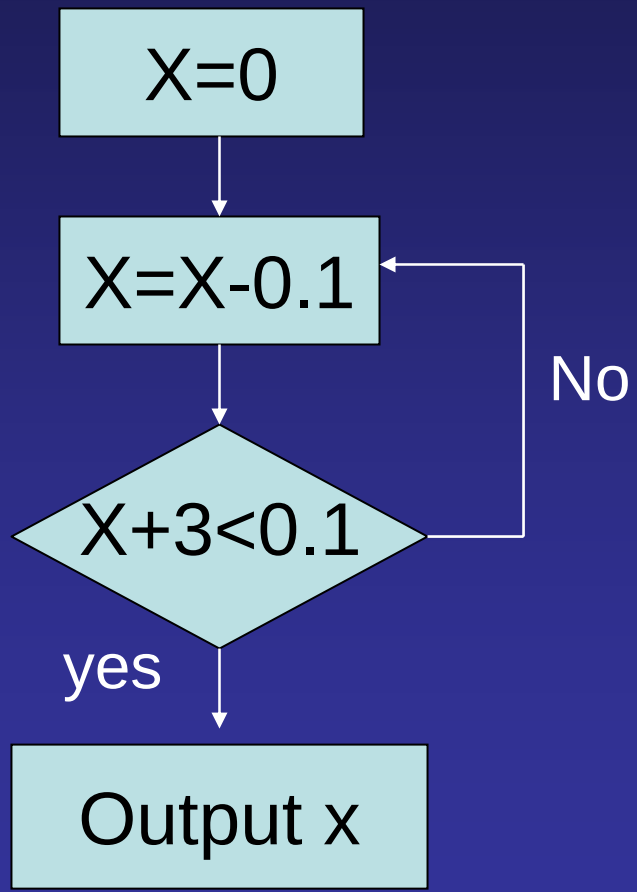
MatLab & Programming

Programming



Programming

What is programming?

Programming is the preparation of a step-by-step instruction for a computer to follow

When is programming “profitable”

*repetitive computation

*automation/real time control

*reusable “code” – objects

Programming languages

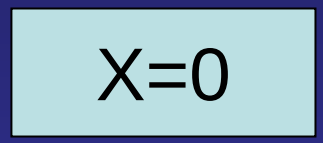
C, C++, C#, java, m-lab script

Anatomy of a program

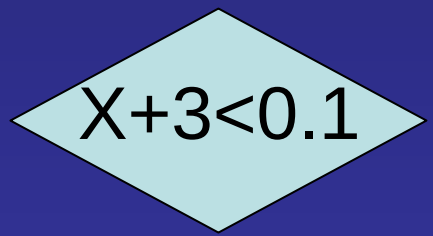
Flow chart – a graphic representation of the logical sequence of instructions

