Winter 2016
COMP-250: Introduction to Computer Science

Lecture 3, January 19, 2016

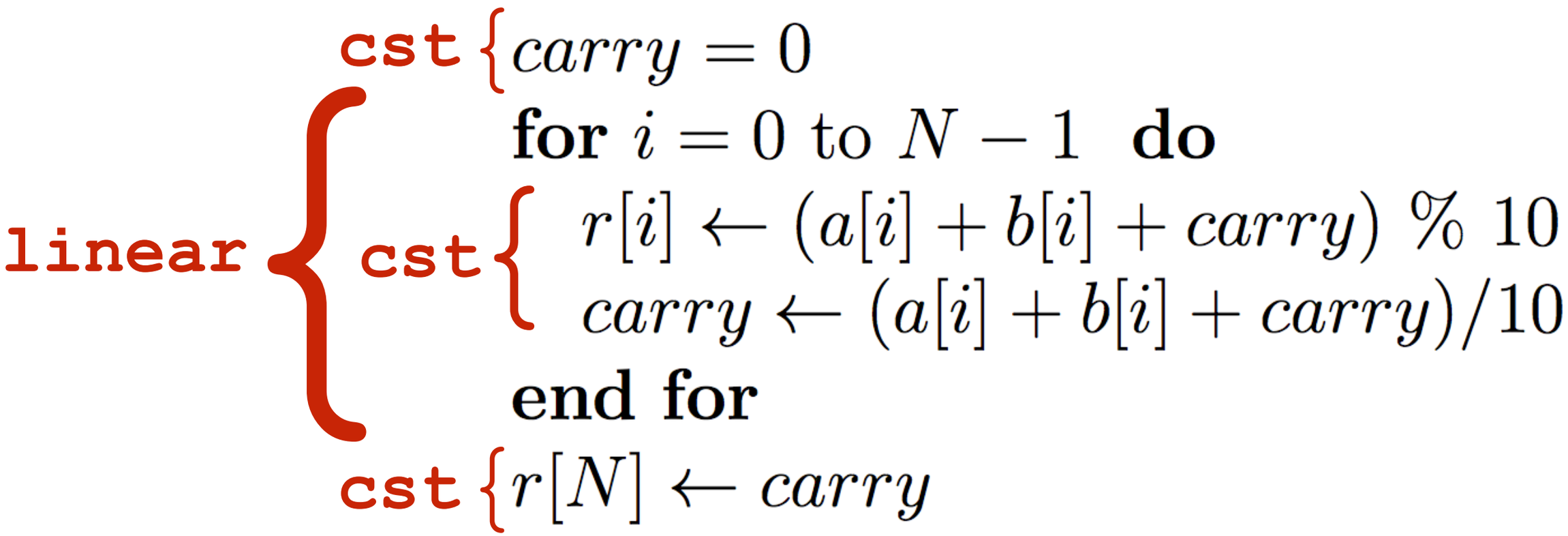
COMP-250 Weekly Schedule

Omar MC-102	Faizy TR-3110	DavidB-R TR-3110	**Thu** 10:00	Chris TR-3110
			Thu 10:30	
			Thu 11:00	
Mon 11:30	**Tue** 11:30	**Wed** 11:30	**Thu** 11:30	**Fri** 11:30
Faiz TR-3110	**Tue** 12:00	DoYeon TR-3110	**Thu** 12:00	Claude MC-110N
	Tue 12:30		**Thu** 12:30	
Mon 13:30	Claude ST-S1/4	DavidB TR-3110	Claude ST-S1/4	
Mon 14:00				
Mon 14:30	**Tue** 14:30		**Thu** 14:30	
Mon 15:00	**Tue** 15:00	**Wed** 15:00	Hardik TR-3110	**Fri** 15:00
Mon 15:30	**Tue** 15:30	**Wed** 15:30		**Fri** 15:30
Mon 16:00	**Tue** 16:00	**Wed** 16:00		**Fri** 16:00

