ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

ΠΛΗΡΟΦΟΡΙΚΗ Ι

Ενότητα 3: Δομές επανάληψης

Μιχάλης Δρακόπουλος

Σχολή Θετικών επιστημών

Τμήμα Μαθηματικών

ΠΛΗΡΟΦΟΡΙΚΗ Ι (MATLAB)
Ενότητα 3

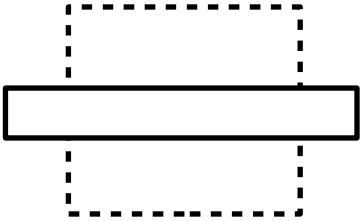
Σημειώσεις βασισμένες στο βιβλίο "Το MATLAB στην Υπολογιστική Επιστήμη και Τεχνολογία – Μια Εισαγωγή"

ΔΟΜΕΣ ΕΠΑΝΑΛΗΨΗΣ

Πρόβλημα: Για δεδομένο αριθμό Α, μα βρεθεί η $\sqrt{A}$.
Γεωμετρική αναδιατύπωση: Για θετικό αριθμό Α, να βρεθεί τετράγωνο με εμβαδόν Α.

- Μια αρχική ιδέα:

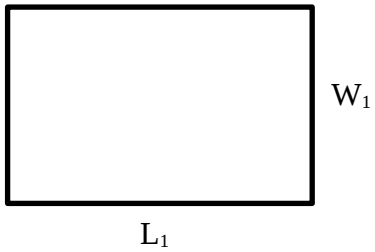
- Παρατήρηση:



Η απάντηση είναι μεταξύ του L και του W:
W < S < L

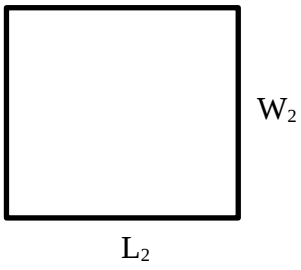
- Βασική ιδέα:

$$L_1 = \frac{L+W}{2}$$
$$W_1 = \frac{A}{L_1}$$



επανάληψη:

$$L_2 = \frac{L_1+W_1}{2}$$
$$W_2 = \frac{A}{L_2}$$



Ένα πρώτο script:

```
A = input('A:');
L0 = A; W0 = A/L0;
L1 = (L0+W0)/2; W1 = A/L1;
L2 = (L1+W1)/2; W2 = A/L2;
L3 = (L2+W2)/2; W3 = A/L3;
L4 = (L3+W3)/2; W4 = A/L4;
```

Δεν υπάρχει λόγος να δημιουργηθούν όλες αυτές οι μεταβλητές:

```
A = input('A:');
L = A; W = A/L;
L = (L+W)/2; W = A/L;
L = (L+W)/2; W = A/L;
L = (L+W)/2; W = A/L;
L = (L+W)/2; W = A/L;
```

Άρα: ανάγκη δομής που επαναλαμβάνει κώδικα για συγκεκριμένο αριθμό επαναλήψεων

→ Ο βρόχος **for**

```
for <τιμές μετρητή επανάληψης>
   εντολές που θα επαναληφθούν;
end
```

Συνήθης μορφή **for**:

```
for <μετ/τη> = <A.T.>:[Βήμα]:<T.T.>
   <εντολές>
end
```

Σημειολογία: < > : υποχρεωτικό μέρος
[ ] : προαιρετικό μέρος

Όταν το βήμα είναι μοναδιαίο:

```
for <μετ/τη> = <A.T.>:<T.T.>
     <εντολές>
end
```

Άρα το προηγούμενο παράδειγμα:

```
A = input('A:');
L = A; W = A/L;
for k = 1:4
   L = (L+W)/2;
   W = A/L;
end
```

και πιο γενικά

```
A = input('A:');
nSteps = input('nSteps:');
L = A; W = A/L;
for k = 1:nSteps;
   L = (L+W)/2;
   W = A/L;
end
```

Άλλα παραδείγματα:

```
for i=1:10
   disp(i);
end
```

```
for odd = 1:2:100
   disp(odd);
end
```

```
for even = 2:2:100
   disp(even);
end
```

```
for countdown = 10:-1:0
   disp(countdown);
end
```

```
for i=1:10
   x=rand;
   fprintf('%.3f\n',x);
end
```

