Some Fundamental Algorithms
Swapping

- **Problem**
  
  Given two variables, a and b exchange the values assigned to them

Let \( a = 10 \) and \( b = 20 \) (Initial configuration)

Want \( a = 20 \) and \( b = 10 \) (Target configuration)

Assignment operator "=" can be used.

\( a=b \) causes copying of the value stored in a into b

If we do \( a=b \) and \( b=a \) what happens?
Swapping

Can use a temporary variable $t$.

Now,

\[
\begin{align*}
  t &= a & \# & t \text{ now gets the value of } a \\
  a &= b & \# & a \text{ gets the value of } b \\
  b &= t & \# & b \text{ gets the value in } t \text{ i.e. the original value of } a
\end{align*}
\]
Counting

• Problem

Given a set of n students who have marks in the range (0,100) in a course. The pass marks in the course are 50. Find how many students have passed the course.

For one student:

Input marks of this student
If the marks of this student > passmarks
   The count of students passed = 1
Otherwise
   The count of students passed = 0
Counting

• Problem
  Given a set of n students who have marks in the range (0,100) in a course. The pass marks in the course are 50. Find how many students have passed the course.

For one student:
  Input marks of this student
  If the marks of this student > passmarks
     The count of students passed = 1
  Otherwise
     The count of students passed = 0
Counting

For n students:
Need to repeat the process and update the count.

This can be achieved by using a loop construct such as while.
count = 0 # initially the count of students who have passed is 0
m = 0  # m is a variable that gives the number of students for whom marks have been processed
while (m < n) do
  m = m+1
  input marks of the current student
  If marks >= 50 count = count +1
end-while
output count
Summation

• Problem
  Given a set of n numbers find the sum of these numbers.

  Let the n numbers be $a_1, a_2, a_3, \ldots, a_n$
  Resulting sum = $a_1+a_2+a_3+\ldots+a_n$

  For first two numbers sum = $a_1+a_2$
  The third number $a_3$ can be added into the sum
  $\text{sum} = \text{sum} + a_3$  # update of sum
  Similarly for next number $a_4$
  $\text{sum} = \text{sum} + a_4$
Summation

sum = 0 # initial value of sum
input n # total number of number to be added
i = 0 # loop index
while (i<n) do
  i=i+1
  input ai
  sum = sum + ai
end-while
output sum
Factorial

• Problem
Given a number \( n \), compute \( n \) factorial (\( n! \)) where \( n \geq 0 \)

We know
\[
\begin{align*}
0! &= 1 \\
1! &= 1 \\
2! &= 1 \times 2 \\
3! &= 1 \times 2 \times 3 \\
n! \text{ can be computed as } n \times (n-1)!
\end{align*}
\]
Factorial

input n
factor=1
for i=1 to n do
    factor = i*factor
output factor
Sine function as series

• Problem
Evaluate \( \sin(x) \) as a series expansion i.e., upto \( n \) terms

\[
\sin(x) = \frac{x}{1!} - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots
\]
Sine function as series

- Problem
  Evaluate \( \sin(x) \) as a series expansion i.e., upto \( n \) terms

\[
\sin(x) = \frac{x}{1!} - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots
\]

One can observe:

\[
\frac{x^i}{i!} = \frac{x}{1} \times \frac{x}{2} \times \frac{x}{3} \times \cdots \times \frac{x}{i}
\]

for \( i \geq 1 \)
Sine function as series

Current term can be computed using the previous term.

\[ \text{current\_term (ith term)} = \text{previous\_term} \times x^2 / i \times (i-1) \]

Change of sign can be simply done

\[ \text{sign} = -\text{sign} \]

Also next \( i \) is obtained by incrementing by 2

\[ i = i + 2 \]

The rest is summation of a series to sum to \( n \) terms.
Fibonacci series

• Problem
  Generate and print first n terms of Fibonacci sequence, which looks like
  
  0, 1, 1, 2, 3, 5, 8, 13 ....

  First two terms are given.
  Third term = second term + first term
  Forth term = third term + second term

  In general
  Current term = sum of the two previous terms
Fibonacci series

Initially  
\[
\begin{align*}
  a &= 0 \\
  b &= 1 \\
  c &= a + b
\end{align*}
\]

Next time  
\[
\begin{align*}
  a &= b \\
  b &= c \\
  c &= a + b \quad \text{(New)}
\end{align*}
\]

This can continue so the algorithm may look like

\begin{verbatim}
input n
a=0
b=1
i=2  # keeps track of number of terms decided
while i<n do
  i=i+1
  c=a+b
  a=b
  b=c
end-while
\end{verbatim}
Fibonacci series

Initially

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>a+b</td>
</tr>
</tbody>
</table>

Next time

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>c</td>
<td>a+b</td>
</tr>
</tbody>
</table>

This can continue so the algorithm may look like

```
input n
a=0
b=1
i=2  # keeps track of number of terms decided
while i<n do
    i=i+1
    c=a+b
    a=b
    b=c
end-while
```

Any improvement possible?