Algorithm – a sequence of instructions designed to solve a specific problem

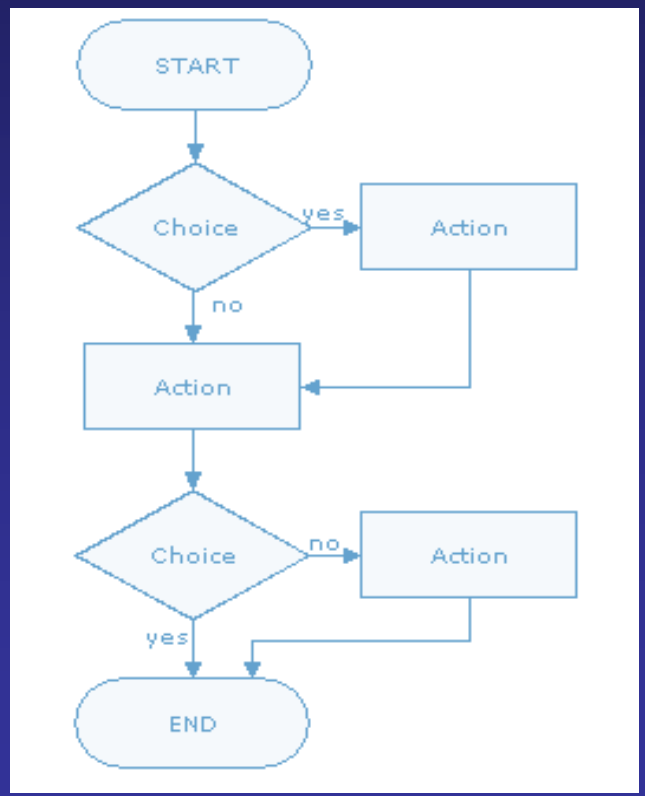
Action



Decision



Terminal

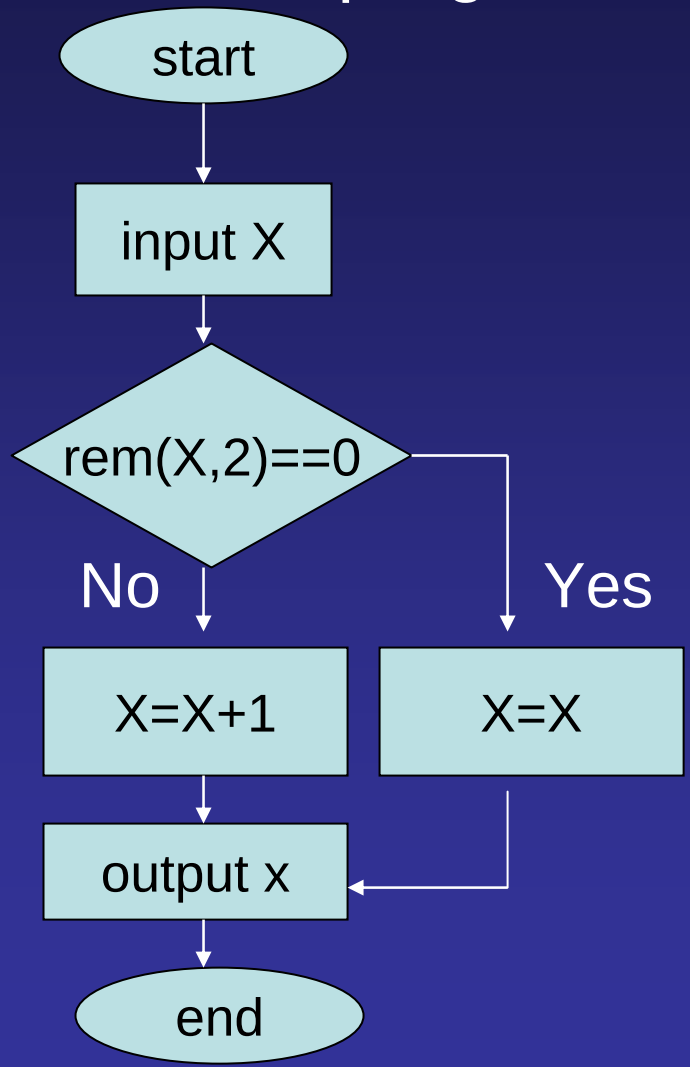


Conditionals

Conditional is a branching point in the program.

Depending on specific condition, the program can take different actions.

Example: a simple program that add 1 to odd integer
input and do nothing to even integer input



