PROBLEM SOLVING & ALGORITHM DEVELOPMENT

Introduction

- An algorithm is a systematic logical step-by-step procedure for solving a problem.
- When we solve a problem using a computer, we first need to design an algorithm concerning the problem.
- Generally, we use flowcharts or pseudocode in the development phase of an algorithm.

Solving problems with MATLAB

To solve a problem, use the following problem solving methodology

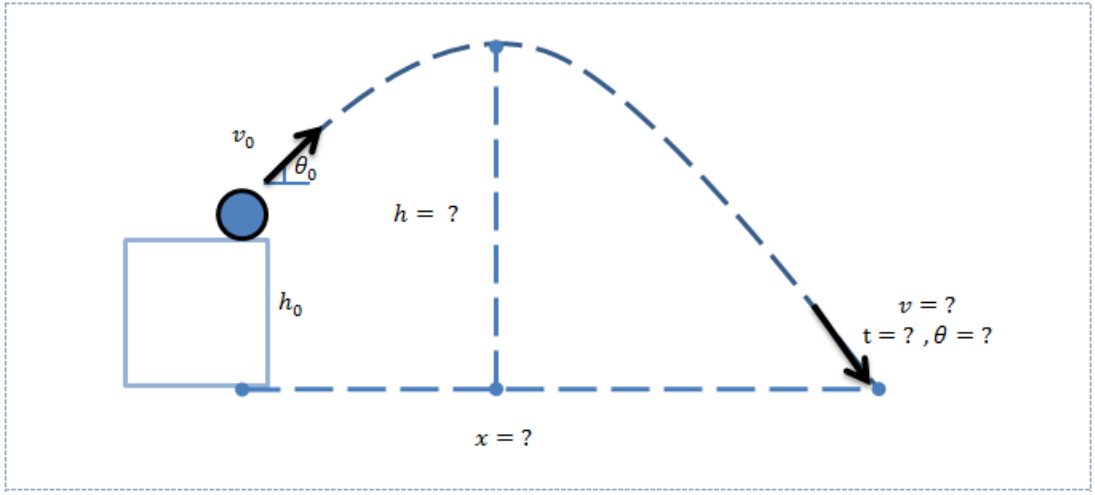
1. State the Problem
2. Describe the Input and Output
3. Develop a Hand Example
4. Develop a MATLAB Solution
   - First, clear the screen and memory: `clear, clc`
   - Now perform the following calculations in the command window or in the editor window
5. Test the solution

Solving problems with MATLAB Example

- **Problem**: For the initially given parameters */Başlangıçta verilen parametreler için,*
  - v_0: the magnitude of initial velocity vector / *hız vektörünün başlangıç değeri,*
  - h_0: initial height / *başlangıç yüksekliği*
  - θ_0: the angle of the velocity vector with the horizontal axis / *hız vektörünün yatay eksenle yaptığı açı*,
  - g: gravity / yer çekimi ivmesi

calculate the final velocity vector (its magnitude as well as its angle with the horizontal axis (v,θ)), the time passes during this travel (t), the horizontal distance it travels (x), and the maximum height it reaches to (h).

Son hız vektörünü (yanı sıra yatay eksenle yaptığı açı (v,θ)), bu seyahat sırasında geçen zamanı (t), kat ettiği yatay mesafeyi (x) ve ulaştığı maksimum yüksekliği (h) hesaplayın.



Solving problems with MATLAB Example ctd.

1) State the Problem:

For the initially given parameters

- v_0: the magnitude of initial velocity vector,
- h_0: initial height,
- θ_0: the angle of the velocity vector with the horizontal axis,
- g: gravity;

calculate the

- final velocity vector (its magnitude as well as its angle with the horizontal axis (v,θ)),
- the time passes during this travel (t),
- the horizontal distance it travels (x),
- the maximum height it reaches to (h).

Solving problems with MATLAB Example ctd.

2) Describe the Input and Output:

In this example

- v_0, h_0, θ_0 and g are the inputs.
- (v,θ), t, x, and h are the outputs.

