

# ENG 1102 - Matlab

- Relational and Logical Operators
- Conditional Statements
- Loops
  
- First --- short MTU survey (Indiv) and RAT (Team)

# Three programs today ...

- Sort 3 numbers
- Net future value calculator
- Doubling time for interest rates

# Flow Control

- Used to determine the sequence of program execution
- Flow Control in MATLAB
  - **Conditional statements**
    - if ... else ... end
  - **Loops**
    - for ... end
    - while ... end

# Conditional Statement

General Format:

**if** *expression*

*block of statements(1)*

**else**

*block of statements (2)*

**end**

Example:

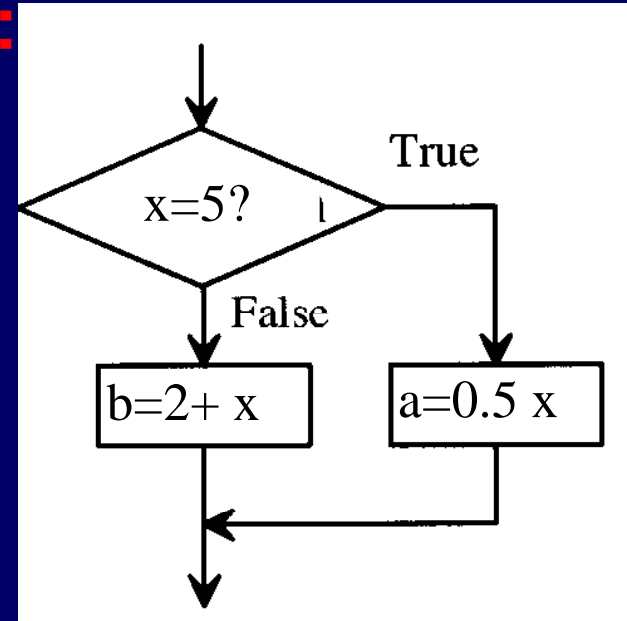
**if**  $x==5$

$a=0.5*x$

**else**

$b=2+x$

**end**



- *block statements(1)* only executed if the *expression* is **true**
- *block statements(2)* only executed if the *expression* is **false**
- Relational ( $>$ ,  $<$ ) and Logical ( $\&$ ,  $|$ ,  $\sim$ ) operations are usually combined with an if statement

# Operators

# Operators

1) arithmetic operators ( $\wedge, *, /, +, -$ )

2) relational operators ( $<, >, ==, \sim =$ )

3) logical operators ( $\sim, \&, |, \text{xor}$ )

(needed for program flow control)

# Relational Operators

- Comparison between arguments
- Result is either true (1) or false (0)
- Relational Operators (MATLAB pg 44)

< less than

<= less than or equal to

> greater than

>= greater than or equal to

== equal to

~= not equal to

# Relational Operators

- Comparison between arguments
- Result is either **true (1)** or **false (0)**
- Relational Operators (MATLAB pg 44)

< less than

<= less than or equal to

> greater than

>= greater than or equal to

 **==** equal to

~= not equal to

# Relational Operator Examples

```
>> a=3+73; b=7*2;
```

```
>> a>b
```

```
ans = 1
```

- ✓ Or can compare arrays on an element-by-element basis

```
>>x = [6, 3, 9]; y = [14, 2, 9]
```

```
>>z = (x < y)
```

```
z =  
true false false  
1 0 0
```

# Logical Operators

- Perform element-by-element operations
- Vectors must be same size except when compared to scalar

- Logical Operators (Palm page 169)

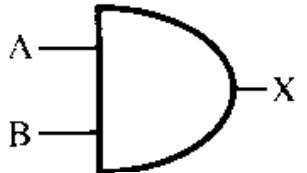

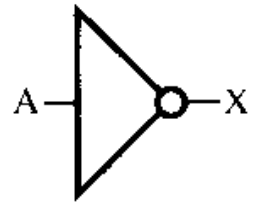

**&** and

**|** or

**~** not

**xor** exclusive or

- Digital symbols used by EE's

Function	Digital Symbol
AND	
OR	
INVERTER	
EXclusive OR	

# Logical Operators

```
if (x > 15 & y ~= 0)
```

combine Relational Expressions

# Logical Operators combine Relational Expressions

## & (and)

- Returns 1 (true) if both expressions are TRUE
- Returns 0 (false) when either expression is FALSE

## | (or)

- Returns 1 (true) if either expression is TRUE
- Returns 0 (false) when both expressions are FALSE