MC=MCENG ST=STBIO TR=TRENG

Analysis of Algorithms

Analysis of Addition



$$\mathtt{Time(N)} = \mathtt{c_1} + \mathtt{c_2} \times \mathtt{N}$$

Analysis of Multiplication

quadratic

$$\textbf{for } j = 0 \text{ to } N-1 \textbf{ do}$$

cst $\{ carry \leftarrow 0$

linear

$$\textbf{for } i = 0 \text{ to } N-1 \textbf{ do}$$

cst $\{ prod \leftarrow (a[i] * b[j] + carry)$

$tmp[j][i+j] \leftarrow prod\%10$

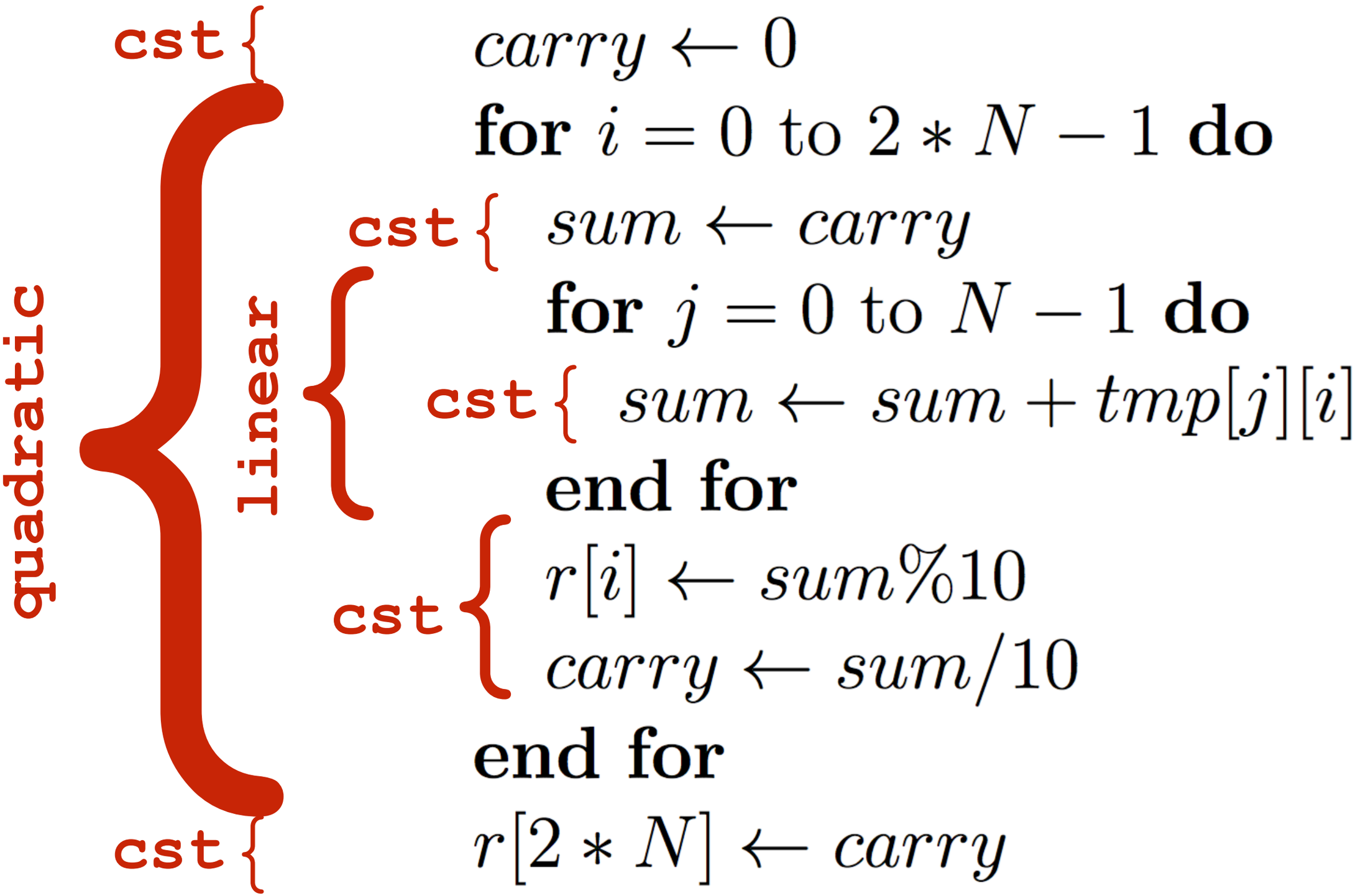
$carry \leftarrow prod/10$

$$\textbf{end for}$$

cst $\{ tmp[j][N+j] \leftarrow carry$

$$\textbf{end for}$$

Analysis of Multiplication



Analysis of Algorithms

Algorithm 2 Multiplication (base 10) of two numbers a and b

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

```
carry ← 0
for i = 0 to 2 ∗ N − 1 do
    sum ← carry
    for j = 0 to N − 1 do
        sum ← sum + tmp[j][i]
    end for
    r[i] ← sum%10
    carry ← sum/10
end for
r[2 ∗ N] ← carry
```

