ΑΣΚΗΣΗ 1

ΑΣΚΗΣΕΙΣ ΕΞΟΙΚΕΙΩΣΗΣ ΣΤΟ ΠΕΡΙΒΑΛΛΟΝ MATLAB

Αντικείμενο της άσκησης:

Η άσκηση αυτή αποσκοπεί στο να εξοικειώσει τους φοιτητές στην χρήση του περιβάλλοντος matlab με την βοήθεια χαρακτηριστικών παραδειγμάτων από τον προγραμματισμό.

1. Άσκηση #1

α) Εκφώνηση

Διαθέτουμε 40 νομίσματα των 50 λεπτών, 40 νομίσματα του 1€, 40 νομίσματα των 2€, και 40 χαρτονομίσματα των 5€. Να αναπτύξετε πρόγραμμα στο MATLAB το οποίο:

i) Θα υπολογίζει και θα εμφανίζει όλους τους δυνατούς συνδυασμούς με τους οποίους μπορούμε να χρησιμοποιήσουμε 40 συνολικά νομίσματα-χαρτονομίσματα (ανεξαρτήτως αξίας) έτσι ώστε η συνολική τους αξία να είναι 40€.

ii) Θα υπολογίζει και θα εμφανίζει το πλήθος των δυνατών συνδυασμών.

β) Ενδεικτική λύση

Για την αλγοριθμική επίλυση του ανωτέρω παραδείγματος απεικονίζουμε με τις ακόλουθες μεταβλητές τα εξής στοιχεία του προβλήματος:

x το πλήθος των νομισμάτων με αξία 0.5 ευρώ. Αυτή η τιμή μπορεί να λάβει τιμές **από 0 έως 40.**

y το πλήθος των νομισμάτων με αξία 1.0 ευρώ. *ομοίως*

z το πλήθος των νομισμάτων με αξία 2.0 ευρώ. *ομοίως*

w το πλήθος των χαρτονομισμάτων με αξία 5 ευρώ. *ομοίως*

Από ότι φαίνεται, με την χρήση βρόχων *for* μπορούμε να αναπαραστήσουμε τις δυνητικές τιμές του πλήθους των νομισμάτων. Για παράδειγμα, η εντολή *for x=0:40* δηλώνει ότι η

μεταβλητή x (*το πλήθος των νομισμάτων με αξία 0.5 ευρώ*) μπορεί να είναι από 0 (δηλαδή καθόλου νομίσματα με αξία 0.5 ευρω) μέχρι 40 (όλα τα νομίσματα με αξία 0.5 ευρώ). Συνεχίζοντας, επιθυμούμε να υπολογίσουμε το πλήθος των νομισμάτων ώστε να έχουμε 40 **συνολικά** νομίσματα-χαρτονομίσματα. Η προφανής έκφραση για το **συνολικό** πλήθος των νομισμάτων είναι: x+y+z+w. Επίσης η προφανής έκφραση για την αναπαράσταση της **αξίας** του παραπάνω **συνολικού** πλήθους είναι: x*0.5 +y*1+z*2+w*5. Ο κώδικας που ακολουθεί δίνει την προτεινόμενη λύση:

```
clear all , clc, close all,
% x το πλήθος των νομισμάτων με αξία 0.5 ευρώ.
% y το πλήθος των νομισμάτων με αξία 1.0 ευρώ.
% z το πλήθος των νομισμάτων με αξία 2.0 ευρώ.
% w το πλήθος των χαρτονομισμάτων με αξία 5 ευρώ.
% Αρχικοποίηση του πλήθους των συνδυασμών που ικανοποιούν τις συνθήκες
PS=0;
for x=0:40
    for y=0:40
        for z=0:40
            for w=0:40
                syn_plithos_nom=x+y+z+w;
                syn_aksia_nom=x*0.5+y*1+z*2+w*5;
                if syn_plithos_nom==40 && syn_aksia_nom==40
                    PS=PS+1;
disp(['Combination #' num2str(PS) ': ' num2str(x) 'x0.5E ' num2str(y)
'x1E ' num2str(z) 'x2E ' num2str(w) 'x5E '])
                end
            end
        end
    end
end
PS
```

Αποθηκεύσατε το ανωτέρω αρχείο με όνομα ***ask1_1.m***. Στην συνέχεια εκτελέστε το με τους ακόλουθους τρόπους:

i) Κλήση του ονόματός του από την γραμμή εντολών. **>> *ask1_1***

ii) Αντιγράψτε όλες τις εντολές του αρχείου και επικολλήστε τις στο παράθυρο εντολών του matlab.

