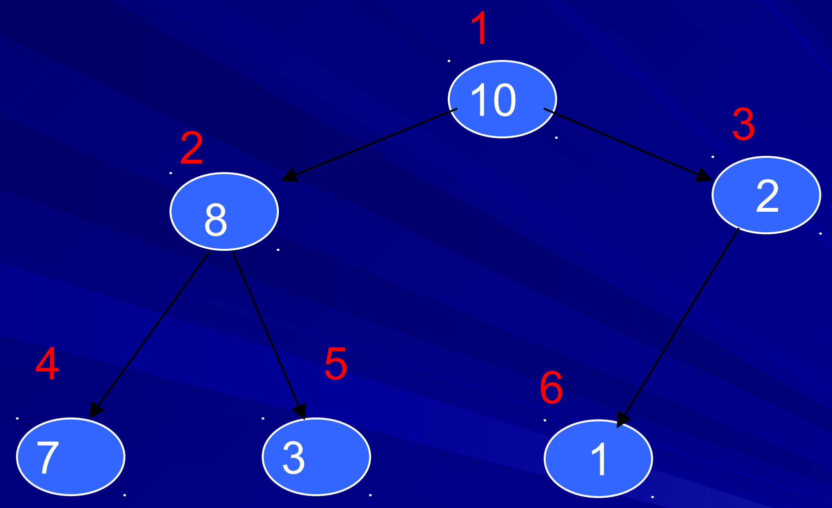
Heaps

Outline

- Heaps
- Permutations and Combinations
- Binomial Heaps
- Fibonacci Heaps

Binary Heap

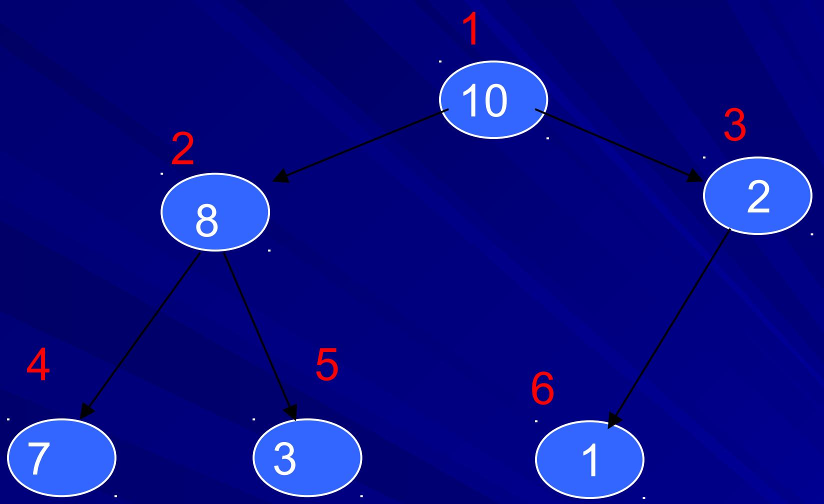
- A binary heap (or just heap) is an array that can be viewed as a nearly complete binary tree.



- A[1]=10, A[2]=8, A[3]=2, A[4]=7, A[5]=3, A[6]=1. So A=[10,8,2,7,3,1].