$$Time(N) = c_1 + c_2 \times N + c_3 \times N^2$$

Analysis of Algorithms

Addition

$$Time(N) = c_1 + c_2 \times N$$

Multiplication

$$Time(N) = c_1 + c_2 \times N + c_3 \times N^2$$

Analysis of Algorithms

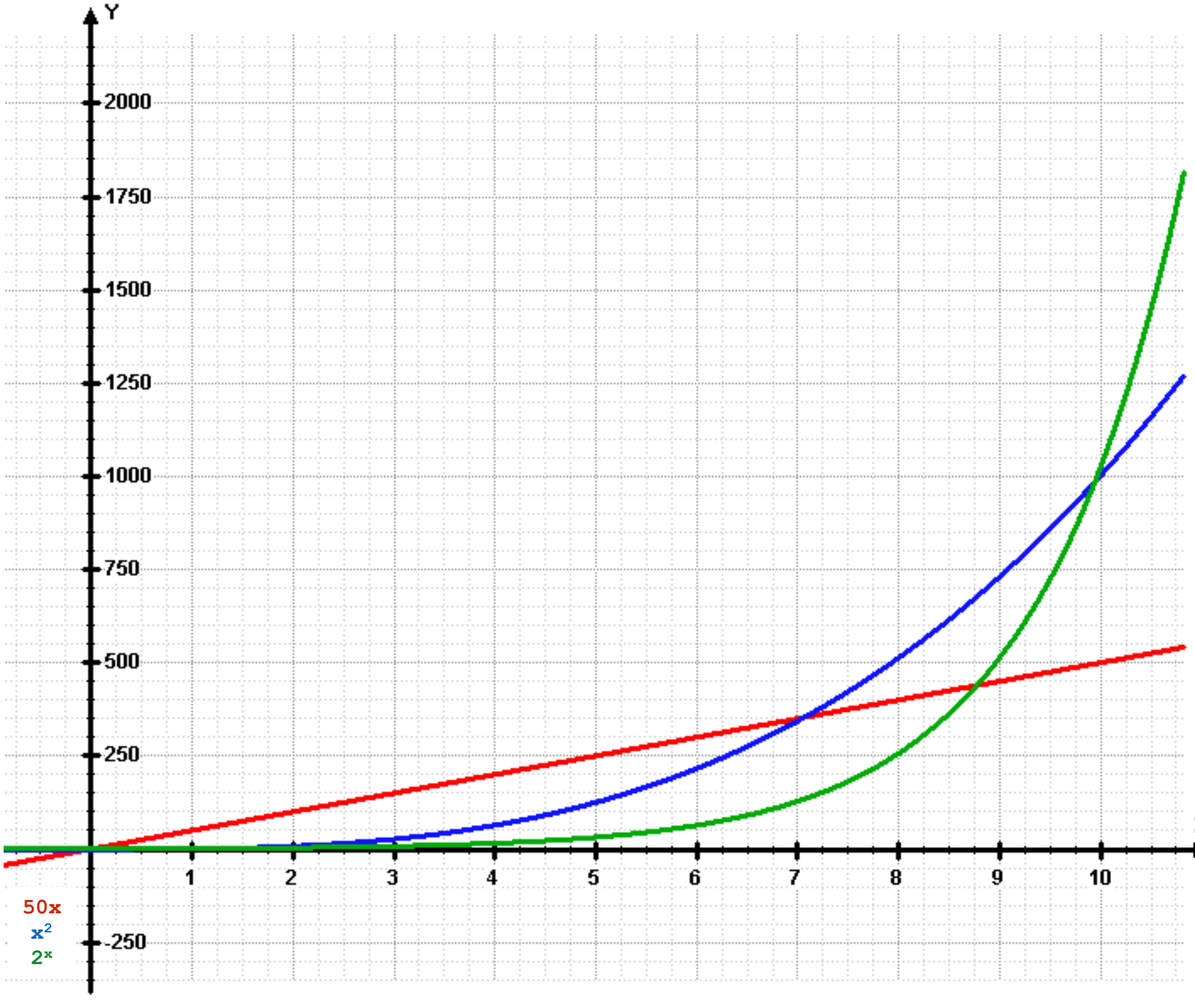
Addition

`Time(N)` **`is`** $O(N)$

Multiplication

`Time(N)` **`is`** $O(N^2)$