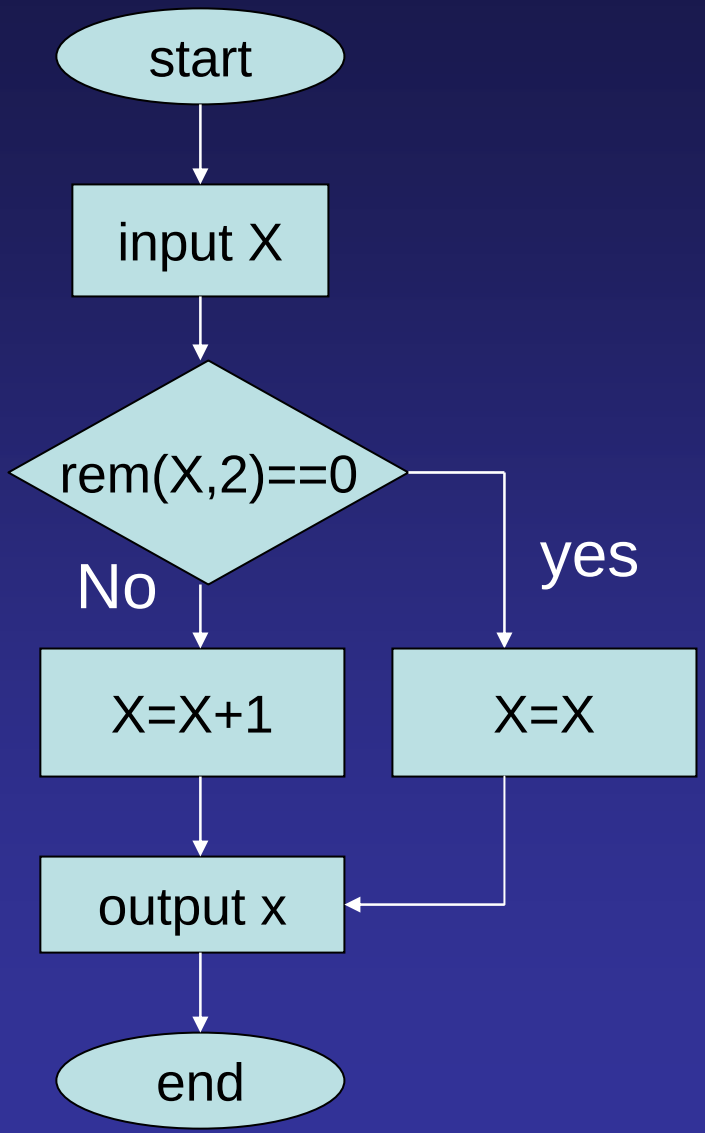
Programming in MatLab

Step 1: Create a m-file (xxx.m)
[Matlab Menu: file->new]

Step 2: Input sequence of MatLab instructions

Step 3: Save (in working directory) and run
[Editor Menu:debug->save & run]

MatLab realization of program




```
x=input('input integer: ');
if (rem(x,2) == 0)
    x=x;
else
    x=x+1;
end
x
```

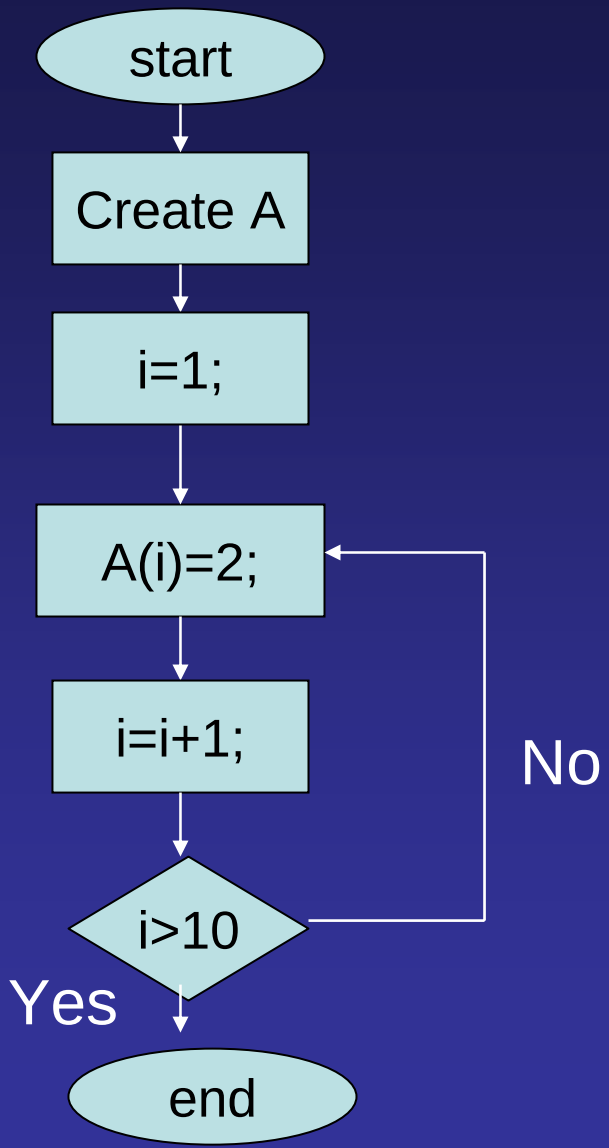
Conditional: If, else, end

```
if logic condition
        action1;
        action1;
else
        action2;
        action2;
end
```