γ) Υλοποίηση της άσκησης με συνάρτηση (function) του matlab :

Στην περίπτωση που θέλουμε να υλοποιήσουμε την άσκηση με χρήση συνάρτησης θα πρέπει να δούμε τις πιθανές εισόδους και τις εξόδους αυτής. Για παράδειγμα, θα μπορούσαμε να δώσουμε ως είσοδο τις αξίες των 4 νομισμάτων-χαρτονομισμάτων. Έτσι ένας πιθανός ορισμός - κλήση της συνάρτησης θα ήταν:

```
function PS=ask1_1f(val1, val2, val3, val4)
PS=0;
for x=0:40
    for y=0:40
        for z=0:40
            for w=0:40
                syn_plithos_nom=x+y+z+w;
                syn_aksia_nom=x*val1+y*val2+z*val3+w*val4;
                if syn_plithos_nom==40 && syn_aksia_nom==40
                    PS=PS+1;
                    disp(['Combination #' num2str(PS) ': ' num2str(x)
'x0.5E ' num2str(y) 'x1E ' num2str(z) 'x2E ' num2str(w) 'x5E '])
                end
            end
        end
    end
end
```

Αποθηκεύουμε το ανωτέρω πρόγραμμα με όνομα `askl1_1f.m` και το καλούμε από την γραμμή εντολών ως εξής για να αναθέσουμε στις μεταβλητές val1, val2, val3, val4 τις τιμές 0.5, 1, 2, 5 αντίστοιχα:

```
>> PS=askl_1f(0.5, 1, 2, 5);
>> PS
```

2. Άσκηση #2

α) Εκφώνηση

Ο μαθηματικός τύπος που υπολογίζει την έκφραση e^x δίνεται από τον τύπο:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots$$

Όσο περισσότεροι όροι προστεθούν στο άθροισμα, τόσο πιο καλή προσέγγιση έχουμε. Από την άλλη, το MATLAB μπορεί να προσεγγίσει το e^x με την βοήθεια της συνάρτησης exp(x). Να δημιουργήσετε συνάρτηση y=my_exp_func(x) η οποία αφού δεχτεί μια τιμή για το x θα υπολογίζει πόσοι όροι πρέπει να προστεθούν στο παραπάνω άθροισμα έτσι ώστε να προσεγγίσουμε το e^x με ακρίβεια καλύτερη του 10^{-5} σε σχέση με το αποτέλεσμα που προκύπτει από την exp(x). Δηλαδή θα πρέπει να ισχύσει:

$$\left|προσεγγιση - e^x\right| < 10^{-5}$$

β) Ενδεικτική λύση

Για την αλγοριθμική επίλυση του προβλήματος παρατηρούμε ότι η συνθήκη $\left|προσεγγιση - e^x\right| < 10^{-5}$ είναι μια συνθήκη τερματισμού (**ΣΤ**). Δεδομένου ότι δεν γνωρίζουμε το πλήθος των όρων για να τερματίσει η (**ΣΤ**) θα χρησιμοποιήσουμε την εντολή βρόχου **while**. Δεδομένου επίσης ότι η εντολή **while** χρησιμοποιεί συνθήκη συνέχειας (**ΣΣ**) η συνθήκη τερματισμού μετατρέπεται σε συνθήκη συνέχειας με την πράξη της άρνησης. Συνεπώς:

$$\textbf{ΣΣ}=\text{ΟΧΙ}(\textbf{ΣΤ}) \rightarrow \text{ΣΣ}=\left|προσεγγιση - e^x\right| \geq 10^{-5}$$

Η ανωτέρω **ΣΣ** χρειάζεται αρχικοποίηση –ανανέωση του όρου *προσέγγιση* και αρχικοποίηση του όρου e^x με χρήση της έτοιμης συνάρτησης exp(x).

Παρατηρούμε επίσης ότι η προσέγγιση με την σειρά: $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots$ μπορεί να γραφεί ως: $e^x = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots$ δηλαδή είναι ένα άθροισμα επαναληπτικής μορφής:

$$S = S + \frac{x^i}{i!}, i = 0,1,.....$$

Όπου η μεταβλητή – μετρητής του βρόχου i συνδέεται με το πλήθος των όρων ως εξής:

πλήθος των όρων = i (ξεκινώντας όμως από το μηδέν);

Ο κώδικας που ακολουθεί παρέχει την προτεινόμενη λύση:

```
function y=my_exp_func(x)
FIXED=exp(x);
S=0; % Αρχικοποίησε το άθροισμα-σειρά στο μηδέν.
i=0; % Αντιστοιχεί στον πρώτο όρο.
oros = x^i/factorial(i);
```

```
S=S+oros; % πρόσθεσε τον πρώτο όρο στο άθροισμα-σειρά.
while abs(S-FIXED)>=10^(-5)
    i=i+1; % Αντιστοιχεί στον επόμενο όρο
    oros = x^i/factorial(i);
    S=S+oros; % πρόσθεσε τον επόμενο όρο στο άθροισμα-σειρά.
end
y=i; % ΠΡΟΣΟΧΗ: το i=0 αντιστοιχεί στον πρώτο όρο.
```