3) Develop a Hand Example (use mathematical expressions):

Let, $\pi = 3.141592$, $g = 9.8$, $v_0 = 20$, $\theta_0 = 75$ (in degrees), $h_0 = 30$.

Then,

$v_{0y} = v_0 \sin(\pi\theta_0 / 180)$ and $v_{0x} = v_0 \cos(\pi\theta_0 / 180)$.

$t_{rise} = (v_{0y} - 0) / g$

$m.g.h_{rise} = 0.5m(v_{0y})^2 \Rightarrow h_{rise} = 0.5(v_{0y})^2 / g$

$h_{fall} = h_{rise} + h_0$ and $m.g.h_{fall} = 0.5m(v_y)^2 \Rightarrow v_y = (2gh_{fall})^{0.5}$

$t_{fall} = (v_y - 0) / g$, $d = v_{0x}(t_{rise} + t_{fall})$, $v_x = v_{0x}$

$\theta_0 = 180*(\arctan(-v_y / v_x)) / \pi$

Solving problems with MATLAB Example ctd.

4) Develop a MATLAB solution:

```
PI=3.141592; % or use pi
G=9.8;
v0=20; theta0=75; h0=30;

%assuming theta0 is given in degrees not in radians
v0y=( v0 * sin(PI*theta0/180.0) );
v0x=( v0 * cos(PI*theta0/180.0) );

t_rise=v0y/G;
h_rise=0.5*(v0y*v0y)/G; % 0.5mv^2=mgh
h_fall=h_rise+h0;
vy=sqrt(2*G*h_fall); %0.5mv^2=mgh
t_fall=vy/G;
d=v0x*(t_rise+t_fall);
vx=v0x;
theta=180*atan(-vy/vx)/PI;
```

Solving problems with MATLAB Example ctd.

5) Test the solution:

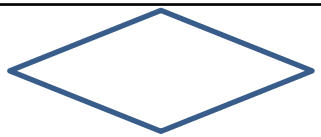
We can run the commands and output the solution as:

```
For a given set of initial values:
Initial Velocity Magnitude: 20[m/s]
Initial Velocity Angle with the horizontal: 75[degrees]
Initial height: 30[m]
Gravity: 9.8[m/(s^2)]

Final parameter set is:
Velocity Magnitude: 31.4325[m/s]
Velocity Angle with the horizontal: -80.5212
Travel time:5.1349[s]
Maximum height it reaches to: 49.0411[m]
The horizontal distance it travels: 26.5801[m]
```