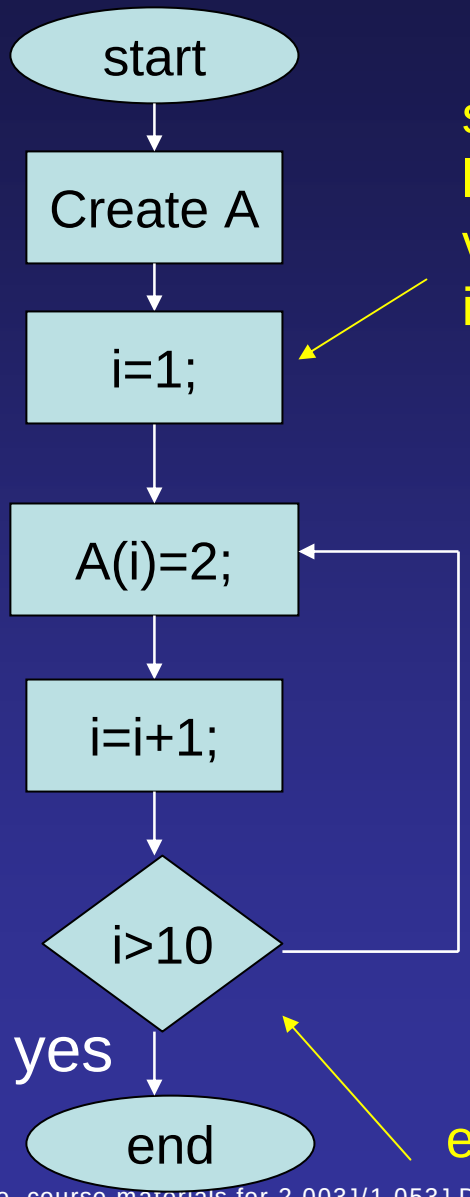
Check out also elseif

Repetition

Example: fill a 1-D matrix A with length 10 with 2s.



Repetition: for loop



start:
loop
variable
initiation

loop
variable
update

no

end conditional

```
A=zeros(10,1);
for i=1:10
    A(i)=2;
end
```

```
for start/end condition
    action1;
    action1;
    action1;
end
```

More Conditionals – elseif

```
if logic condition
        action1;
        action1;
else
        action2;
        action2;
end
```

```
if logic condition 1
        action1;
        action1;
else if logic condition 2
        action2;
        action2;
else if logic condition 3
        action3;
        action3;
else
        action4;
        action4;
end
```

More Conditionals – switch

```
switch variable
    case var1
        action1;
        action1;
    case var2
        action2;
        action2;
    case var3
        action3;
        action3;
    otherwise
        action4;
        action4;
end
```

Switch -- examples

```
a=2;
switch a
    case 1
        disp('1')
    case {2; 3; 4}
        disp('2 or 3 or 4')
    case 5
        disp('5')
    otherwise
        disp('something else')
end
```

```
a='M';
switch a
    case 'a'
        disp('A')
    case {'b'; 'c'; 'd'}
        disp('B')
    case 'M'
        disp('m')
    otherwise
        disp('something else')
end
```

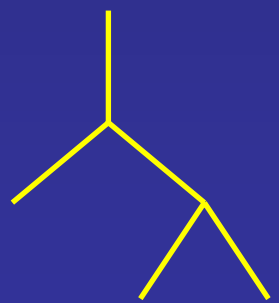
Conditionals – if or switch

When should we use “if-elseif-else-end” or “switch-case-otherwise-end”?