Αποθηκεύουμε το ανωτέρω πρόγραμμα με όνομα `my_exp_func.m` και το καλούμε από την γραμμή εντολών ως εξής:

```
>> x=1; y=my_exp_func(x); y
```

Η απάντηση του matlab για το y (=8) σημαίνει χρειαζόμαστε εννέα (9) όρους για να προσεγγίσουμε το e^1 με ακρίβεια 5 δεκαδικών ψηφίων σύμφωνα με τον τύπο: $e^1 = \sum_{i=0}^{8} \frac{x^i}{\mathrm{i!}}$.

3. Άσκηση #3

α) Εκφώνηση

Να γραφτεί πρόγραμμα στο MATLAB το οποίο θα παράγει N ο πλήθος αριθμούς με χρήση της διαδικασία Fibonacci:

$$a_1 = 1$$
$$a_2 = 1$$
$$a_i = a_{i-1} + a_{i-2}$$

β) Ενδεικτική λύση

Το παραπάνω πρόγραμμα μπορεί να γραφτεί με την βοήθεια του παρακάτω προγράμματος:

```
function f=fib_non_recursive(N)
if N==1
    f=1;
elseif N==2
    f=[1 1];
else
    f=[1 1];
    for i=3:N
        f(i)=f(i-1)+f(i-2);
    end
end
```

Αποθηκεύουμε το ανωτέρω πρόγραμμα με όνομα `fib_non_recursive.m` και το καλούμε από την γραμμή εντολών ως εξής:

```
>> f=fib_non_recursive(12)
```

Από την άλλη είναι γνωστό ότι το παραπάνω πρόγραμμα μπορεί να γραφτεί με την βοήθεια του παρακάτω **αναδρομικού** προγράμματος:

```
function y=fib_recursive(N)
y=[]; % Αρχικοποίησε την μεταβλητή της ακολουθίας (y) στο κενό.
for i=1:N % Για κάθε όρο
    y=[y my_fibo(i)]; % προσαρτήστε ένα νέο όρο στην (y)
end

function f=my_fibo(i) % Με αναδρομικό τρόπο.
if i==1
    f=1;
elseif i==2
    f=1;
else
    f=my_fibo(i-1)+my_fibo(i-2);
end
```

Αποθηκεύουμε το ανωτέρω πρόγραμμα με όνομα `fib_recursive.m` και το καλούμε από την γραμμή εντολών ως εξής:

```
>> f=fib_recursive(12)
```

Παρατηρούμε ότι τα αποτελέσματα είναι ίδια. Η έννοια του αναδρομικού προγράμματος σημαίνει ότι μια συνάρτηση (στην προκειμένη η συνάρτηση `my_fibo)` μπορεί να καλέσει τον εαυτό της: Το matlab όπως και η C/C++ υποστηρίζει την αναδρομή.

4. Εργαστηριακή Άσκηση #4 (υλοποίηση στο εργαστήριο)

α) Εκφώνηση

Να γραφτεί πρόγραμμα στο MATLAB το οποίο θα παράγει το παραγοντικό N! ενός ακεραίου N. Ο υπολογισμός του παραγοντικού θα γίνεται χρησιμοποιώντας τόσο τον μη αναδρομικό τύπο του $N!=1\times2\times3\times\ldots\times N,\ 0!=1$ όσο και τον αναδρομικό $N!=N\times(N-1)!,\ 0!=1$.

5. Εργαστηριακή Άσκηση #5 (υλοποίηση στο εργαστήριο)