## ~ (not)

- Precedes any logical expression
- Changes the value of the expression to the opposite

## xor (exclusive or *(either or, but not both)*)

- Returns 1 (true) if one or the other is TRUE
- Returns 0 (false) if both are TRUE or both are FALSE
- **Note: xor is used differently than the others. It works as a function e.g. xor (A,B)**

# Truth Tables

<b>C =</b>	<b>A</b>	<b>&amp; B</b>
1	1	1
0	0	1
0	1	0
0	0	0

<b>C =</b>	<b>A</b>	<b>  B</b>
1	1	1
1	0	1
1	1	0
0	0	0

<b>C =</b>	<b>~(A</b>	<b>&amp; B)</b>
0	1	1
1	0	1
1	1	0
1	0	0

<b>C =</b>	<b>A</b>	<b>xor B</b>
0	1	1
1	0	1
1	1	0
0	0	0

# As a Team... 2 minutes

- In your Command Window, enter:

```
xnum=44;
```

```
ynum=21;
```

```
bb=33;
```

- Evaluate:

```
ynum<xnum
```

```
acdc=bb>xnum
```

```
zztop=xnum>bb
```

```
acdc & zztop
```

```
~(zztop | acdc)
```

# As a Team... 2 minutes

- In your Command Window, enter:

```
xnum=44
```

```
ynum=21
```

```
bb=33
```

- Evaluate: (can you predict the result?)

```
ynum<xnum
```

```
acdc=bb>xnum
```

```
zztop=xnum>bb
```

```
acdc & zztop
```

```
~(zztop | acdc)
```

```
c=xnum^2<=ynum*27&(zztop | bb~=ynum)
```

# if ... else

## General Format:

**if** *expression*

*block of statements(1)*

**else**

*block of statements (2)*

**end**

## Example:

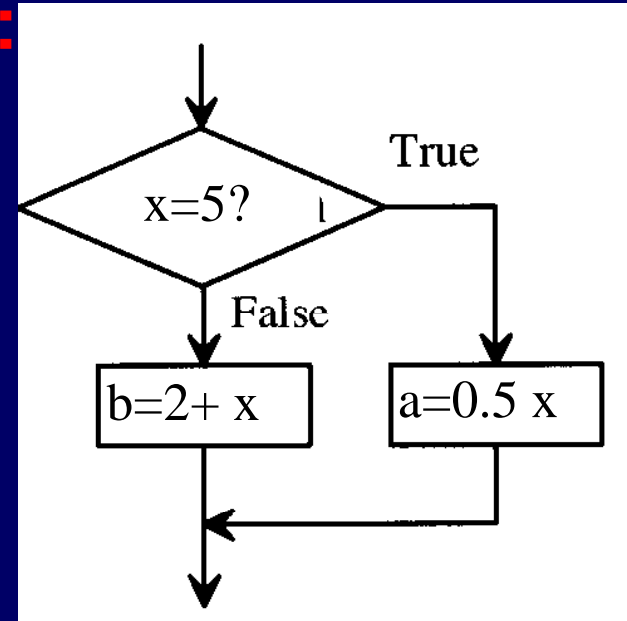
**if**  $x==5$

$a=0.5*x$

**else**

$b=2+x$

**end**



- *block statements(1)* only executed if the *expression* is **true**
- *block statements(2)* only executed if the *expression* is **false**
- Relational ( $>$ ,  $<$ ) and Logical ( $\&$ ,  $|$ ,  $\sim$ ) operations are usually combined with an if statement