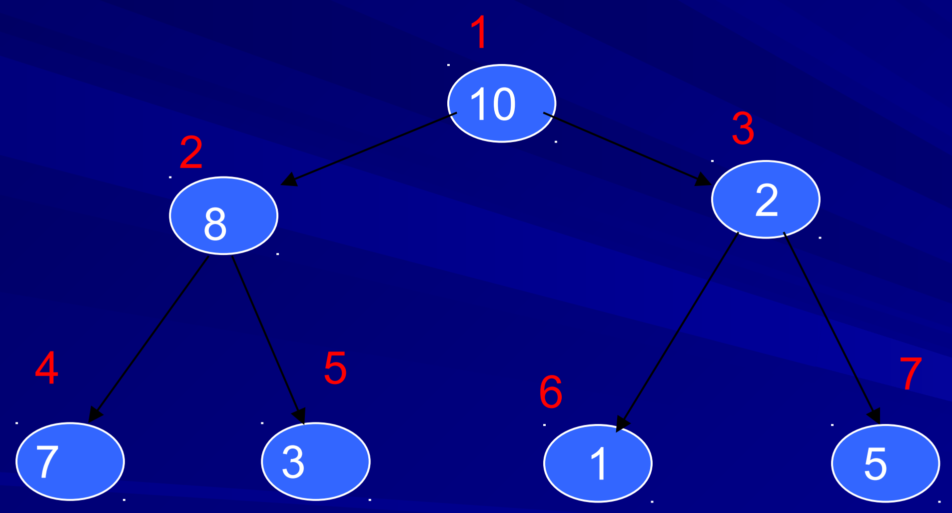
Heap

- Parent(4)=2
- Parent (5)=2
- Parent (6)=3
- Indexes & Keys
  - i ->index;
  - parent(i) -> index;
  - A[i]->key
- What is the parent(i)?
- What is the left_child(i)?
- What is the right_child(i)?
- This is a **max-heap**, i.e. A[parent(i)]>=A[i] for all i.
- Min heap, A[parent(i)]<=A[i] for all i.



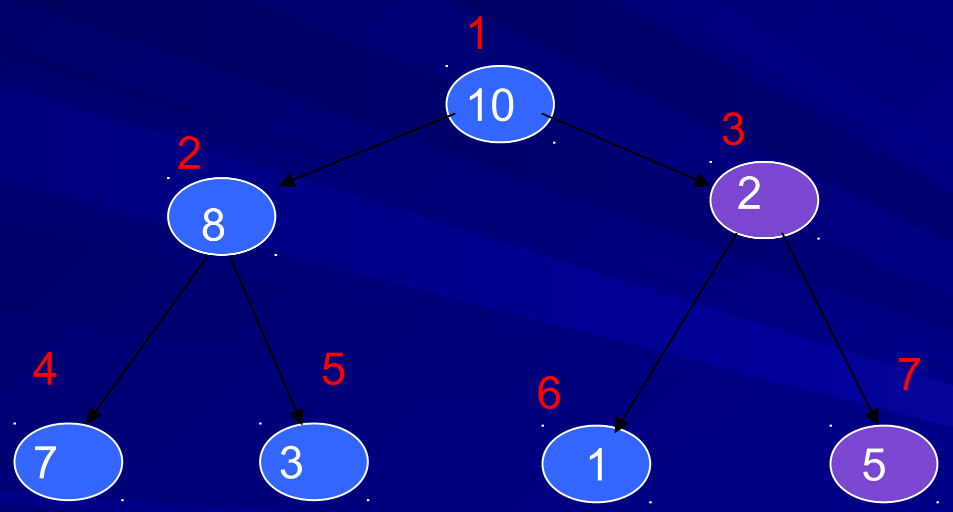
Insert(H,x)

- Suppose we want to insert a 5 into the heap. Insert(H,5)



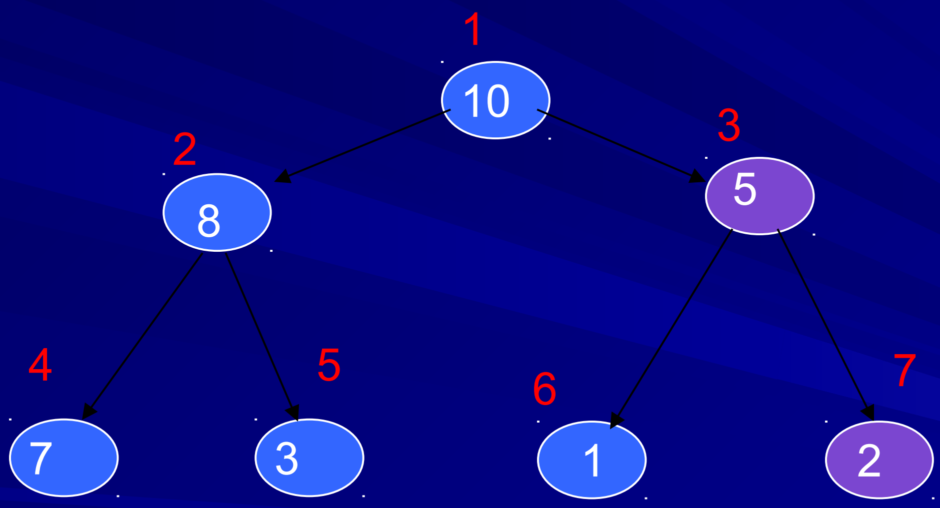
Insert(H,x)

