CS 225

Data Structures

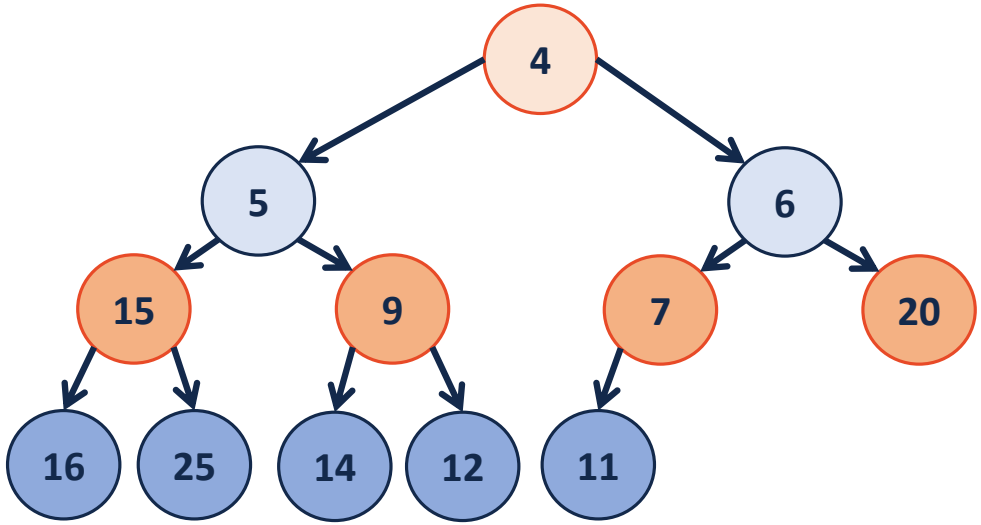
March 8 – Build Heap

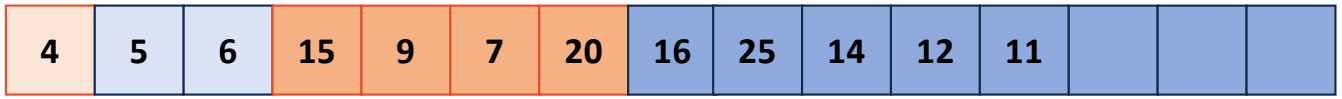
G Carl Evans

(min)Heap

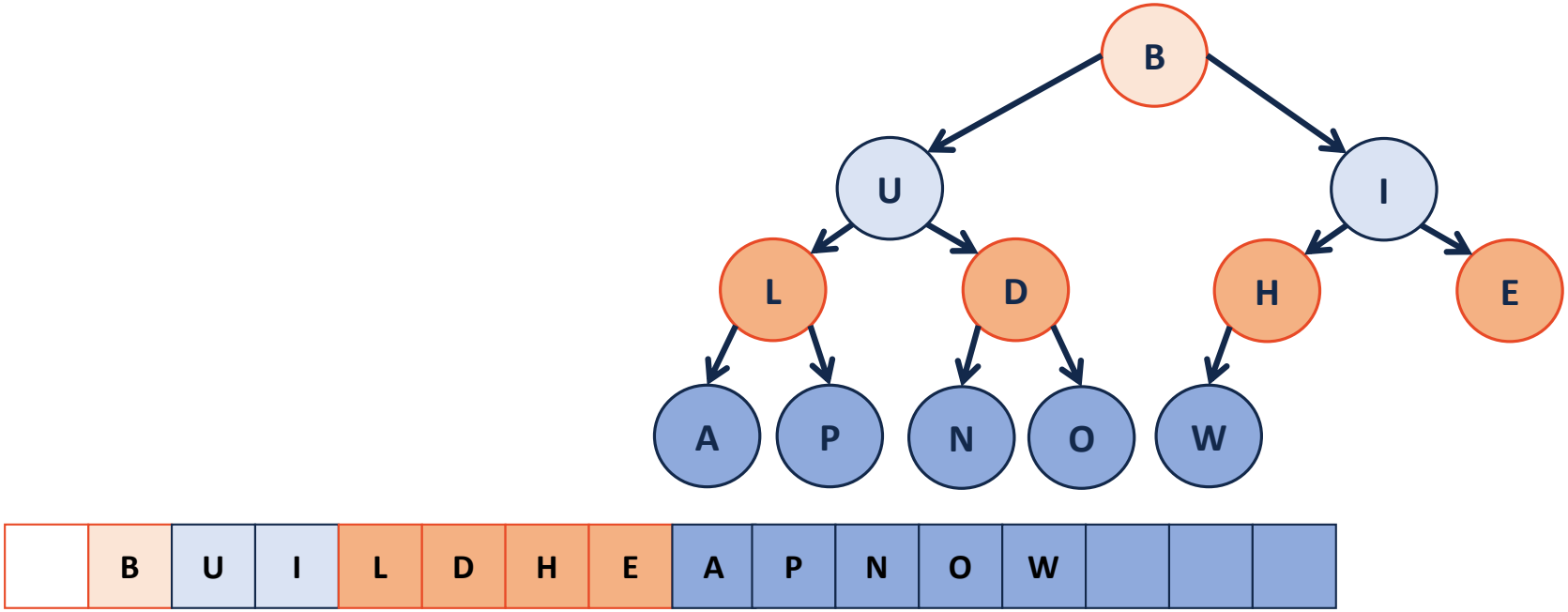
A complete binary tree T is a min-heap if:

- **T = {}** or
- **T = {r, T_L, T_R}**, where **r** is less than the roots of {T_L, T_R} and {T_L, T_R} are min-heaps.

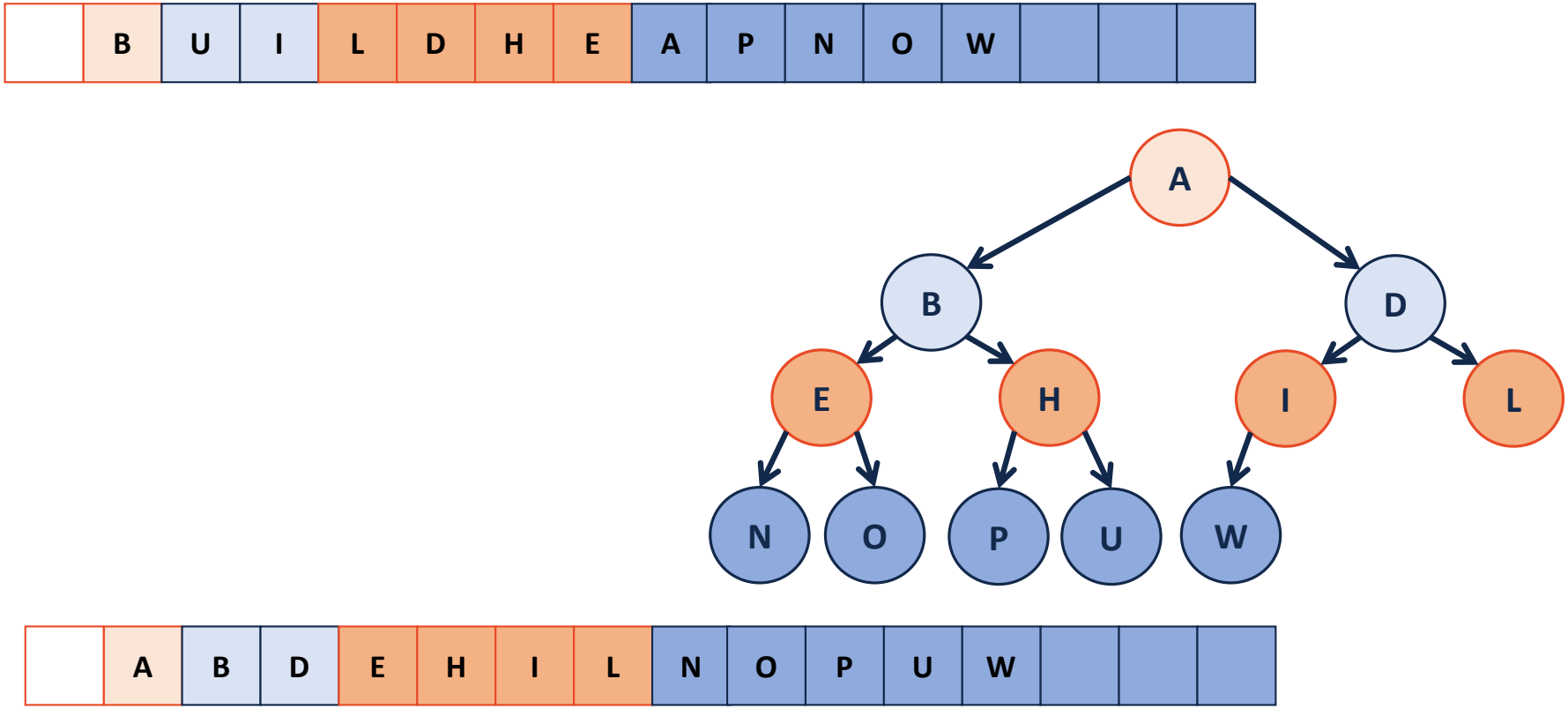




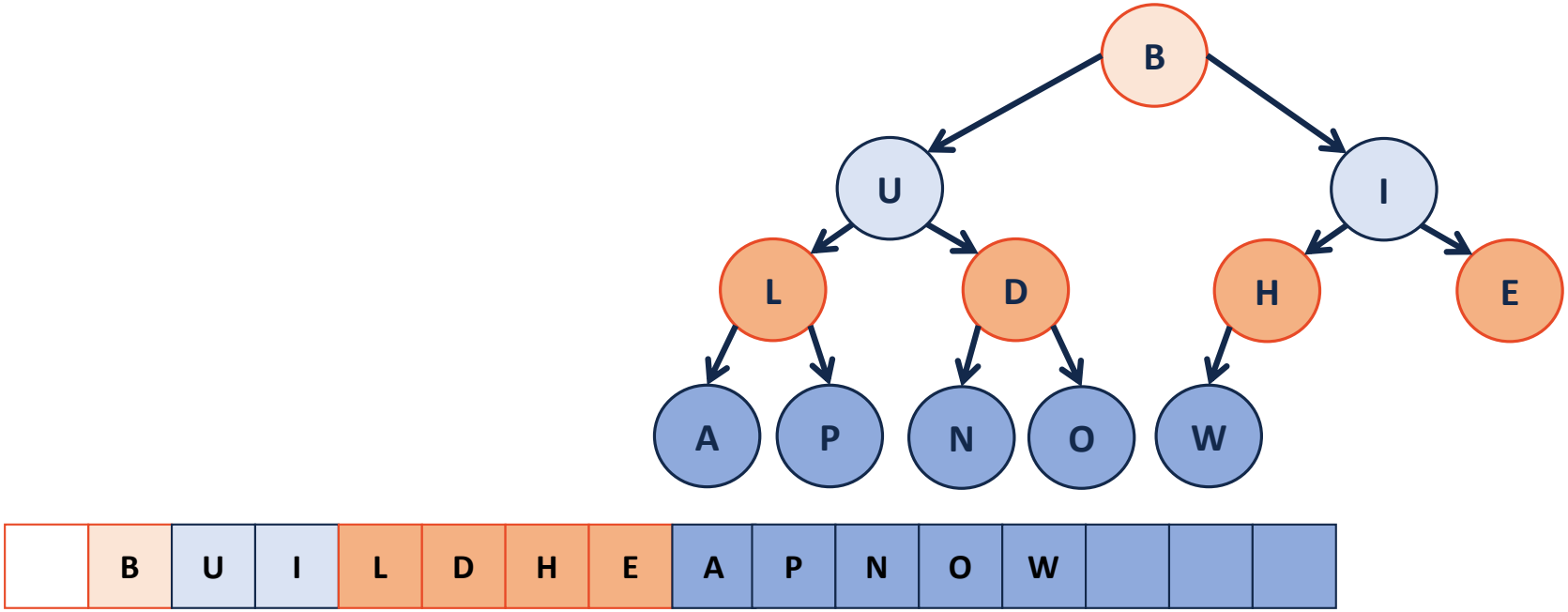
buildHeap



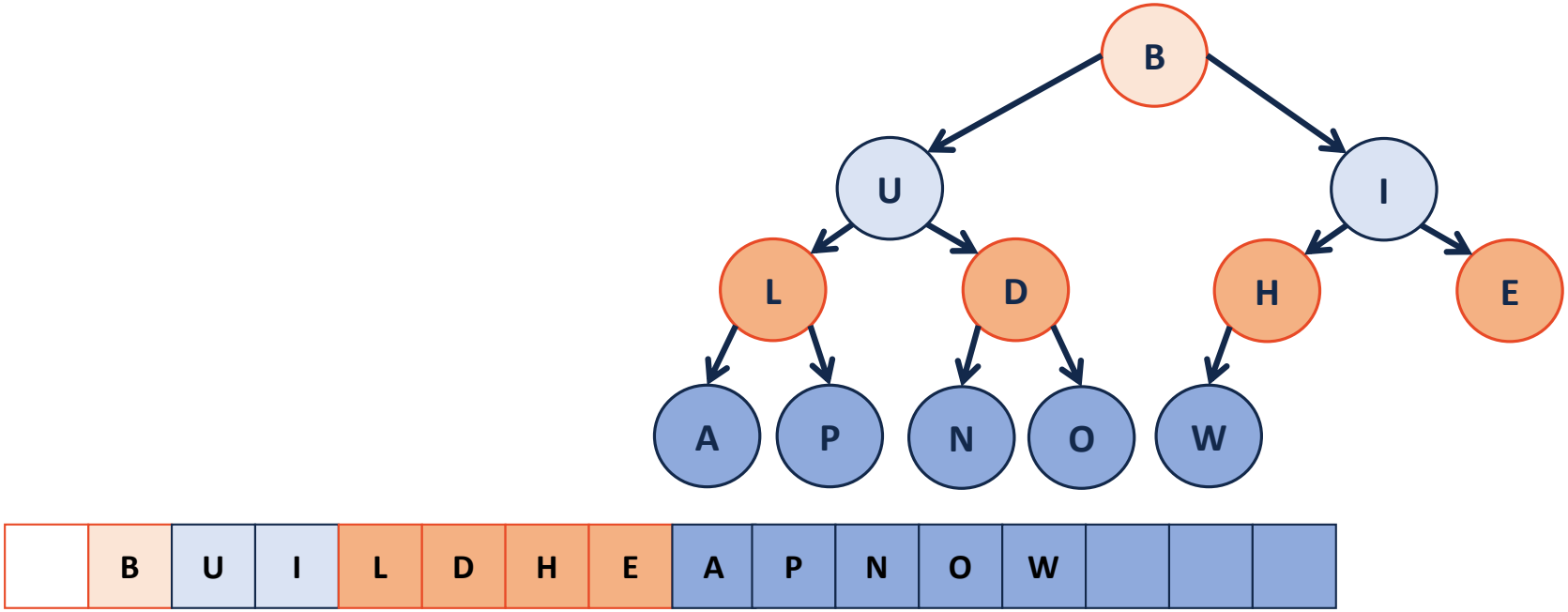
buildHeap – sorted array