Flowcharts

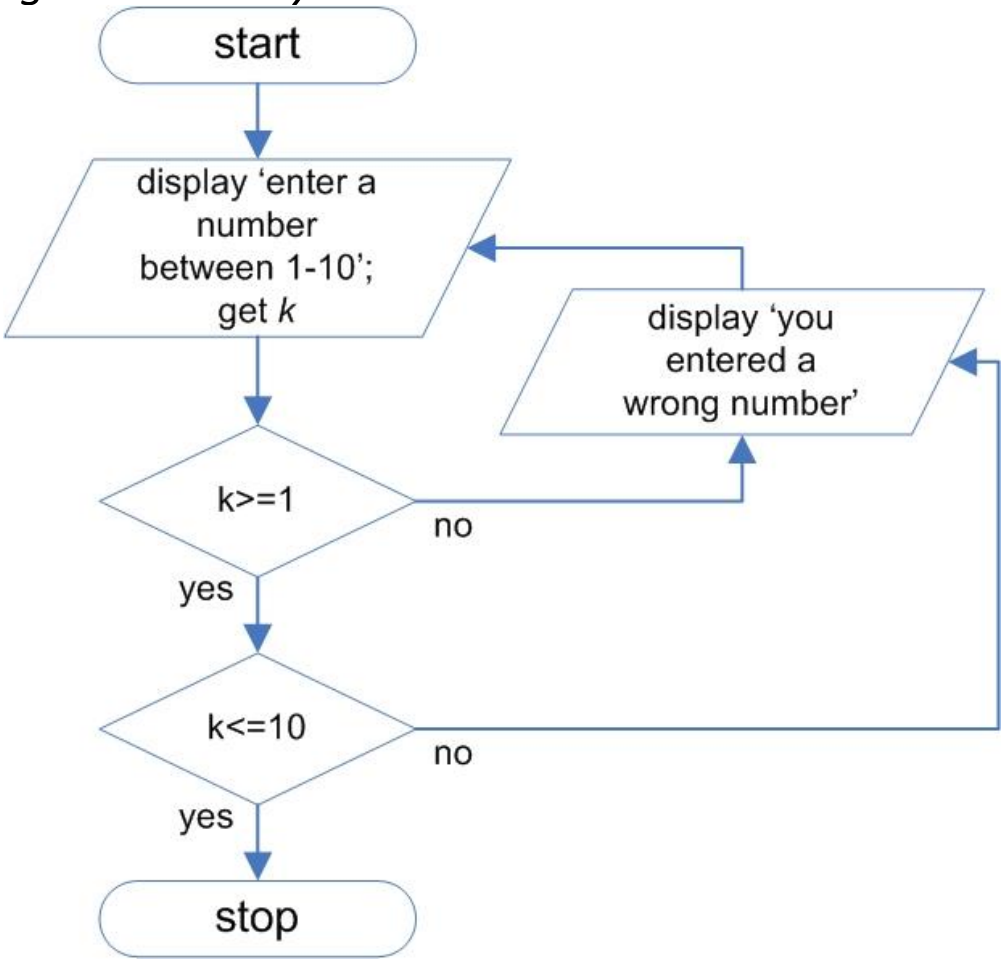
- Flowchart is a tool to distinguish the problem into smaller problems and to order them sufficiently to obtain the solution.
- We use shapes such as boxes, diamonds, etc. and arrows to build flowcharts.
- Mostly used shapes are given as follows:

Shape	Name	Description
	Flow line	
	Terminal	Start or stop
	Decision	Yes (true) or no (false) question. Ex. Is *k* equal to 10? Or *k*=10?
	Input / Output	Recieve and display data. Ex. get input from keyboard; display it.
	Process	Perform something. Ex. add *a* to *b*.

Example - 1

- Ask user to input a number between 1-10.
  Kullanıcıdan 1-10 arasında bir sayı girmesini isteyiniz.

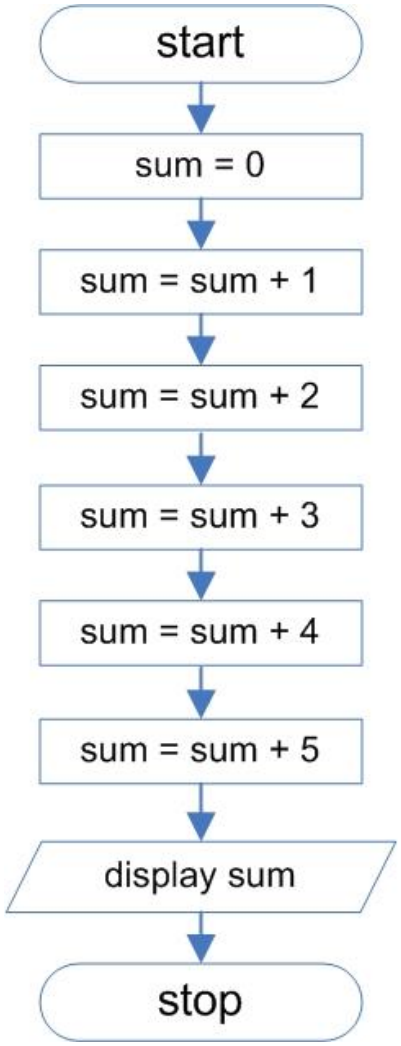
1. start
2. get the value (k)
3. if k is smaller than 1, go to step-4, otherwise go to step-5
4. display ‘you entered a wrong number’ and go to step-2
5. if k is larger than 10, go to step-4
6. stop



Example - 2

- Sum up numbers from 1 to 5.
  1'den 5'e kadar sayıları toplayınız.