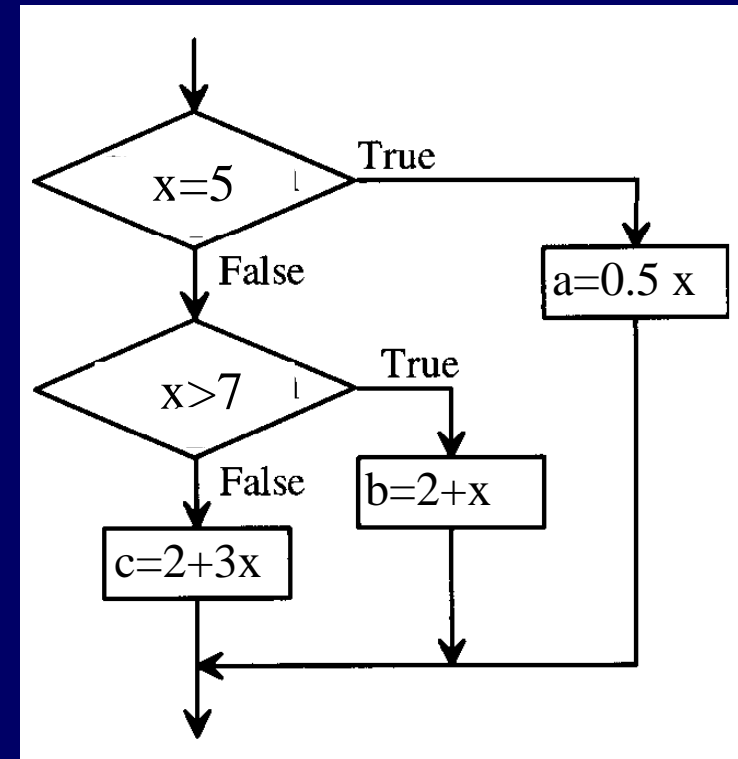
# if ... elseif ... end

## General Format:

```
if expression (1)
    block of statements a
elseif expression (2)
    block of statements b
else
    block of statements c
end
```

## Example:

```
if x==5
    a=0.5 * x
elseif x>7
    b=2+x
else
    c=2+3x
end
```



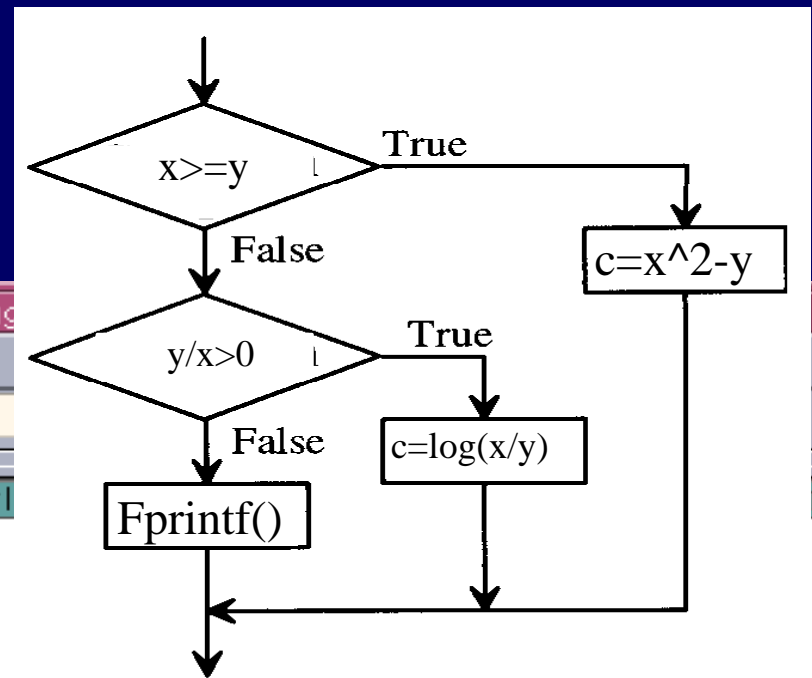
- **elseif** block of statements *b* is evaluated only if expression (1) is false and expression (2) is true

# Example

```
MATLAB Editor/Debugger
File Edit View Debug Tools Window Help
Stack:
if_example.m - /home/engfac/jczarl
x = -2; y = 5;
if x >= y
    c = x^2 - y;
elseif y/x > 0.0
    c = log(y/x);
else
    fprintf('WARNING: either x and y are both negative or x<y\n');
    fprintf('x = %f y = %f\n',x,y);
end
if_exam...
```