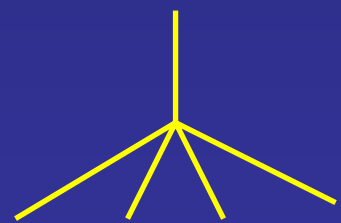
There are no fix rules … whatever makes the inherent logic clearer to the programmer and the reader

“if” is more binary decision process while “case” is more tree-like

“if”



“case”



Nesting – layers and layers

```
a=3;
if rem(a,2) ~= 0
        a=a+1;
else
        if a== 0
                a=a-1;
         else
                a=a+2;
         end;
end;
disp(a)
```

```
a='M';
switch a
    case 'a'
        disp('A')
    case {'b'; 'c'}
        switch a
                case {'b'}
                        disp('B')
                case {'c'}
                        disp('C')
        end;
otherwise
        disp('something else')
end;
```

Loops: more for loops

```
for start/end condition
        action1;
        action1;
        action1;
end
```

Ending condition is tested at the "for" statement

```
for a=1:5
        disp(a);
end
```

Output: 1, 2, 3, 4, 5

```
for a=1:2:5
        disp(a);
end
```

Output: 1, 3, 5

```
for a=1:-2.5:-5
        disp(a);
end
```

Output: 1, -1.5,-4

```
for a=-10:-2.5:-5
        disp(a);
end
```

Output:

Nesting “For” loops

```
for start/end condition1
        action1;
        action1;
        for start/end condition2
                action2;
                action2;
        end;
end;
```

Example of Nested “For” Loops

Filling a 3x3 matrix where the element value is equal to the sum of its row and column number except for the diagonal elements which are zeros

```
A=zeros(3,3);
for i=1:3
    for j=1:3
        if i~=j
            A(i,j)=i+j;
        end;
    end;
end;
disp(A)
```

Output:

0	3	4
3	0	5
4	5	0

How many times did the “if” loop get executed?

More Conditionals -- while

```
while start/end condition1
        action1;
        action1;
end;
```

```
a=1;
while a < 5
        disp(a)
        a=a+1;
end;
```

Ending condition is tested at the “while” statement

Output: 1, 2, 3, 4

Looping: “for” or “while”

Use “for” loop if you know how many time you want to repeat

Use “for” loop if index is stepwise incremented

Use “while” loop if you need to have more flexible control of end condition
Make sure that the “while” loop will end!

```
a=3;
while a < 10
        disp(a);
        a=a-1;
end;
```