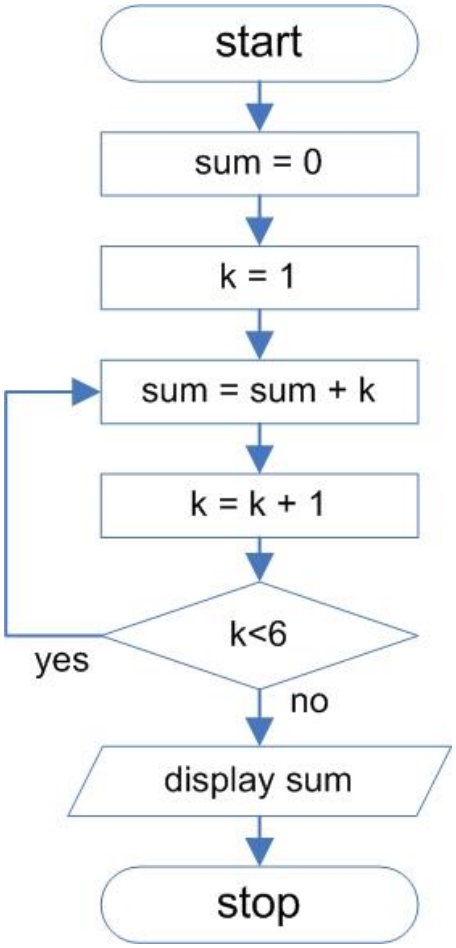
1. start
2. sum = 0
3. sum = sum + 1
4. sum = sum + 2
5. sum = sum + 3
6. sum = sum + 4
7. sum = sum + 5
8. output the sum
9. stop



Example - 2

- Sum up numbers from 1 to 5.
  1'den 5'e kadar sayıları toplayınız.

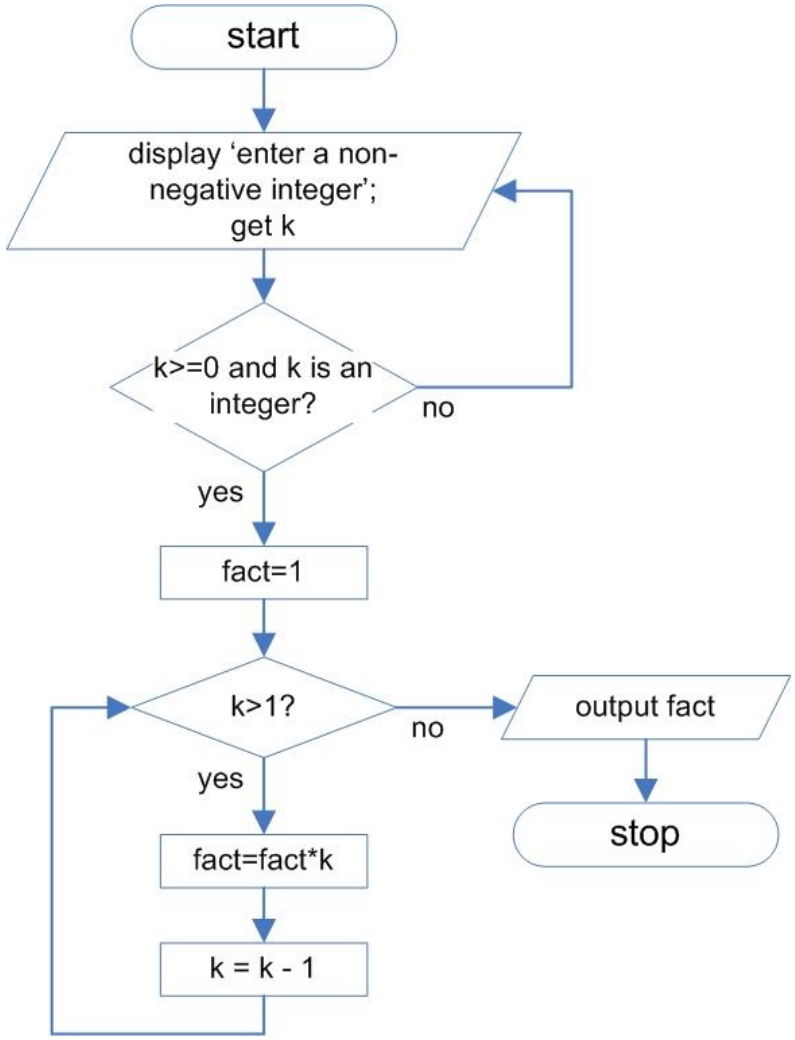
1. start
2. sum = 0
3. k = 1
4. sum = sum + k
5. k = k + 1
6. if k<6 go to step-4
7. output the sum
8. stop