Ready

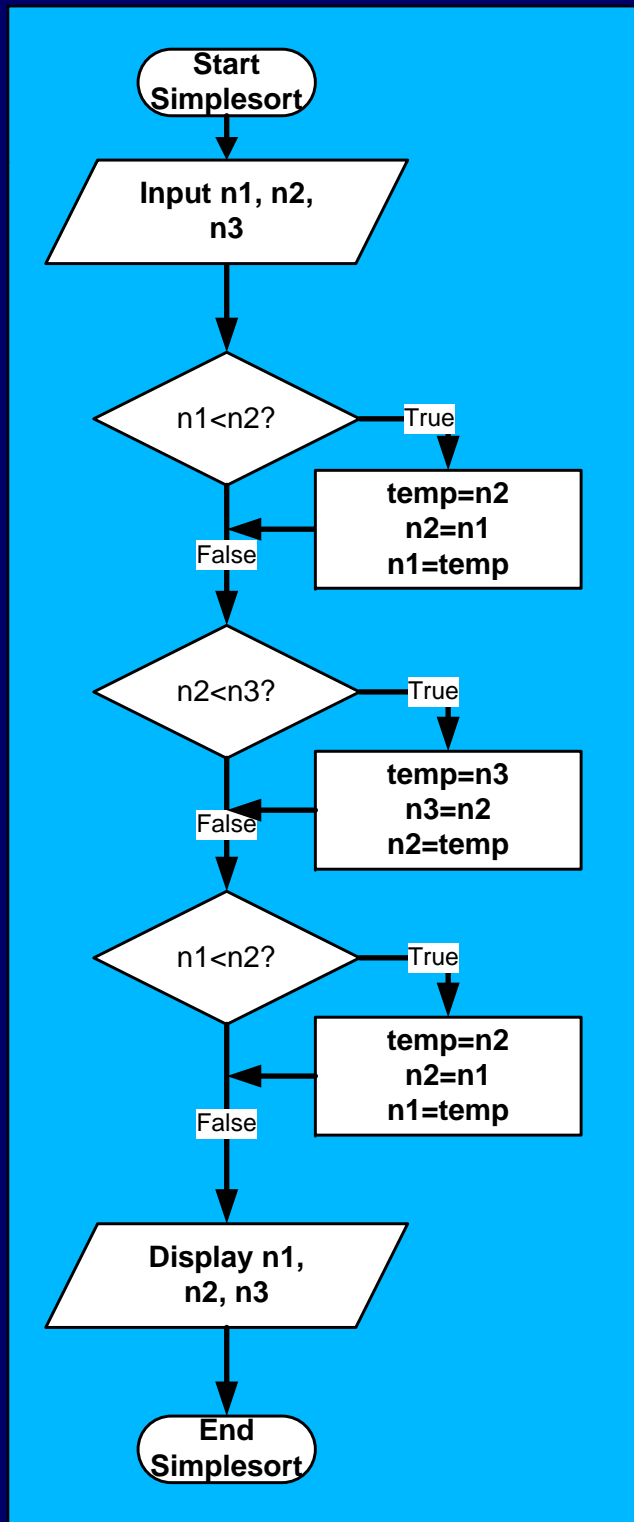
Line 10 9:21 AM



```
>> if_example
WARNING: either x and y are both negative or x<y
x = -2.000000 y = 5.000000
```

## As a team...

- Write a program using logic statements to:
  - input three numbers into variables **n1**, **n2**, **n3**
  - Sort the numbers in **descending** order (i.e., **n1** biggest, **n3** smallest)
  - Display **n1**, **n2** and **n3**



# for Loops

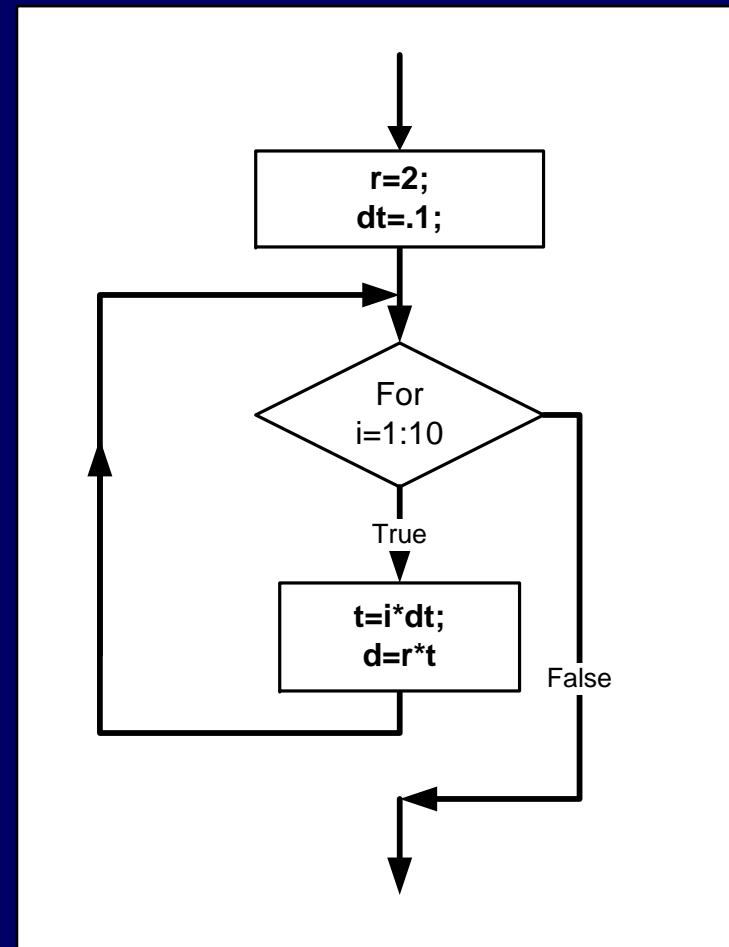
- Used when you know how many loops you need --- has built in counter