Example: Calculate the air-borne time & horizontal distance of a projectile

Initial velocity: 5i+5j, initial position: origin

$$\ddot{y} = -g \qquad \ddot{x} = 0$$

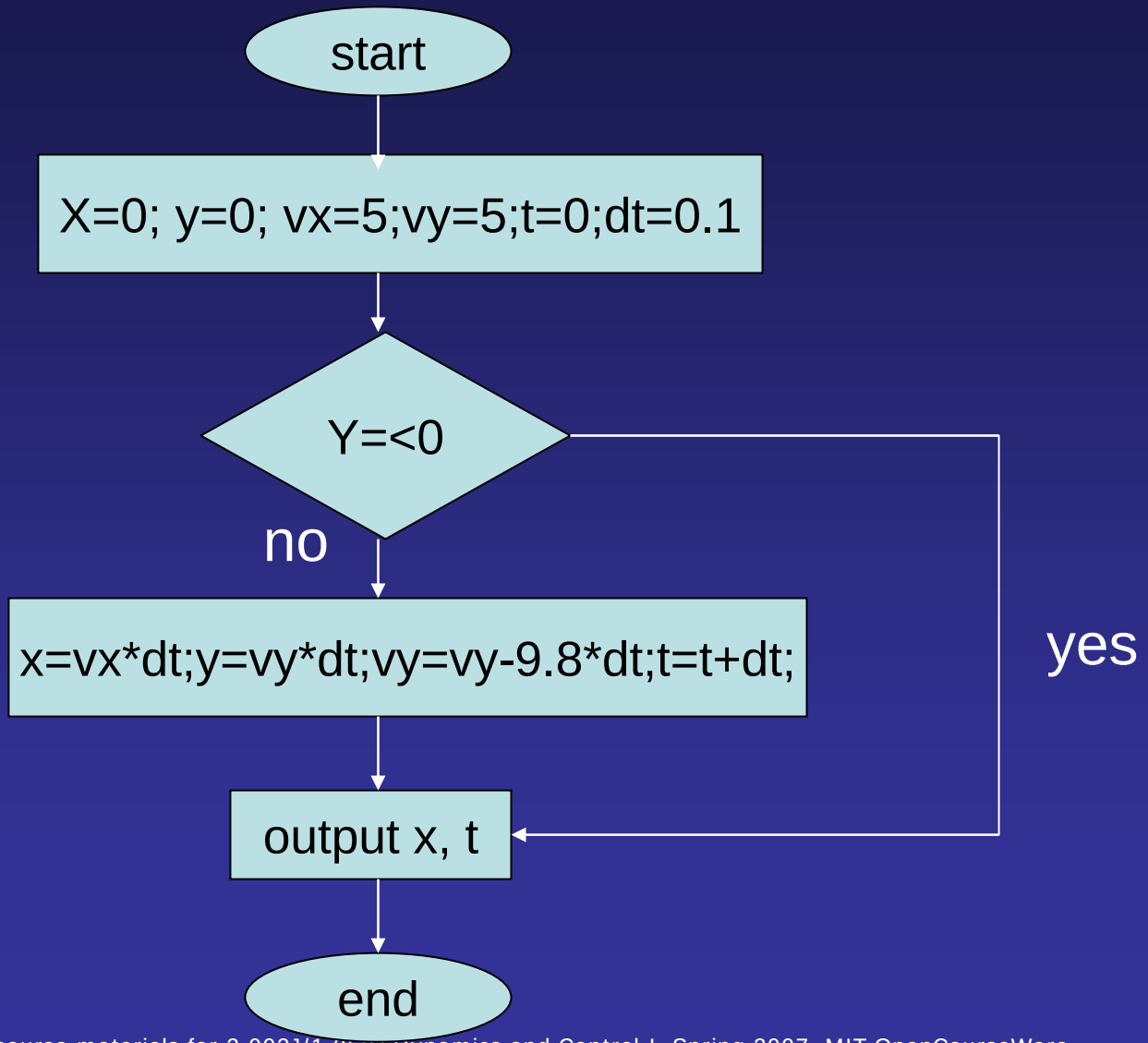
$$\dot{y} = -gt + v_{0y} \qquad \dot{x} = v_{ox}$$

$$y = -\frac{1}{2}gt^2 + v_{oy}t + y_0 \qquad x = v_{ox}t + x_0$$

Set y=0 to calculate t

$$t_{air-borne} = \frac{2v_{0y}}{g} \qquad x = \frac{2v_{0x}v_{0y}}{g}$$

Example: Calculate the air-borne time & horizontal distance of a projectile numerically



MatLab Code for Projectile

```
clear all;
x(1)=0;
y(1)=0;
vx(1)=5;
vy(1)=5;
dt=0.01;
t(1)=0;
i=1;
```

```
while y>=0
    x(i+1)=x(i)+vx(i)*dt;
    y(i+1)=y(i)+vy(i)*dt;
    vx(i+1)=vx(i);
    vy(i+1)=vy(i)-9.8*dt;
    t(i+1)=t(i)+dt;
    i=i+1;
end;

disp(x(i));
disp(t(i));
plot(x,y);
```