Analysis of Algorithms



Unary Representation

Multiplication

Decimal/binary Representation

Addition

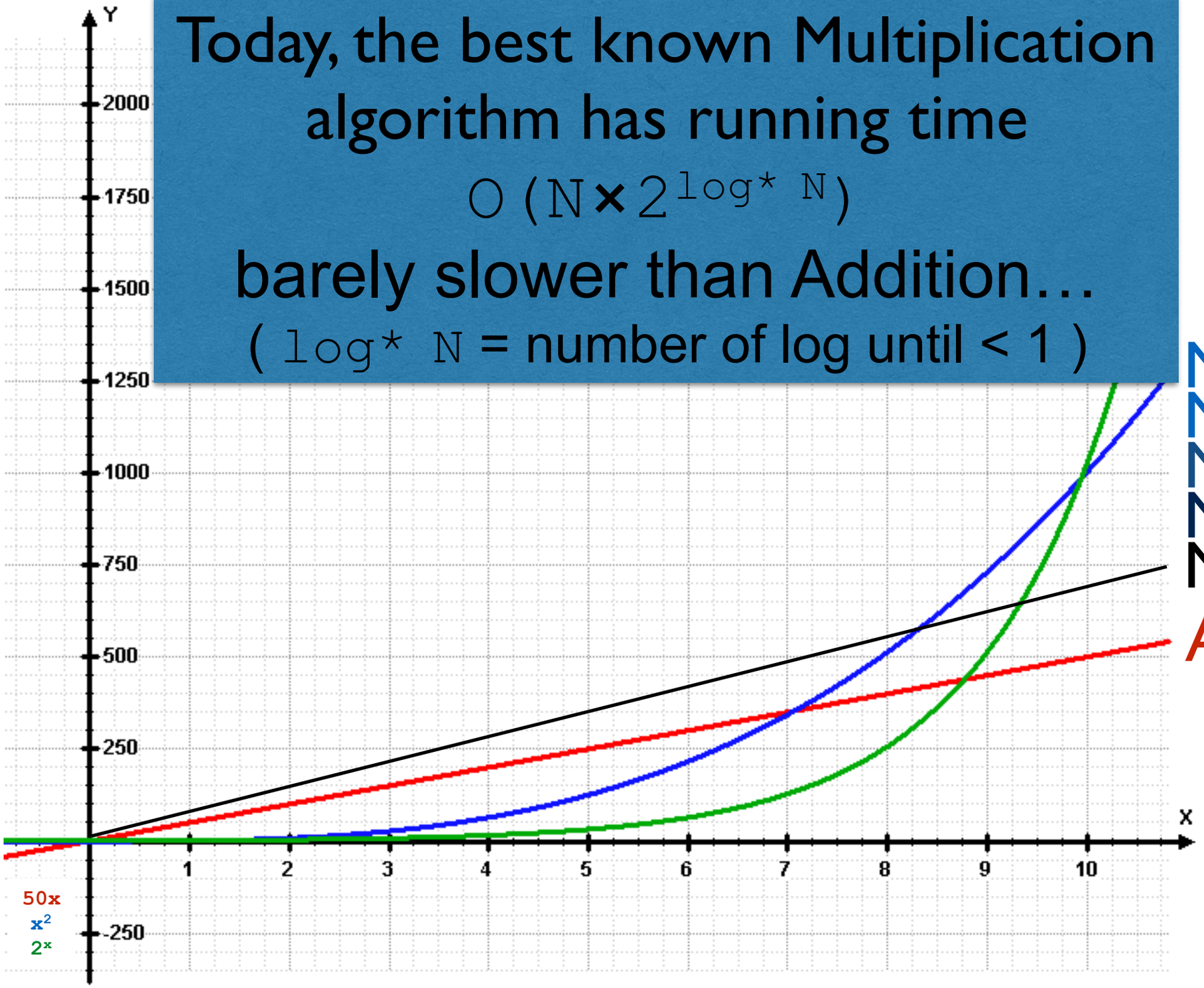
Analysis of Algorithms

Today, the best known Multiplication algorithm has running time

$O(N \times 2^{\log^* N})$

barely slower than Addition…

($\log^* N$ = number of log until < 1)