Υπολογισμός αθροίσματος: 1+2+3+…+N:	Υπολογισμός γινομένου: 1*2*3*…*N:

`sum = 0;`
`for i=1:N`
`  sum = sum+i;`
`end`
`disp(sum);`
	`prod = 1;`
	`for i=1:N`
	`  prod = prod*i;`
	`end`
	`disp(prod);`

Πρόβλημα: Ένα κλαδί μοναδιαίου μήκους κόβεται σε τυχαίο σημεία. Τι μήκος θα έχει κατά μέσο όρο το μικρότερο κομμάτι;

Προσομοίωση: *η αναπαράσταση ενός πειράματος ή μιας φυσικής διεργασίας με τη χρήση ενός προγράμματος (μοντέλο).*

```
% μία δοκιμή του πειράματος
breakPt = rand;
if breakPt < 0.5
   shortPiece = breakPt;
else
   shortPiece = 1-breakPt;
end
```

```
breakPt = rand;
shortPiece = min(breakPt, 1-breakPt);
```

Αλγόριθμος λύσης του προβλήματος:

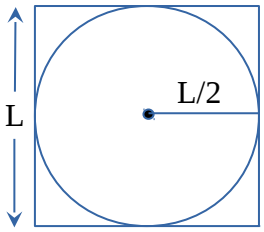
- Επανάληψη του πειράματος *n* φορές
- Υπολογισμός μέσου όρου
- Εμφάνιση αποτελέσματος

```
n = 10000; % αριθμός δοκιμών
total = 0; % τρέχον άθροισμα για το μήκος
for i=1:n
   breakPt = rand;
   shortPiece = min(breakPt, 1-breakPt);
   total = total+shortPiece;
end
avgLength = total/n;
fprintf('Το μέσο μήκος είναι %.2f\n', avgLength);
```

Παράδειγμα: μ.ο. 10 αριθμών από τον χρήστη:

```
n=10; total=0;
for k=1:n
   num = input('Δώσε έναν αριθμό:');
   total = total+num;
end
avg = total/n;
fprintf('Μέσος όρος: %.3%f\n', avg);
```

Πρόβλημα: Προσέγγιση του π με τη μέθοδο Monte Carlo



→ Ρίχνουμε *n* βελάκια στο τετράγωνο

→ hit: το βελάκι μέσα στον κύκλο

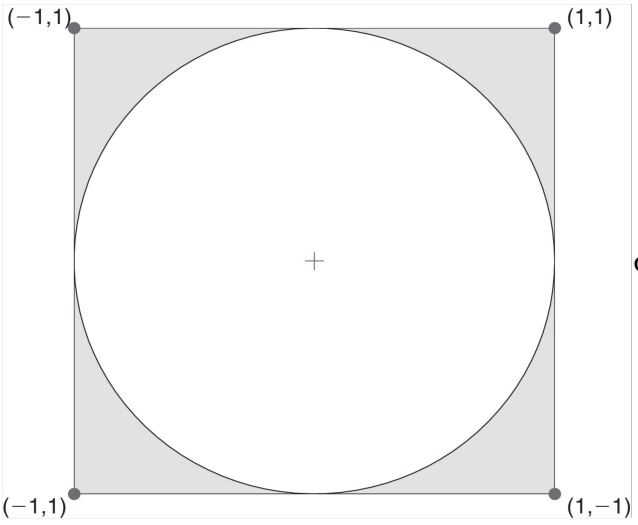
→ Το ποσοστό των hits προσεγγίζει τον λόγο των δύο εμβαδών:

$$\frac{hits}{n} \simeq \frac{E_{κύκλου}}{E_{τετραγώνου}} = \frac{\pi \frac{L^2}{4}}{L^2} \Rightarrow \pi \simeq 4\frac{hits}{n}$$

Άρα:

```
for i=1:n
   % ρίχνω το βελάκι i
   % αν το βελάκι είναι μέσα στον κύκλο:
   % hits = hits+1
end
myPi = 4*hits/n;
```

Ας υποθέσουμε ότι έχουμε τον μοναδιαίο κύκλο με κέντρο (0,0):



→ Ένα σημείο (x,y) είναι hit εάν $x^2+y^2 \le 1$

→ Πρέπει να βρεθεί τρόπος να παράγουμε τυχαία σημεία στο τετράγωνο.