General Format:

```
for index = expression  
    block of statements  
end
```

Example:

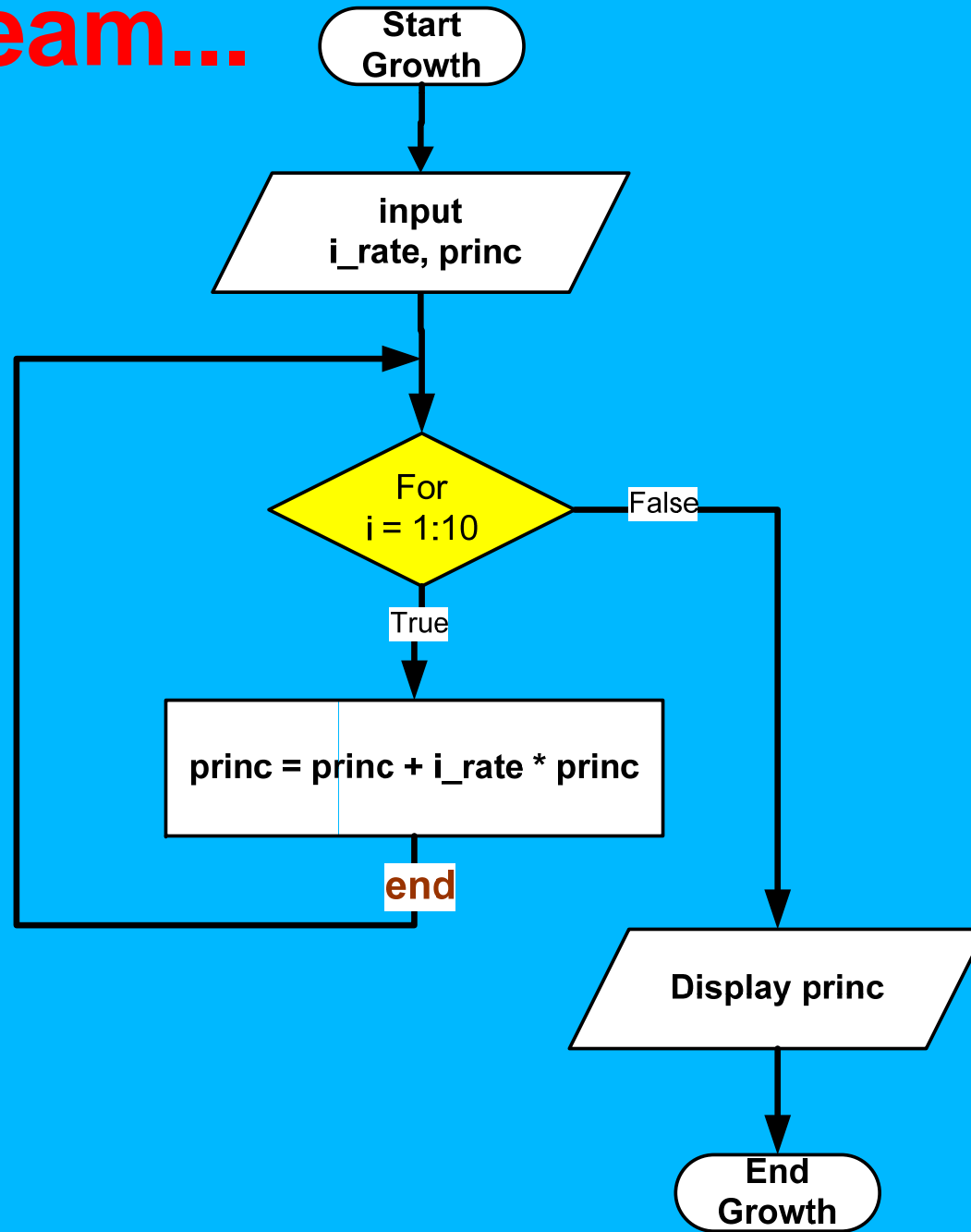
```
r=2  
dt=.1  
for i=1:10  
    t=i*dt;  
    d=r*t;  
end
```



