Lecture 1

Grade school algorithms

Sept. 8, 2017

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

$$
\begin{array}{rrrrrl}
 & 1 & 0 & 1 & & \text{carry} \\
 & 2 & 3 & 4 & 3 & \\
+ & 5 & 8 & 1 & 9 & \\
\hline
 & 8 & 1 & 6 & 2 &
\end{array}
$$

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.

$$\begin{array}{ccccc} & a[3] & a[2] & a[1] & a[0] \\ + & b[3] & b[2] & b[1] & b[0] \\ \hline r[4] & r[3] & r[2] & r[1] & r[0] \end{array}$$

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")

$carry = 0$
for $i = 0$ to $N-1$ **do**
  $r[i] \leftarrow (a[i] + b[i] + carry)\ \%\ 10$
  $carry \leftarrow (a[i] + b[i] + carry)/10$
end for
$r[N] \leftarrow carry$

(To be explained on next slides.)

Grade School Addition ("pseudocode")

"mod"

$carry = 0$
for $i = 0$ to $N - 1$ **do**
  $\boxed{r[i] \leftarrow (a[i] + b[i] + carry)\ \%\ 10}$
  $carry \leftarrow (a[i] + b[i] + carry)/10$
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")

$carry = 0$

for $i = 0$ to $N-1$ **do**

$\quad r[i] \leftarrow (a[i] + b[i] + carry)\ \%\ 10$

$\quad \boxed{carry \leftarrow (a[i] + b[i] + carry)/10}$

end for

$r[N] \leftarrow carry$

Integer division
(ignore remaider)

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!

Roman Numeral Table

1	I	14	XIV	27	XXVII	150	CL
2	II	15	XV	28	XXVIII	200	CC
3	III	16	XVI	29	XXIX	300	CCC
4	IV	17	XVII	30	XXX	400	CD
5	V	18	XVIII	31	XXXI	500	D
6	VI	19	XIX	40	XL	600	DC
7	VII	20	XX	50	L	700	DCC
8	VIII	21	XXI	60	LX	800	DCCC
9	IX	22	XXII	70	LXX	900	CM
10	X	23	XXIII	80	LXXX	1000	M
11	XI	24	XXIV	90	XC	1600	MDC
12	XII	25	XXV	100	C	1700	MDCC
13	XIII	26	XXVI	101	CI	1900	MCM

MathATube.com

Grade school subtraction

$$\begin{array}{r} 924 \\ -\ \underline{352} \\ 572 \end{array}$$

How to write an algorithm for doing this?

Grade school subtraction

$$\begin{array}{r} \overset{8}{\cancel{9}}\,{}^{1}24 \\ - \underline{352} \\ 572 \end{array}$$

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 * b ?

(assuming integers)

Multiplication

Q: What do we mean by a * b ?

(assuming integers)

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

a is the "multiplicand"

b is the "multiplier"

Multiplication

Q: What do we mean by a * b ?

(assuming integers)

A: (a + a + ….. + a), b times

or (b + b + … + b), a times

The definition of multiplication suggests a slow algorithm:

$$
\begin{array}{l}
product = 0 \\
\textbf{for } i = 1 \text{ to } b \ \ \textbf{do} \\
\quad product \leftarrow product + a \\
\textbf{end for}
\end{array}
$$

You learned a much faster algorithm in grade school.

Grade school multiplication

$$
\begin{array}{r}
352 \\
\times\ 964 \\
\hline
1\,4^{2}0\,8 \\
2\,1^{3}1^{1}2 \\
3\,1^{4}6^{1}8 \\
\hline
3\,3\,9^{1}3\,2\,8
\end{array}
$$

a[N] “multiplicand”

b[N] “multiplier”

tmp[N] [2N]

r[2N]

Grade school multiplication

Step 1: make 2D table *tmp* [ ][ ]

