Exam 1: ch1-4

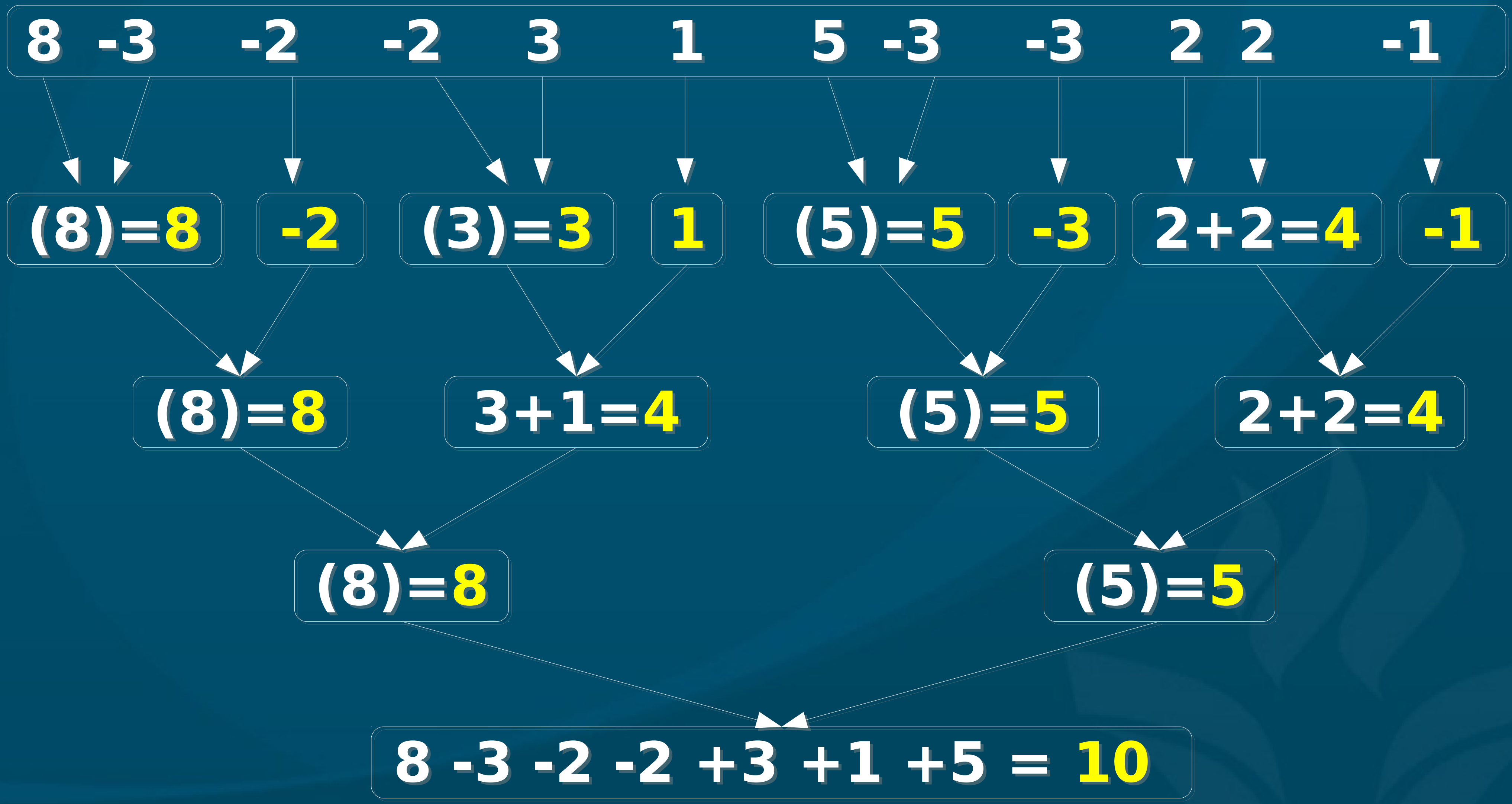
Exam first until 14:30
Open book, paper notes
No electronic devices
Please show your work