# As a team...

- Write a program that calculates how much your money will grow over eight years (with simple, annual, compounding):
  - **input** interest rate, **i\_rate**, and initial principle, **princ**, from keyboard
  - Setup a **for** loop that compounds the interest. For example: **princ=princ\*(1+i\_rate);**
  - When the loop ends, print (disp) the final amount and the growth in terms of percent. For example: **growth=princ/ini\_princ\*100;**
  - Can you modify the program so you can input the number of years?

As a team...



# while Loops

- Used for tasks requiring repetition

General Format:

**while** *expression*

*block of statements*

**end**

Example:

**r=2**

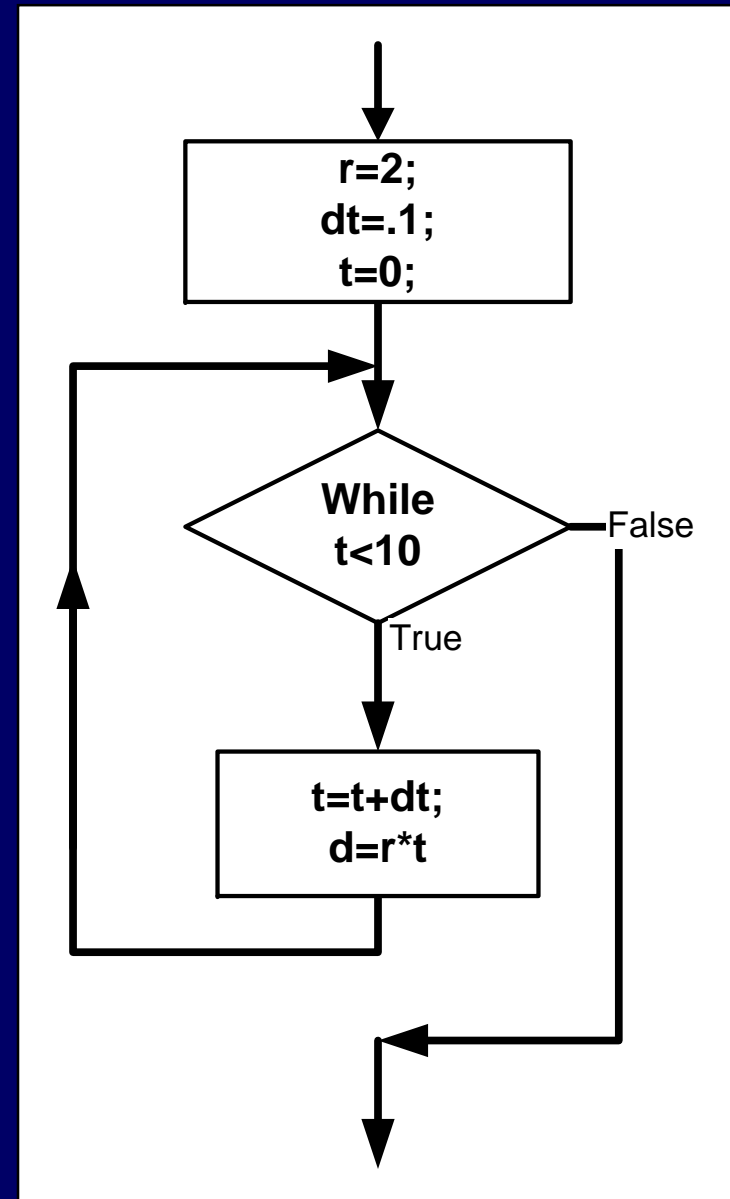
**dt=.1; t=0;**

**while t<10**

**t=t+dt;**

**d=r\*t;**

**end**



## As a team...

- Write a program that calculates how quick your money will double:
  - **input** interest rate, **i\_rate**, and set initial principle, **princ**, to 1000
  - Setup a **while** loop that compounds the interest. E.g. **princ = princ\*(1+i\_rate);**
  - Inside the loop, count the years; i.e., **yr=yr+1;**
  - When the loop ends, display the final amount and the number of years it took to double, **prin , yr**

# Flow Control Loops

## *Do* Loop (or *For/next* Loop)

- Used for tasks requiring a pre-set repetition of block of statements
  - These flow charts will each loop 10 times.