Example - 3

- Ask user to input a non-negative integer and compute its factorial.
  Kullanıcıdan negatif olmayan bir tamsayı girmesini ve faktöriyelini hesaplamasını isteyiniz.

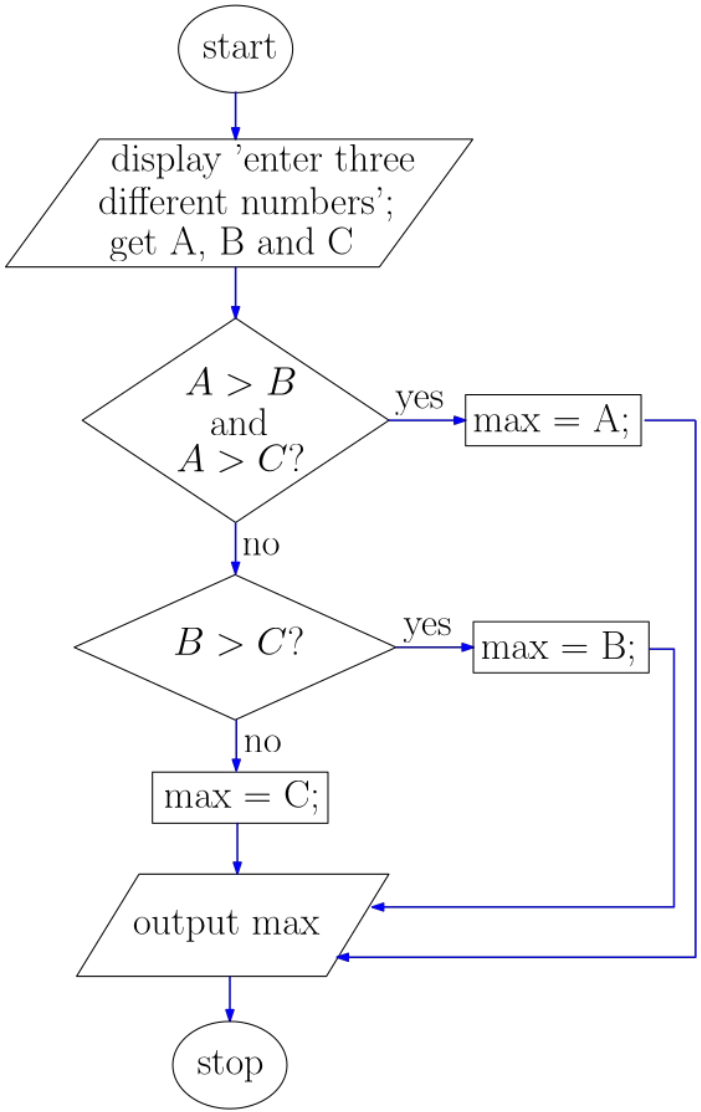
1. start
2. display ‘enter a non-negative integer’, get the value (k)
3. if k is negative or it is not an integer, go to step-2
4. fact = 1
5. if k is less than or equal to 1, go to step-9
6. fact = fact * k
7. k = k – 1
8. if k is larger than 1, go to step-6
9. output fact
10. stop



Example - 4

- Find the largest among three different numbers entered by the user.
  Kullanıcı tarafından girilen üç farklı sayı arasından en büyüğünü bulun.

