδος
0	0	1
0	1	0
1	0	0
1	1	1

[Σημείωση: Θα πρέπει να κατασκευάσετε τη λογική πρόταση, χρησιμοποιώντας τα ==, ,~, ~=, `&&`, `||`].

Κουκουδάκης Νίκος (nicolaskoukoudakis@gmail.com)
Φράγκος Τάσος (afragos@post.com)

Άσκηση [επιπλέον, για όποιον θέλει]: Στην ηλεκτρική μηχανική φτιάχνονται λογικές μηχανές χρησιμοποιώντας τα ακόλουθα αντικείμενα (που ονομάζονται «λογικές πύλες», ή στα Αγγλικά, «logic gates»). Με `0` θα συμβολίζουμε την κατάσταση στην οποία δεν περνάει ρεύμα και με `1` την κατάσταση στην οποία περνάει.

AND (δηλαδή «και»): Λειτουργεί όπως το `&&`. Μόνο όταν και οι δύο είσοδοι αριστερά δέχονται ρεύμα (δηλαδή είναι σε κατάσταση `1 1`) η συσκευή δίνει ρεύμα στην έξοδο.

Δηλαδή:

Είσοδος 1	Είσοδος 2	Έξοδος AND
0	0	0
0	1	0
1	0	0
1	1	1

OR (δηλαδή «ή το ένα ή το άλλο ή και τα δύο»): Λειτουργεί όπως το `||`. Όταν έστω και μία από τις εισόδους αριστερά δέχεται ρεύμα, η συσκευή δίνει ρεύμα στην έξοδο.

Δηλαδή:

Είσοδος 1	Είσοδος 2	Έξοδος OR
0	0	0
0	1	1
1	0	1
1	1	1

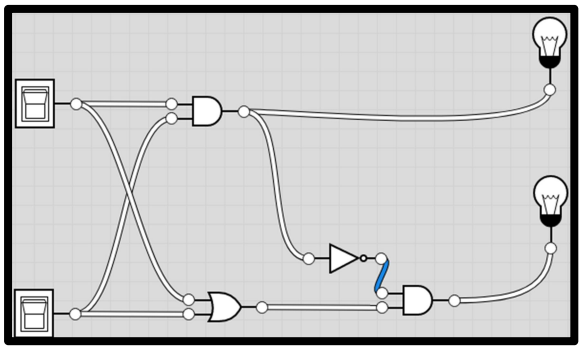
NOT (δηλαδή «άρνηση»): Λειτουργεί όπως το `~`. Όταν υπάρχει ρεύμα στην είσοδο αριστερά, στην έξοδο δεν υπάρχει. Αντίθετα, εάν δεν υπάρχει ρεύμα στην είσοδο, στην έξοδο υπάρχει.

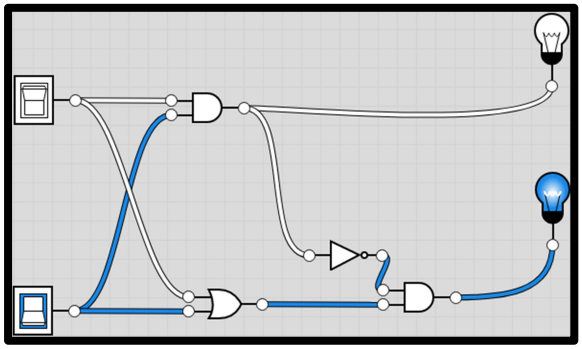
Δηλαδή:

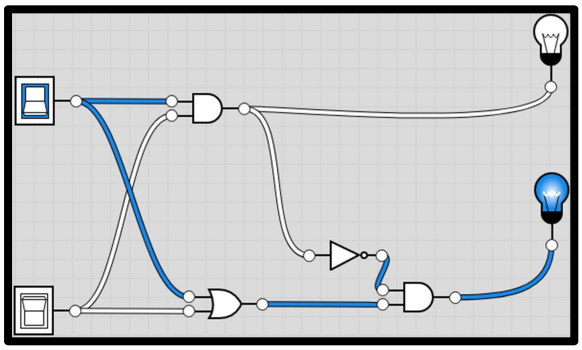
Είσοδος	Έξοδος NOT
0	1
1	0

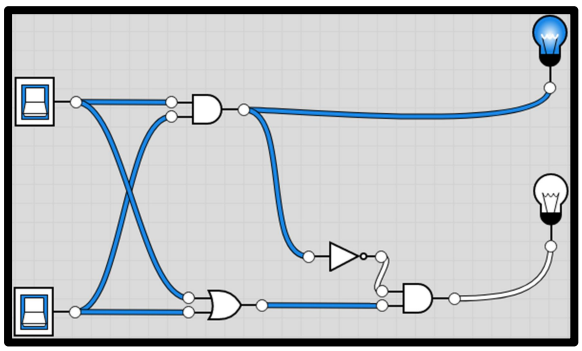
Παρακάτω έχουμε σχεδιάσει ένα κύκλωμα-αριθμομηχανή, το οποίο μπορεί να κάνει τις πράξεις `0+0`, `1+0`, `0+1`, `1+1`, και δίνει αποτέλεσμα σε δυαδική μορφή.

Κουκουδάκης Νίκος (nicolaskoukoudakis@gmail.com)
Φράγκος Τάσος (afragos@post.com)









Οι διακόπτες στ' αριστερά δουλεύουν ως είσοδοι και οι λάμπες ως έξοδοι. Όταν κάθε διακόπτης προσθέτει `1`. Η κάτω λάμπα αντιπροσωπεύει το 2^0=1 και η πάνω το 2^1=2.

Δίπλα απεικονίζονται οι καταστάσεις όπου ένας διακόπτης ή και οι δύο έχουν ενεργοποιηθεί.

Ερώτηση: Εάν μεταφράζαμε το κύκλωμα αυτό σε λογικές πρότασεις στο Matlab, τι μορφή θα είχαν;

Αυτό που θα χρειαστεί να φτιάξετε, με τη βοήθεια των συμβόλων `&&`, `||`, `~`, είναι δύο μαθηματικές προτάσεις που θα δείχνουν πότε κάθε λάμπα είναι αναμμένη (καθεμία θα έχει έξοδο `0` αν δεν είναι αναμμένη, `1` αν είναι αναμμένη).

Δηλαδή, αν `s1`, `s2` είναι οι δύο διακόπτες, πρέπει να φτιαχθούν:

```
out1(s1, s2)
```

```
out2(s1, s2)
```

για την κάτω λάμπα και την πάνω λάμπα αντίστοιχα.

Λύση: Ακολουθώντας το διάγραμμα έχουμε ότι:

```
out1 = (s1||s2)&&(~(s1&&s2))
```

και:

```
out2 = s1&&s2
```

Τώρα, αν θέλουμε να φτιάξουμε ένα (άχρηστο) πρόγραμμα που υπολογίζει όπως παραπάνω τα αθροίσματα `0+0`, `0+1`, `1+0`, `1+1`, μπορούμε να γράψουμε:

Κουκουδάκης Νίκος (nicolaskoukoudakis@gmail.com)
Φράγκος Τάσος (afragos@post.com)

```
s1=input('Give me the first input: ');
s2=input('Give me the second input: ');
out1=(s1||s2)&&(~(s1&&s2));
out2=s1&&s2;
fprintf('The sum in binary notation is %1.0f%1.0f',
out2, out1)
```

Έχοντας κάνει αυτήν την εισαγωγή στις λογικές συνθήκες, μπορούμε να προχωρήσουμε στους βρόγχους `if`.

Βρόγχοι `if`:

- Βρόγχοι `if`: Με την εντολή `if`[κάποια λογική συνθήκη] μπορούμε να εκτελέσουμε εντολές μόνο σε συγκεκριμένες περιπτώσεις, όταν η λογική συνθήκη είναι αληθής. Για παράδειγμα, το παρακάτω θα εκτελεστεί μόνο όταν `x>=0`:

  ```
  if x>=0
       disp('This displays only when x>=0')
  endif
  ```