Θέλουμε: -1<x<1 και -1<y<1, δηλαδή εύρος = 2 και διάστημα (-1, 1).

Η `rand` δίνει τυχαίους αριθμούς στο διάστημα (0, 1) (εύρος = 1).

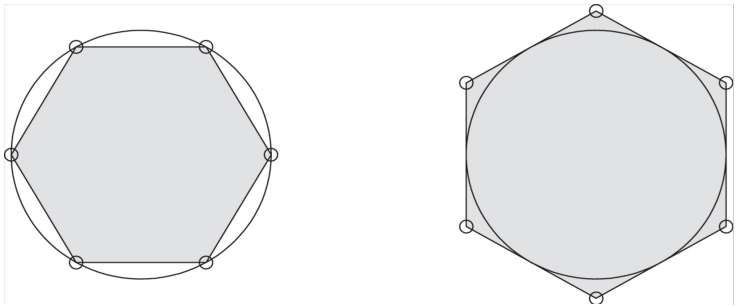
Με `rand` x 2 → εύρος 2, αλλά στο διάστημα (0, 2).

Άρα: `2*rand-1` → τυχαίοι αριθμοί στο (-1, 1).

To script:

```
n = 10000;
hits = 0;
for i=1:n
   %ρίχνω το βελάκι i
   x = 2*rand-1;
   y = 2*rand-1;
   % ελέγχω αν είναι hit
   if x^2 + y^2 <= 1
      hits = hits+1;
   end
end
myPi = 4*hits/n;
```

Πρόβλημα: Έστω *n* σημεία πάνω στον μοναδιαίο κύκλο.



$A_n = \frac{n}{2} \cdot \eta\mu\left(\frac{2\pi}{n}\right)$ $B_n = n \cdot \varepsilon\varphi\left(\frac{\pi}{n}\right)$

Όσο αυξάνεται το *n* (τα σημεία), το A_n αυξάνεται και το B_n μειώνεται και προσεγγίζουν το Ε του κύκλου, δηλαδή το π. Άρα, ο μέσος όρος τους αποτελεί μια καλή προσέγγιση του π:

$\pi \simeq \frac{A_n + B_n}{2}$, με το σφάλμα να εξαρτάται από την απόλυτη διαφορά των δύο εμβαδών.

Μας ενδιαφέρει ποια είναι η μικρότερη τιμή του *n* (n_*) για την οποία ικανοποιείται η σχέση:

$\left|A_{n_*} - B_{n_*}\right| \le \delta$, όπου δ η ανοχή σφάλματος π.χ. δ=0.00001.

<u>Αλγόριθμος:</u>

Δώσε την ανοχή σφάλματος δ και θέσε n=3.

Υπολόγισε τα A_3, B_3 και το όριο του σφάλματος B_3-A_3

Όσο το όριο του σφάλματος > δ, επανέλαβε:

Αύξησε το *n* κατά 1

Ενημέρωσε τα A_n και B_n και το όριο του σφάλματος.

Θέσε $n_* = n$ και εμφάνισε το $\frac{A_{n_*} + B_{n_*}}{2}$ (προσέγγιση του π).

Ο αλγόριθμος αυτός δεν μπορεί να υλοποιηθεί με `for` γιατί δεν είναι γνωστός εκ των προτέρων ο αριθμός των επαναλήψεων. Για τέτοιες επαναλήψεις υπάρχει ο βρόχος `while`.

→ Ο βρόχος **`while`**

```
while <λογική συνθήκη>
   <εντολές>
end
```

Σημαντικό: Πρέπει κάποια από τις εντολές να μεταβάλλει τη συνθήκη ώστε κάποτε να γίνεται false, διαφορετικά: ατέρμονας βρόχος.