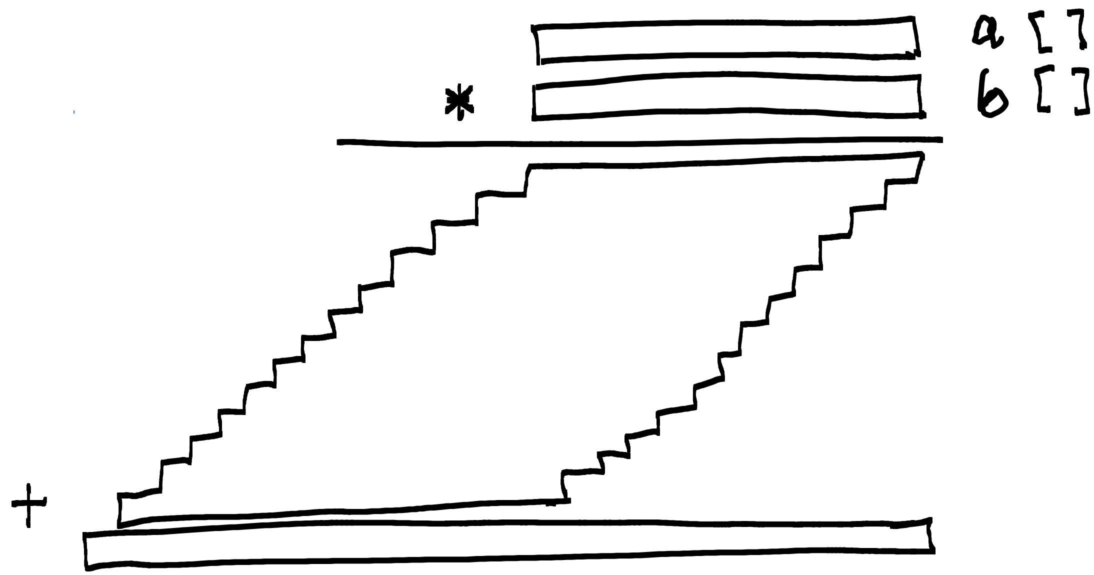
$$
\begin{array}{l}
\textbf{for } j = 0 \text{ to } N-1 \textbf{ do} \\
\quad carry \leftarrow 0 \\
\quad \textbf{for } i = 0 \text{ to } N-1 \textbf{ do} \\
\quad\quad prod \leftarrow (a[i] * b[j] + carry) \\
\quad\quad tmp[j][i+j] \leftarrow prod\%10 \\
\quad\quad carry \leftarrow prod/10 \\
\quad \textbf{end for} \\
\quad tmp[j][N+j] \leftarrow carry \\
\textbf{end for}
\end{array}
$$

Grade school multiplication

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

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

Grade school multiplication specifies that we build a temporary 2D array of size N*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: $a = q*b + r, \quad 0 <= r < b$

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.

$$
\begin{aligned}
& q = 0 \\
& r = a \\
& \textbf{while } r \geq b \textbf{ do} \\
& \quad q \leftarrow q + 1 \\
& \quad r \leftarrow r - b \\
& \textbf{end while}
\end{aligned}
$$

You learned a much faster algorithm in grade school.

Grade school division ("long division")

```
          5 ...
      ________________
723 | 41672542996
      3615
        ----
        552  ...etc
```

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

$$
\begin{array}{ll}
carry = 0 & 1 \\
\textbf{for } i = 0 \text{ to } N - 1 \ \ \textbf{do} & \\
\left.\begin{array}{l}
\quad r[i] \leftarrow (a[i] + b[i] + carry)\ \%\ 10 \\
\quad carry \leftarrow (a[i] + b[i] + carry)/10
\end{array}\right\} & N \\
\textbf{end for} & \\
r[N] \leftarrow carry & 1
\end{array}
$$

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

Grade School Addition

$$
\begin{array}{l}
carry = 0 \\
\textbf{for } i = 0 \text{ to } N-1 \ \ \textbf{do} \\
\quad r[i] \leftarrow (a[i] + b[i] + carry) \ \% \ 10 \\
\quad carry \leftarrow (a[i] + b[i] + carry)/10 \\
\textbf{end for} \\
r[N] \leftarrow carry
\end{array}
$$

c1

c2 * N

c3

The time it takes is c1 + c3 + c2*N for some unspecified constants.

When we analyze algorithms, we often ignore these constants.

Grade School Multiplication

```
for j = 0 to N − 1 do
    carry ← 0                              N
    for i = 0 to N − 1 do
        prod ← (a[i] ∗ b[j] + carry)
        tmp[j][i + j] ← prod%10            N^2
        carry ← prod/10
    end for
    tmp[j][N + j] ← carry                  N
end for
carry ← 0                                  1
for i = 0 to 2 ∗ N − 1 do
    sum ← carry                            N
    for j = 0 to N − 1 do
        sum ← sum + tmp[j][i]              N^2
    end for
    r[i] ← sum%10
    carry ← sum/10                         N
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(…) 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