  και θα εμφανίσει, μόνο όταν `x>=0`:

  ```
  This displays only when x>=0
  ```

  Προσοχή: Για να δουλέψουν τα παραπάνω, θα πρέπει η τιμή του x να έχει κάπως προσδιοριστεί (από εμάς ή από το πρόγραμμα).
- `elseif`: Ενδέχεται σε δύο διαφορετικά ενδεχόμενα να θέλουμε να εκτελέσουμε δύο διαφορετικά είδη εντολών. Αυτό μπορεί να επιτευχθεί μέσω του `if`, με την επιπλέον εντολή `elseif`.

  ```
  if x>=0
       disp('This displays only when x>=0')
  elseif x==-1
       disp('This displays only when x==-1')
  endif
  ```

  Το παραπάνω πρόγραμμα θα εμφανίσει:

  ```
  This displays only when x>=0
  ```

  μόνο όταν `x>=0`. Εάν `x==-1`, θα εμφανιστεί:

  ```
  This displays only when x==-1
  ```

Κουκουδάκης Νίκος (nicolaskoukoudakis@gmail.com)
Φράγκος Τάσος (afragos@post.com)

Για τις υπόλοιπες περιπτώσεις, το πρόγραμμα δεν θα εμφανίσει τίποτα.

Προσοχή: Για να δουλέψουν τα παραπάνω, θα πρέπει η τιμή του x να έχει κάπως προσδιοριστεί (από εμάς ή από το πρόγραμμα).

- `else`: Με την επιπλέον εντολή `else` στο `if` μπορούμε να συμπεριλάβουμε όσες περιπτώσεις απέμειναν. Για παράδειγμα:

```
if x>0
     disp('This displays only when x>0')
elseif x<0
     disp('This displays only when x<0')
else
     disp('This displays only when x==0')
endif
```

Ένα άλλο παράδειγμα (χωρίς `elseif`):

```
if x>=0
     disp('This displays only when x>=0')
else
     disp('This displays only when x<0')
endif
```

Προσοχή: Για να δουλέψουν τα παραπάνω, θα πρέπει η τιμή του x να έχει κάπως προσδιοριστεί (από εμάς ή από το πρόγραμμα).

Προσοχή: Από την στιγμή που το `if` εκτελεστεί, κανένα `elseif` ή `else` δεν εκτελείται. Αντίστοιχα, αν κάποιο `elseif` εκτελεστεί, κανένα επόμενο `elseif` ή `else` δεν εκτελείται.

Άσκηση 4: Φτιάξτε ένα πρόγραμμα-ευρετήριο, το οποίο θα ζητά από τον χρήστη ποιο στοιχείο του παρακάτω προσώπου χρειάζεται να εμφανιστεί, και θα το εμφανίζει.

Ονομετεπώνυμο: Allan Turing,

Ημ/νία γέννησης: 23/06/1912

Τηλέφωνο: 10001001

Ηλ. ταχυδρομείο: itsallanturing@themail.uk

Διεύθυνση κατοικίας: Maida Vale, London, England

Επισκόπηση του προγράμματος:

Κουκουδάκης Νίκος (nicolaskoukoudakis@gmail.com)
Φράγκος Τάσος (afragos@post.com)

```
Type 1 for name,
Type 2 for date of birth,
Type 3 for telephone number,
Type 4 for email,
Type 5 for residence: 1
Name: Allan Turing.
```

(Η μόνη πληροφορία που δίνουμε προκειμένου να πάρουμε μία έξοδο είναι ένας αριθμός στο σύνολο `{1,2,3,4,5}`*).*

Άσκηση 5: Φτιάξτε ένα πρόγραμμα το οποίο θα δέχεται έναν φυσικό αριθμό και θα επιστρέφει εάν αυτός διαιρείται ή όχι με τους αριθμούς 2,4,8,16 ή 5.

Επισκόπιση του προγράμματος:

```
Give me a natural number: 40
40 is divisible by 2
40 is divisible by 4
40 is divisible by 8
40 is not divisible by 16
40 is divisible by 5
```

(Η μόνη πληροφορία που δίνουμε προκειμένου να πάρουμε μία έξοδο είναι ένας αριθμός στο σύνολο $\mathbb{N}$*).*

[Υπόδειξη: Φτιάξτε δύο `if`, ένα για τα πολλαπλάσια του `2` κι ένα για το `5`].

Κουκουδάκης Νίκος (nicolaskoukoudakis@gmail.com)
Φράγκος Τάσος (afragos@post.com)