What is an algorithm?

An algorithm is a sequence of instructions or rules or operations for manipulating data to produce some result.

Think of an algorithm as a recipe. In CS, the recipe works with digital information such as numbers, text strings, images, sounds,....

See Khan Academy course on Algorithms for a good intro
Today: grade school arithmetic

- addition
- subtraction
- multiplication
- division

You learned algorithms for performing these operations!
Grade school addition

You needed to memorize single digit sums to do this.
(Remember how you learned single digit sums?)
What is the algorithm for addition?

Let’s use an array for a, b, and the result r.

\[
\begin{array}{ccccc}
\hline
\end{array}
\]
Grade School Addition

For each column $i$  

compute single digit sum $a[i] + b[i]$ and add the carry value from previous column

determine the result $r[i]$ for that column

determine the carry value for the next column

Grade School Addition
(“pseudocode”)

\[
\begin{aligned}
carry &= 0 \\
\text{for } i = 0 \text{ to } N - 1 \text{ do} \\
\quad r[i] &\leftarrow (a[i] + b[i] + carry) \mod 10 \\
\quad carry &\leftarrow (a[i] + b[i] + carry)/10 \\
\text{end for} \\
r[N] &\leftarrow carry
\end{aligned}
\]

(To be explained on next slides.)
Grade School Addition ("pseudocode")

\[
carry = 0
\]

for \(i = 0\) to \(N - 1\) do

\[
\begin{align*}
  r[i] & \leftarrow (a[i] + b[i] + carry) \mod 10 \\
  carry & \leftarrow (a[i] + b[i] + carry) / 10
\end{align*}
\]

end for

\[
r[N] \leftarrow carry
\]

compute single digit sum \(a[i] + b[i]\) and
add the carry value from previous column
determine the result \(r[i]\) for that column
Grade School Addition ("pseudocode")

\[
\begin{align*}
carry &= 0 \\
\text{for } i &= 0 \text{ to } N - 1 \text{ do} \\
    r[i] &\leftarrow (a[i] + b[i] + carry) \mod 10 \\
    carry &\leftarrow \left(\frac{a[i] + b[i] + carry}{10}\right) \\
\text{end for} \\
r[N] &\leftarrow carry
\end{align*}
\]

Integer division (ignore remainder)

determine the carry value for the next column
The grade school addition algorithm is non-trivial.

It makes use of a good *number representation*: it represents each number as *sum of powers of 10*.

(Hindu-Arabic system invented ~2000 years ago)

*Do you understand how it works?*
Imagine an addition algorithm that is based on Roman numerals:

It would be rather awkward!
Grade school subtraction

\[
\begin{align*}
924 \\
- 352 \\
\end{align*}
\]

572

How to write an algorithm for doing this?
How to write an algorithm for doing this?
How to describe the “borrowing” step?
(You will implement this in Assignment 1.)
Multiplication

Q: What do we mean by $a \times b$?

(assuming integers)
Q: What do we mean by $a \times b$? (assuming integers)

A: $(a + a + \ldots + a)$, $b$ times

$a$ is the “multiplicand”

$b$ is the “multiplier”
Q: What do we mean by $a \times b$?
(assuming integers)

A: $(a + a + ... + a)$, $b$ times

or $(b + b + ... + b)$, $a$ times
The definition of multiplication suggests a slow algorithm:

\[
\begin{align*}
\text{product} &= 0 \\
\text{for } i = 1 \text{ to } b \text{ do} \\
&\quad \text{product} \leftarrow \text{product} + a \\
\text{end for}
\end{align*}
\]

You learned a much faster algorithm in grade school.
Grade school multiplication

\[
\begin{array}{c}
352 \\
\times 964 \\
\hline
1408 \\
2112 \\
3168 \\
\hline
3391328
\end{array}
\]

“multiplicand”

“multiplier”

\[
\begin{array}{c}
a[N] \\
b[N] \\
\hline
\text{tmp}[N][2N] \\
\hline
r[2N]
\end{array}
\]
Grade school multiplication