α) Εκφώνηση Μια φορά κι ένα καιρό, στη μακρινή Κίνα ήταν ένας αυτοκράτορας που είχε πάθος με τα παιχνίδια - κυρίως τα επιτραπέζια. Έφτασε όμως μια μέρα που είχε παίξει και είχε βαρεθεί όσα παιχνίδια υπήρχαν. Διέταξε λοιπόν να του φτιάξουν ένα παιχνίδι με απλούς κανόνες, το οποίο όμως κάθε φορά που θα το έπαιζε να είναι διαφορετικό ώστε να μην βαρεθεί ποτέ. Όποιος κατάφερνε να του φτιάξει ένα τέτοιο παιχνίδι θα μπορούσε να ζητήσει οποιαδήποτε αμοιβή. Ένας λοιπόν από τους συμβούλους του σκέφτηκε να δημιουργήσει το σκάκι. Ο αυτοκράτορας ενθουσιάστηκε και του είπε "Ποια θες να είναι τώρα η αμοιβή σου; Μήπως θες να παντρευτείς την κόρη μου και να γίνεις ο διάδοχός μου στο θρόνο;", "Όχι" απάντησε ο σύμβουλος "κάτι πιο απλό. Θέλω στο πρώτο από τα τετράγωνα που έχει το σκάκι να βάλεις ένα κόκκο ρύζι, στο δεύτερο 2, στο τρίτο 4, στο τέταρτο 8 κ.ο.κ διπλασιάζοντας κάθε φορά τον αριθμό των κόκκων από ρύζι. Η αμοιβή μου θα είναι όλο το ρύζι που θα υπάρχει πάνω στη σκακιέρα". "Μόνο αυτό;" Είπε ο αυτοκράτορας και διέταξε να πληρώσουν αμέσως το σύμβουλο. Τελικά όμως το ρύζι που έπρεπε να δώσουν στο σύμβουλο ήταν τόσο πολύ που ο αυτοκράτορας έδωσε όλη την περιουσία του για να τον ξεχρεώσει.

Να γραφεί πρόγραμμα σε γλώσσα MATLAB το οποίο να:
Α) υπολογίζει τον αριθμό των κόκκων ρυζιού που πήρε ο σύμβουλος.
Β) λαμβάνοντας υπόψη ότι κάθε κόκκος ρύζι μπορεί να ζυγίζει κατά μέσο όρο 0,1 γραμμάρια, να υπολογίζει το βάρος του ρυζιού σε τόνους.
Γ) θα εμφανίζει το μήνυμα "Ο σύμβουλος πήρε ..... κόκκους ρύζι που αντιστοιχούν σε ..... τόνους".

6. Εργαστηριακή Άσκηση #6 (υλοποίηση στο εργαστήριο)

α) Εκφώνηση Θεωρείστε ένα κύκλωμα που αποτελείται από μια πηγή συνεχούς τάσεως με εσωτερική αντίσταση $R_S = 10\Omega$. Η τιμή της πηγής τάσης είναι V=10 Volts. Μεταβάλλετε την αντίσταση R_L και υπολογίστε την παρεχόμενη ισχύ στο φορτίο για κάθε τιμή της R_L. Η ισχύς υπολογίζεται από τον παρακάτω τύπο:

$$P_L = I^2 \times R_L$$

όπου I το παρεχόμενο ρεύμα στο φορτίο. Το ρεύμα φορτίου μπορεί να υπολογιστεί και από τον παρακάτω τύπο:

$$I = \frac{V}{R_{Tot}} = \frac{V}{R_S + R_L}$$

Το πρόγραμμα πρέπει να εκτελεί τα παρακάτω βήματα:

i. Δημιουργία ενός πίνακα τιμών αντίστασης R_L από 1 Ω έως 100 Ω σε βήματα του 1 Ω.
ii. Υπολογισμός του ρεύματος για κάθε τιμή της R_L.
iii. Υπολογισμό της παρεχόμενης ισχύος στο φορτίο R_L.

iv. Σχεδιασμός της παρεχόμενης ισχύος στο φορτίο συναρτήσει της αντίστασης R_L.

β) Ενδεικτική λύση

Για την αλγοριθμική προσέγγιση του ανωτέρω προβλήματος παραθέτουμε τον ακόλουθο κώδικα χωρίς την χρήση βρόχων επανάληψης.

```
clear all, close all
RL=[1:1:100] (Ω);
V=10 (Volts);
RS=10 (Ω);
Rtot=RS+RL (Ω);
I=V./Rtot(A);
PL=I.^2.*RL (W);
plot(RL,PL)
grid on
xlabel('RL (OHM)')
ylabel('PL (WATT)')
```

γ) Ερωτήσεις:

i) Ο ανωτέρω κώδικας είναι σωστός; Αν όχι, μπορείτε να τον τροποποιήσετε;

ii) Ερμηνεύσατε τον ανωτέρω κώδικα.

iii) Παραθέσατε κώδικα που να επιλύει το ανωτέρω πρόβλημα με χρήση βρόχων επανάληψης.

7. Εργαστηριακή Άσκηση #7 (Εργασία)

Δημιουργείστε κώδικα MATLAB που θα παρέχει τις συντεταγμένες 8 βασιλισσών στο σκάκι οι οποίες όταν τοποθετηθούν πάνω στην σκακιέρα δεν θα επηρεάζουν η μία την άλλη όπως φαίνεται στο παράδειγμα. Στην συγκεκριμένη άσκηση θα πρέπει να παραχθεί ένα αρχείο .mat το οποίο θα μορφοποιεί τις θέσεις των βασιλισσών ως εξής:
f 8 d 7 g 6 …