buildHeap - heapifyUp

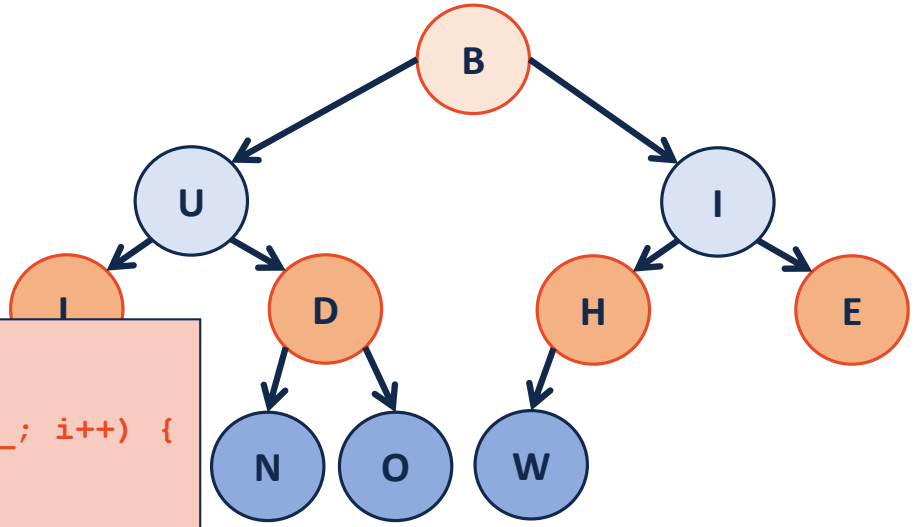


buildHeap - heapifyDown



buildHeap

1. Sort the array – it's a heap!



2.

```
1  template <class T>
2  void Heap<T>::buildHeap() {
3    for (unsigned i = 2; i <= size_; i++) {
4      heapifyUp(i);
5    }
6  }
```

3.

```
1  template <class T>
2  void Heap<T>::buildHeap() {
3    for (unsigned i = parent(size); i > 0; i--) {
4      heapifyDown(i);
5    }
6  }
```

Proving buildHeap Running Time

Theorem: The running time of buildHeap on array of size **n** is: _____.