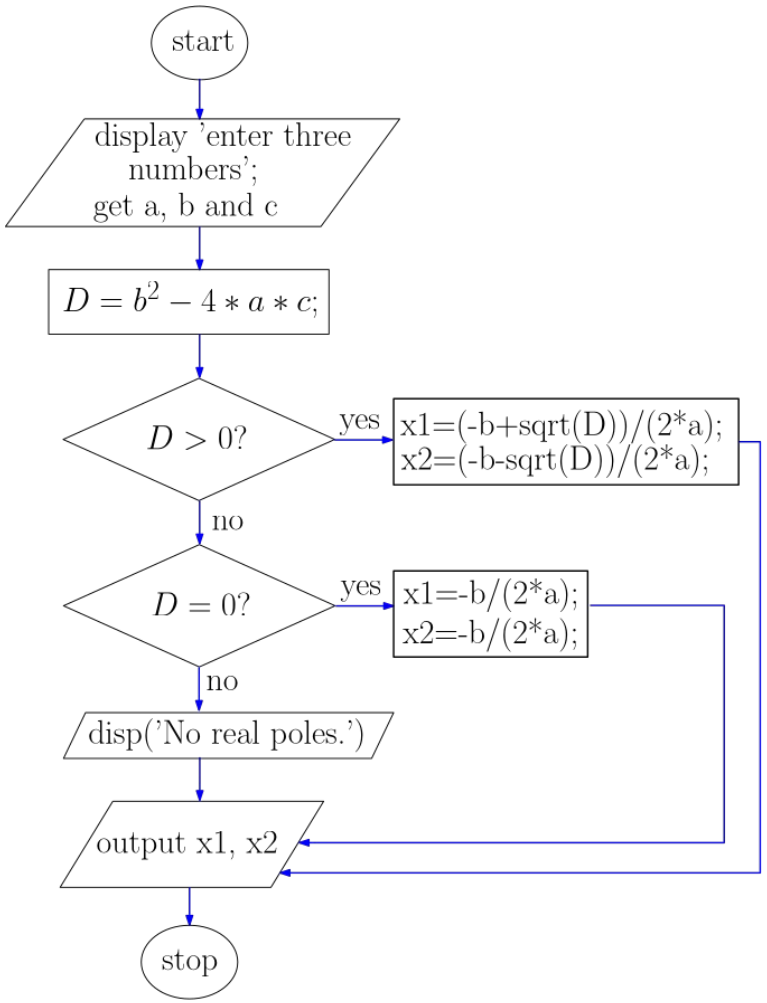
1. start
2. get the values of A,B,C
3. if A > B and A> C, go to step-9
4. max=A;
5. else if B > C, go to step-9
6. max=B;
7. else, go to step-9
8. max=C;
9. output max
10. stop



Example - 5

- Find all the real roots of the equation ax^2+bx+c=0. The coefficients will be entered by the user.
  ax^2+bx+c=0 denkleminin tüm reel köklerini bulun. Katsayılar kullanıcı tarafından girilecektir.