Multiplication
Multiplication
Multiplication
Multiplication
Multiplication

Addition

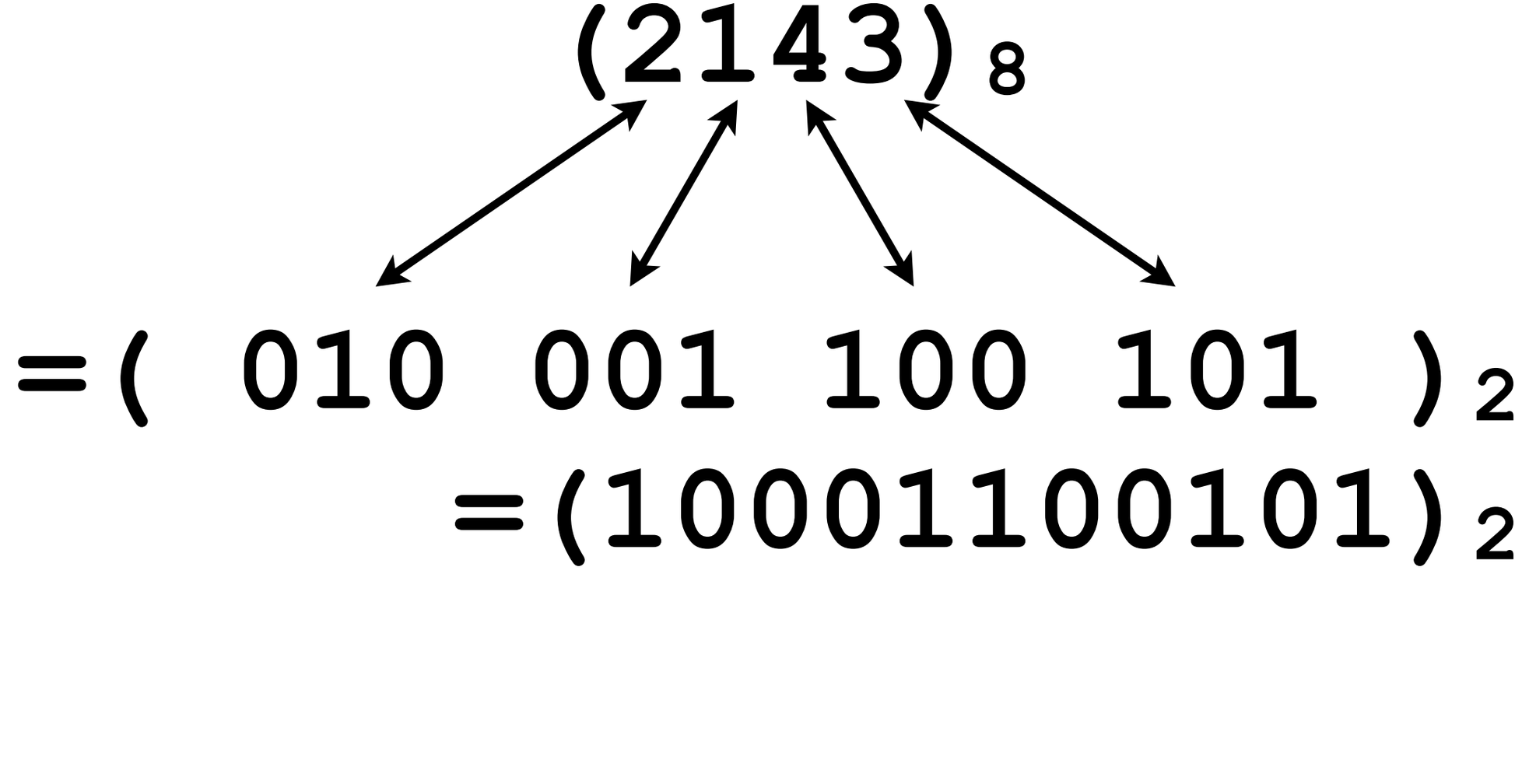
Bases and Binary Representation

Base 8 vs Base 2

$$(2143)_8$$

$$=(????)_2$$

Base 8 vs Base 2



Base 2 vs Base 8

$$(100100101010101)_2$$
$$=(10\ 010\ 010\ 101\ 010\ 101)_2$$
$$=(2\ 2\ 2\ 5\ 2\ 5)_8$$
$$=(222525)_8$$

Base 2 vs Base 10

$$(1001001010101010101)_2$$
$$=(65536+8192+1024+256+64+16+4+1)_{10}$$
$$=(75093)_{10}$$

Powers of 2 in Base 10

$2^0=1$ $2^1=2$ $2^2=4$

$2^3=8$ $2^4=16$ $2^5=32$

$2^6=64$ $2^7=128$ $2^8=256$

$2^9=512$ $2^{10}=1024$ $2^{11}=2048$

$2^{12}=4096$ $2^{13}=8192$ $2^{14}=16384$

$2^{15}=32768$ $2^{16}=65536$

$2^{32} = 4\,294\,967\,296$

Powers of 10 in Base 2

$10^0 = 1$

$10^1 = 1010$

$10^2 = 1100110$

$10^3 = 1111101000 \approx 2^{10}$

$10^4 = 10011100010000$

Base 10 vs Base 2

$$(75093)_{10}$$
$$=(111*1001110001000$$
$$+101*1111101000$$
$$+1001*1010$$
$$+101)_2$$
$$=(10010010101010101)_2$$

to Base 2

Algorithm 3 Convert integer to binary

INPUT: a number m

OUTPUT: the number m **expressed in base 2 using a bit array** $b[\ ]$