Strategy:

-

-

-

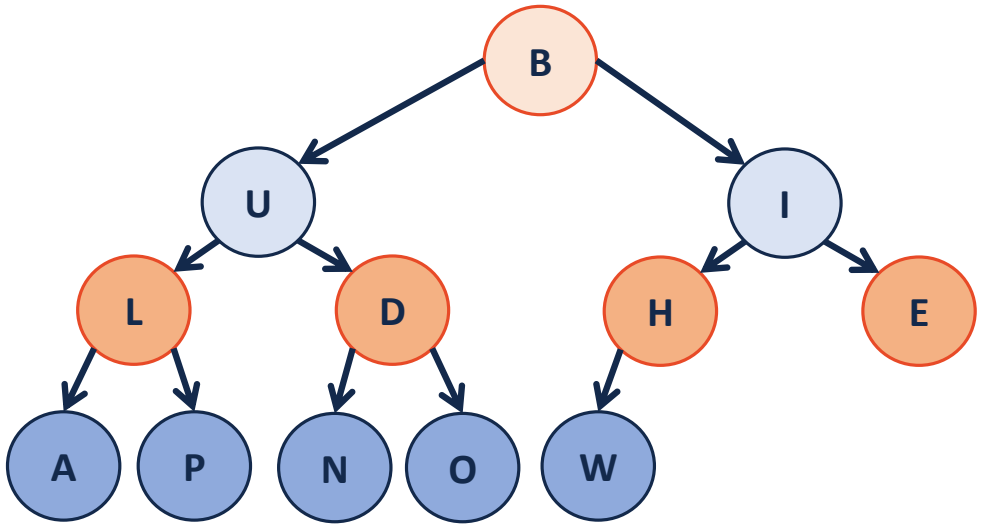
Proving buildHeap Running Time

S(h): Sum of the heights of all nodes in a complete tree of height **h**.

S(0) =

S(1) =

S(h) =



Proving buildHeap Running Time

Proof the recurrence:

Base Case:

General Case:

Proving buildHeap Running Time

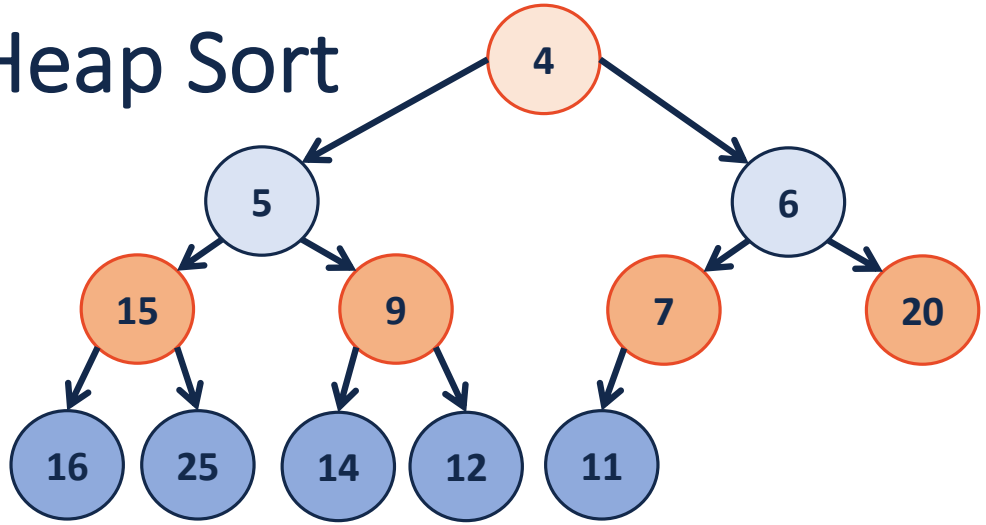
From S(h) to RunningTime(n):

S(h):

Since $h \leq \lg(n)$:

RunningTime(n) ≤

Heap Sort



1.

2.

3.

Running Time?