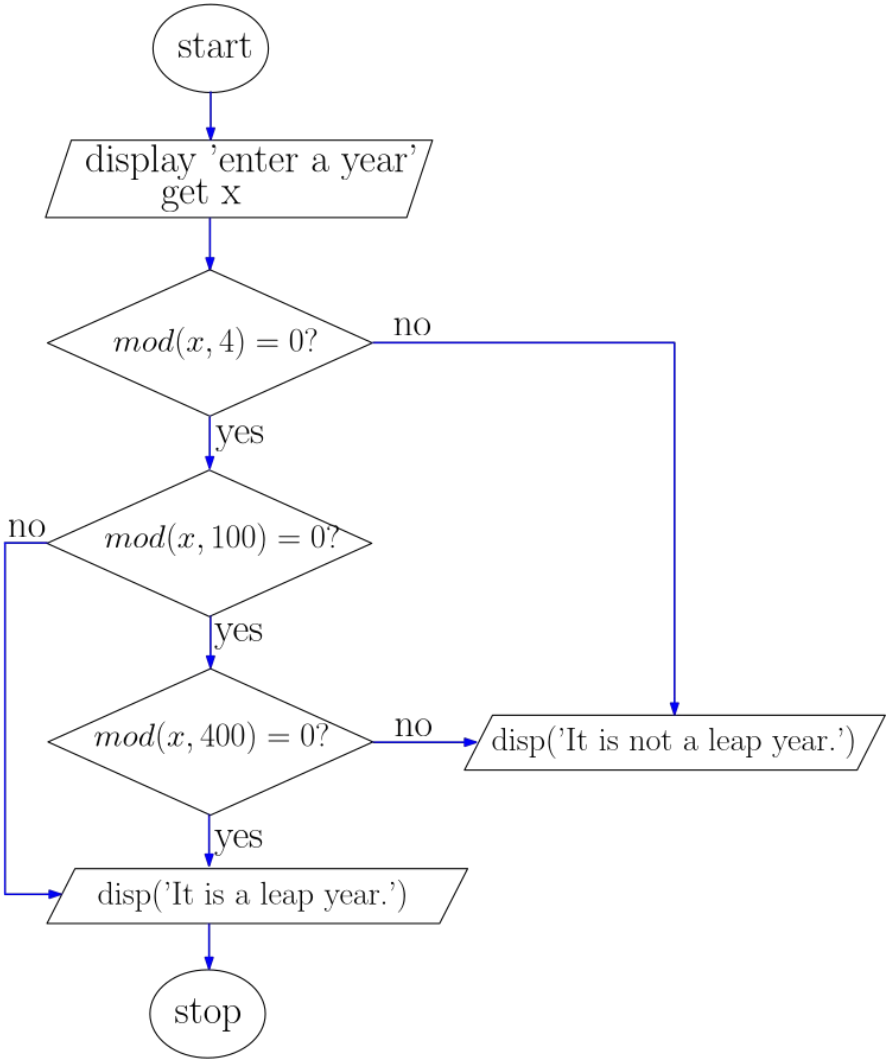
1. start
2. get the values of a,b,c
3. D = b^2 - 4*a*c;
4. if D>0, go to step-10
5. x1 = (-b + sqrt(D)) / (2*a);
   x2 = (-b - sqrt(D)) / (2*a);
6. if D=0, go to step-10
7. x1 = -b / (2*a);
   x2 = -b / (2*a);
8. if D<0,
9. disp('No real poles.')
10. output x1,x2
11. stop



Example - 6

- Ask the user to input a year, determine if it is a **leap** year or **not**.
  *Kullanıcının girdiği yılın **artık** yıl olup olmadığını belirleyiniz.*

1. start
2. get the year, x
3. if mod(x,4)=0, go to step 4, otherwise go to step 7
4. if mod(x,100)=0, go to step 5, otherwise go to step 6
5. if mod(x,400)=0, go to step 6, otherwise go to step 7
6. disp('It is a leap year.')
7. disp('It is not a leap year.')
8. stop



Example - 7

- Determine the number of terms needed for the sum of the harmonic series to exceed a given integer K entered by the user:

Aşağıdaki gibi bir harmonik serinin toplamının kullanıcı tarafından girilen belirli bir K tamsayısını aşması için gereken terim sayısını belirleyiniz.

$$\sum_{n=1}^{N} \frac{1}{n} > K$$

1. start
2. get the number, K
3. sum=0; n=1;
4. if sum<=K, go to step 5, otherwise go to step 6
5. sum = sum +(1/n); n = n+1;, go to step 4
6. output n
7. stop