To script:

```
delta = input('Δώσε το δ:');
n = 3;
A_n = (n/2)*sin(2*pi/n);
B_n = n*tan(pi/n);
errorBound = B_n - A_n;
while errorBound>delta
   n = n+1;
   A_n = (n/2)*sin(2*pi/n);
   B_n = n*tan(pi/n);
   errorBound = B_n - A_n;
end
nStar = n;
myPi = (A_n + B_n)/2;
fprintf(......);
```

Άλλα παραδείγματα:

- Άθροισμα άγνωστων πλήθους θετικών αριθμών:

```
sum = 0;
x = input('Δώσε θετικό αριθμό:');
while x>0
   sum = sum+x;
   x = input('Δώσε θετικό αριθμό ή μη-θετικό αριθμό για τερματισμό');
end
disp(sum);
```

→ Αν θέλαμε να εμφανίσουμε και πόσοι αριθμοί δόθηκαν;

- Άθροισμα ψηφίων ακεραίου:

Για τον ακέραιο *n*:

- Εύρεση τελευταίου ψηφίου: `rem(n,10)`
  π.χ. `rem(1975,10)` → `5`
- Αποκοπή τελευταίου ψηφίου: `fix(n/10)`
  π.χ. `fix(1975/10)` → `197`

```
n = input('Δώσε ακέραιο:');
dsum = 0;
while n>0   % αν βάζαμε n>=0: ατέρμονες επαναλήψεις
   dsum  =dsum+rem(n,10);
   n = fix(n/10);
end
fprintf('Το άθροισμα των ψηφίων του %g είναι %g\n', n,dsum);
```

Σχέση `for` και `while`:

`for μετ/τη = AT:B:TT`
`  εντολές;`
`end`
	`μετ/τη = AT;`
	`  while μετ/τη <= TT`
	`  εντολές;`
	`  μετ/τη = μετ/τη + B;`
	`end`

- Το `while` μπορεί να χρησιμοποιηθεί πάντα.
- Το `for` προτιμάται όταν το πλήθος των επαναλήψεων είναι γνωστό.
- Όταν το πλήθος των επαναλήψεων είναι άγνωστο: `while`

Π.χ., χρόνια για διπλασιασμό κεφαλαίου 1000 ευρώ με ετήσιο επιτόκιο 5%.

```
money = 1000;
years = 0;
while money < 2000
   money = money * 1.05;
   years = years + 1;
end
disp(years);
```

Πιο γενικό:

```
money = input('Κεφάλαιο?');
years = 0;
goal = 2*money;
while money < goal
  …
…
```

Η εντολή `break`

Π.χ. υπολογισμός γινομένου αγνώστου πλήθους αριθμών:

Χωρίς `break`

```
endOfData=0;
prod=1;
x=input('Δώσε αριθμό:');
while x~=endOfData
   prod=prod*x;
   x=input('Δώσε αριθμό ή...
         0 για τερματισμό');
end
disp(prod);
```

Με `break`

```
endOfData=0;
prod=1;
while true
   x=input('Δώσε αριθμό...
       ή 0 για τερματισμό');
   if x==endOfData
      break;
   end
   prod=prod*x;
end
disp(prod);
```

Η `break` τερματίζει τις επαναλήψεις `for` ή `while` που το εκτελούν.

Ένθετα `for` (nested)

```
N = input('N?');
for i=1:N
   for j=1:N
      if i<j
         fprintf('.');
      else
         fprintf('*');
      end
   end
fprintf('\n');
end
```

π.χ. για N=5:

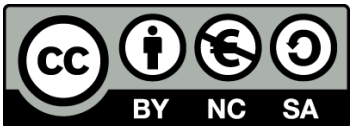
```
 *   *   *   *   *

 .   *   *   *   *

 .   .   *   *   *

 .   .   .   *   *

 .   .   .   .   *
```

Σημειώματα

Σημείωμα Αναφοράς

Μιχάλης Δρακόπουλος. «Πληροφορική Ι. Ενότητα 3: Δομές επανάληψης». Έκδοση: 1.0. Αθήνα 2014. Διαθέσιμο από τη δικτυακή διεύθυνση: http://opencourses.uoa.gr/modules/document/?course=MATH105.

Σημείωμα Αδειοδότησης



[1] http://creativecommons.org/licenses/by-nc-sa/4.0/

Ως **Μη Εμπορική** ορίζεται η χρήση:

- που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
- που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
- που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

Διατήρηση Σημειωμάτων

- Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:
- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.