$i \leftarrow 0$
while $m > 0$ **do**
  $b[i] \leftarrow m\%2$
  $m \leftarrow m/2$
  $i \leftarrow i + 1$
end while

75093 $=(10010010101010101)_2$

/2	%2
37546	1
18773	0
9386	1
4693	0
2346	1
1173	0
586	1
293	0
146	1

/2	%2
73	0
36	1
18	0
9	0
4	1
2	0
1	0
0	1

why does it work ?

```
m = 2* (m/2) + m%2
```

```
  10010010101010101
= 2*1001001010101010 + 1
```

```
m = β* (m/β) + m%β
```

why does it work ?

$$m = \sum_{i=0}^{n-1} b_i \beta^i$$

$$= (b_{n-1} b_{n-2} \ldots b_1 b_0)_\beta$$

$$= (b_{n-1} b_{n-2} \ldots b_1 0)_\beta + b_0$$

$$= \beta * (b_{n-1} b_{n-2} \ldots b_1)_\beta + b_0$$

$$= \beta * (b_{n-1} b_{n-2} \ldots b_1 b_0 / \beta) + b_0$$

$$m = \beta * (m/\beta) + m \% \beta$$

to Base β

Algorithm 3 Convert integer to binary

INPUT: a number m

OUTPUT: the number m **expressed in base** β **using a bit array** $b[\ ]$

$i \leftarrow 0$

while $m > 0$ **do**

  $b[i] \leftarrow m\%$ β

  $m \leftarrow m/$ β

  $i \leftarrow i + 1$

end while

75093 $=(4400333)_5$

/5	%5
15018	3
3003	3
600	3
120	0
24	0
4	4
0	4

Winter 2016
COMP-250: Introduction to Computer Science

Lecture 3, January 19, 2016