Step 1: make 2D table \( tmp[i][j] \)

\[
\text{for } j = 0 \text{ to } N - 1 \text{ do} \\
\quad \text{carry} \leftarrow 0 \\
\quad \text{for } i = 0 \text{ to } N - 1 \text{ do} \\
\quad\quad \text{prod} \leftarrow (a[i] * b[j] + \text{carry}) \\
\quad\quad \text{tmp}[j][i + j] \leftarrow \text{prod} \mod 10 \\
\quad\quad \text{carry} \leftarrow \text{prod} / 10 \\
\quad \text{end for} \\
\quad \text{tmp}[j][N + j] \leftarrow \text{carry} \\
\text{end for}
\]
Grade school multiplication

Step 2:  for each column in table, sum up the rows

\[
\begin{align*}
carry & \leftarrow 0 \\
\text{for } i = 0 \text{ to } 2 \times N - 1 \text{ do} & \\
\text{ } & \quad sum \leftarrow carry \\
\text{for } j = 0 \text{ to } N - 1 \text{ do} & \\
\text{ } & \quad sum \leftarrow sum + \text{tmp}[j][i] \\
\text{end for} & \\
\text{r}[i] & \leftarrow \text{sum}\%10 \\
carry & \leftarrow \text{sum}/10 \\
\text{end for}
\end{align*}
\]
Grade school multiplication specifies that we build a temporary 2D array of size $N \times N$. (the jaggy shape)

In Assignment 1, you will implement an algorithm that does not use such a 2D array.
Division

Q: What do we mean by $a / b$?
   (assuming integers, and $a > b$)
Division

Q: What do we mean by $a / b$?
   (assuming integers, and $a > b$)

A: We mean: “How many times can we subtract $b$ from $a$ before our answer is between 0 and the remainder?”
Division

Q: What do we mean by $a / b$ ?
   (assuming integers, and $a > b$)

A: $a = q*b + r$, $0 \leq r < b$

q is quotient, r is remainder
Slow division algorithm

To compute $a / b$, repeatedly subtract $b$ from $a$ until the result is less than $b$.

\[
q = 0 \\
r = a \\
\text{while } r \geq b \text{ do} \\
\quad q \leftarrow q + 1 \\
\quad r \leftarrow r - b \\
\text{end while}
\]

You learned a much faster algorithm in grade school.
Grade school division ("long division")

\[
\begin{array}{r}
723 & \overline{)41672542996} \\
3615 & \\
\hline
552 & \ldots \text{etc}
\end{array}
\]

How would you write out the algorithm? (You will do it in Assignment 1.)
Computational Complexity

What do we mean by ‘fast’ and ‘slow’?

Suppose we want to perform arithmetic operations on two integers $a, b$ which have $N$ digits each.

How many ‘steps’ does each algorithm take?
Grade School Addition

\begin{align*}
carry &= 0 \\
\text{for } i = 0 \text{ to } N - 1 & \text{ do} \\
& \quad r[i] \leftarrow (a[i] + b[i] + carry) \mod 10 \\
& \quad carry \leftarrow (a[i] + b[i] + carry)/10 \\
\end{align*}

end for

\begin{align*}
r[N] &\leftarrow carry \\
\end{align*}

We mean that each part of the program is executed 1 or N times.
Grade School Addition

```plaintext
carry = 0
for i = 0 to N - 1 do
    r[i] ← (a[i] + b[i] + carry) % 10
    carry ← (a[i] + b[i] + carry)/10
end for
r[N] ← carry
```

The time it takes is $c_1 + c_3 + c_2 \times N$ for some unspecified constants.

*When we analyze algorithms, we often ignore these constants.*
for $j = 0$ to $N - 1$ do
  $carry \leftarrow 0$
  for $i = 0$ to $N - 1$ do
    $prod \leftarrow (a[i] \times b[j] + carry)$
    $tmp[j][i + j] \leftarrow prod \% 10$
    $carry \leftarrow prod / 10$
  end for
  $tmp[j][N + j] \leftarrow carry$
end for

$carry \leftarrow 0$
for $i = 0$ to $2 \times N - 1$ do
  $sum \leftarrow carry$
  for $j = 0$ to $N - 1$ do
    $sum \leftarrow sum + tmp[j][i]$
  end for
  $r[i] \leftarrow sum \% 10$
  $carry \leftarrow sum / 10$
end for
Computational Complexity

We say...

Grade school addition takes time $O( N )$.

Grade school multiplication takes time $O( N^2 )$.

We will see a formal definition of $O( \ldots )$ in a few weeks.
TODO

• Install Eclipse. Tutorial next week
  (Wed for Sec. 001 and Thurs for Sec. 002)

• MATH 240 issue for ECSE students