+ big O, R, +
+ Heaps
+ BST
+ loop invariants
+ code analysis
+ Stock, Shares
+ pseudo
\[ f(n) = 15n^4 + 3n^3 + n^2 \log n + \log \log n. \]

\[ f(n) = O(n^4) \quad \text{true} \]

\[ = O(n^3) \quad \text{false} \]

\[ = O(n^3 \log n) \quad \text{false} \]

\[ = O(n^5) \quad \text{true} \]

\[ \log \log n < \log n < \log n \leq n^2 \]

\[ \log = o(n) = o(2^n) \]

\[ \exists c > 0, \forall N, \forall n > N: \]

\[ f(n) < cn^4 \iff 15n^4 + 3n^3 + n^2 \log n + \log \log n \leq cn^4. \]

\[ c = 15 + 3 + 1 + 1 = 20 \]

\[ N = 1. \]

\[ \forall n > N: \]

\[ 15n^4 \leq 15n^4 \leq 15n^4 \]

\[ 3n^3 \leq 3n^4 \]

\[ n^2 \log n < n^4 \]

\[ \log n < n^2 \]

\[ f(n) < 20n^4 = cn^4 \]
Build-Max-heap for

Start: \( \left\lfloor \frac{n}{2} \right\rfloor = \left\lfloor \frac{9}{2} \right\rfloor = 4 \)

\{ for \( i = 4 \) down to 1: \}

\text{Max-heapify}(A, i)

\( i = 4 \cdot \)

\( i = 3 \cdot \)
Build max heap with array representation:

1 2 3 4 5 6 7 8 9

7 8 9 10 11 12 13 14 15

i = 4

children: 2 4 = 8
2 4 5 = 9

i = 3

children 6 7

i = 2

children 4 5

i = 1

children 2 3

15 14 13 12 11 9 8 10

4 = 8, 9

15 14 13 12 11 9 8 10

4 = 8, 9

15 14 13 10 11 12 9 8

done.
loop invariant
\[ \varphi : P(i,j) = \begin{cases} \text{true} & i \leq 10, \; i+j = 9 \\ \text{false} & \text{otherwise} \end{cases} \]

Initially:
\[ i = 0, \quad j = 0 \]

When:
\[ i < 10 \]

\[ i++ \]

\[ j = j - 1 \]

\[ j = 0 \]

\[ i+j = 9 \]

\[ i \geq 0 \quad \text{and} \quad i < 10 \] (\( \varphi \) is true)

\[ j = -1 \]

\[ i \leq 10 \] (\( \varphi \) is true)

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i \geq 10 \] (\( \varphi \) is false)

\[ j = 0 \]

\[ i+j = 9 \] (\( \varphi \) is false)

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 10 \]

\[ i+j = 9 \] (\( \varphi \) is false)

\[ i = 9 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 8 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 7 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 6 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 5 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 4 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 3 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 2 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 1 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = 0 \]

\[ i+j = 9 \] (\( \varphi \) is true)

\[ i = -1 \]

\[ i+j = 9 \] (\( \varphi \) is true)

Result:
\[ j = -1 \]
Stack:

Remember:

When a stack is implemented by an array and we apply the size doubling technique (doubling the size of the array when it is full), then a sequence of $n$ insertions into an empty stack would take $O(n)$ (linear time).

See doc cam file Jan-16.