Loop Statements

looping statements, counted loops, conditional loops, action, iterate, loop or iterator variable, running sum, running product, factorial, preallocate, echo printing, nested loop, outer loop, inner loop, infinite loop, counting, error-checking

Looping Statements

- Loops are used to repeat actions.
- Conditional Loops
  - while
- Counted Loops
  - for

while loops

```
while condition
        action
end
```

- while loops are used when you don't know or cannot determine ahead of time how many times the loop will be executed.
- the action should at some point alter the condition to be false. otherwise you get an "infinite loop"
  - while true ; end
  - Use Ctrl+C to break out of an infinite loop. (Ctrl+C can also be used to stop execution of any long-running matlab command).

Exercise

- Write a function [n,f]=factgthigh(high) that returns the first integer n and its factorial that is greater than the input "high".

>> [n,f] = factgthigh (100)

n =

  5

f =

  120

Error checking user input in a while loop

- Exercise: Write a function inputposnumber() that asks the user for a positive number and returns it.

>> x = inputposnumber

Enter a positive number: -5

Invalid! Enter a positive number: 5

OK!

x =

  5

- Exercise: Write a function inputposint() that asks the user for a positive <u>integer</u> number and returns it.

Exercise: countnumbersgreaterthanT

- **Write a function that takes an input vector v and a number T and returns number of elements in v greater than T.**

- Exercise: if T is not given, use T=10.

Flow control: continue, break

while condition / for range
    [statements A]
    continue
    [statements B]
    break
    [statements C]
end

Exercise

```
a='x';
while ~strcmp(a,'y')&&~strcmp(a,'n')
    a=input('Enter (y/n) :','s');
end
```

- Fill-in the body of the while loop so that the code is equivalent to above.

```
while true
    a=input('Enter (y/n) :','s');

    ....

    ....
```