2 Oct 2012
CMPT231
Dr. Sean Ho
Trinity Western University

Exam 1: 40pts

- Illustrate the maximum subarray algorithm *(p.72)* on this 12-item array: [8, -3, -2, -2, 3, 1, 5, -3, -3, 2, 2, -1]
- Prove from definition: $n^2 + 2n \lg n = \Theta(n^2)$
- Find functions $f,g \geq 0$ so that $f \neq O(g)$ and $g \neq O(f)$ (prove)
- Solve the following recurrence (with proof) ($T(1) = \Theta(1)$):
  - $T(n) = 2T(n/3) + \Theta(\lg(n^2))$
- Consider the following pseudocode:
  - fib(n):
    - if $n \leq 2$: return 1
    - else: return fib(n-1) + fib(n-2)
  - Write a recurrence for its complexity
  - Solve the recurrence *(hint: p.60 Eq 3.25)*
  - Re-implement the function to perform in $\Theta(n)$

Exam 1: solutions #1 (6pts)



Exam 1: solutions #2 (7pts)

- We want to show $\exists\ n_0, c_1, c_2 > 0: \forall\ n>n_0$,
  - $0 \le c_1 n^2 \le n^2 + 2n \lg n \le c_2 n^2$
- We do this by showing $0n^2 \le 2n \lg n \le 1n^2$:
  - if $n>1$, then $0 \le 2n \lg n$.
  - if $n>5$, then $n^2 \le 2^n$
    $\Rightarrow \lg(n^2) \le n$
    $\Rightarrow 2\lg n \le n$
    $\Rightarrow 2n \lg n \le n^2$.
- Thus, with $n_0=5$, $c_1=1$, $c_2=2$, we have: $\forall\ n>5$,
  - $0 \le 1n^2 \le n^2 + 2n \lg n \le 2n^2$.
- Hence $n^2 + 2n \lg n \in \Theta(n^2)$.

Exam 1: solutions #3 (8 pts)

- Many possible choices here; one technique is to ensure that f and g take turns touching zero:
  - e.g., $f(n) = n(1 + (-1)^n)$, $g(n) = n(1 - (-1)^n)$.
  - Both f and g are in $O(n)$, but
  - When n is odd, $f(n)=0$, and $g(n)=2n$
  - When n is even, $f(n)=2n$, and $g(n)=0$.
- So for any n_0, $c > 0$:
  - Any odd $n>n_0$ will have $g(n) = 2n > 0 = cf(n)$,
    - (And thus $g \notin O(f)$)
  - Any even $n>n_0$ will have $f(n) = 2n > 0 = cg(n)$,
    - (And thus $f \notin O(g)$)

Exam 1: solutions #4 (7pts)

- Master method case 1 (according to textbook):
  - $f(n) = \lg(n^2) = 2\lg n$
  - $\Theta(n^{\log\_b(a) - \varepsilon}) = \Theta(n^{\log\_3(2) - \varepsilon})$.
    - For any $\varepsilon < \log_3(2)$ (e.g., $\varepsilon = 0.5$), this is still a polynomial
- Any polylog is slower than any polynomial:
  - $\forall\ a, b>0$, $\lg^b n = o(n^a)$ *(see p.57)*
- Hence $2\lg n \in o(n^{\log\_3(2) - \varepsilon}) \subset O(n^{\log\_3(2) - \varepsilon})$, so Case 1 applies.
- The solution is $T(n) = \Theta(n^{\log\_3(2)})$.

Exam 1: solutions #5 (12 pts)

- Recurrence: $T(n) = T(n-1) + T(n-2) + \Theta(1)$ for $n>2$, and $T(n) = \Theta(1)$ for $n \le 2$
- Solution: $T(n) = \Theta(\text{Fib}(n))$
  - By Eq. 3.25, $T(n) = \Theta(|\varphi|^n)$ (exponential)
- A $\Theta(n)$ solution:
  - fib(n):
    - if n ≤ 2: return 1
    - (pa, gdpa) = (1, 1)
    - for i in 3 .. n:
      - cur = pa + gdpa
      - gdpa = pa
      - pa = cur
    - return cur

A $\Theta(1)$ solution:

```
fib(n):
    phi = (1+ sqrt(5))/2
    return pow(phi, n)
```