- The next index would be 7



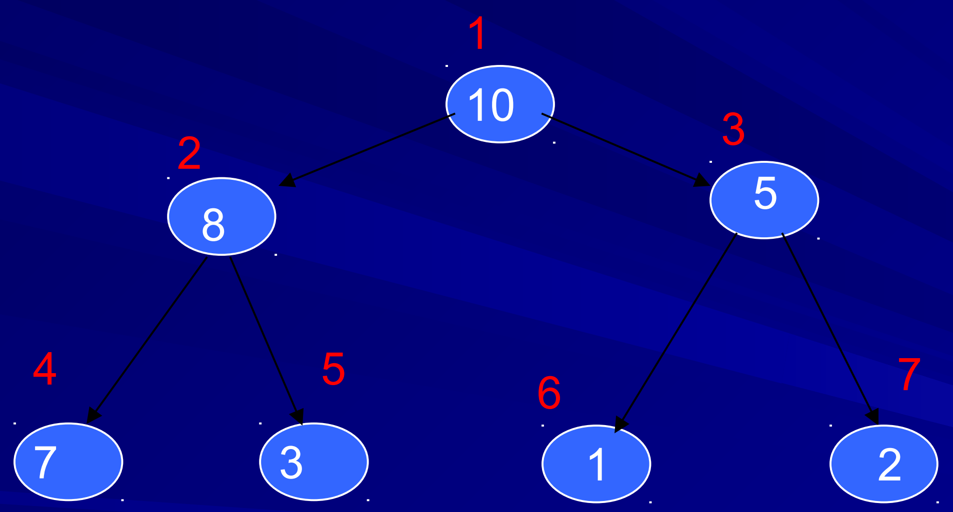
Insert(H,x)

- A[i]>A[parent[i], exchange
- Set i=parent(i) and repeat



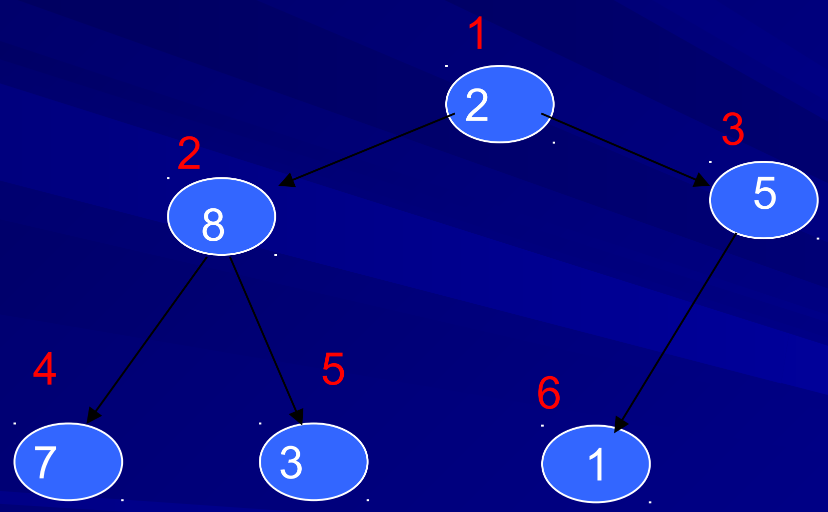
Extract-Max(H)

- Deletes the root node



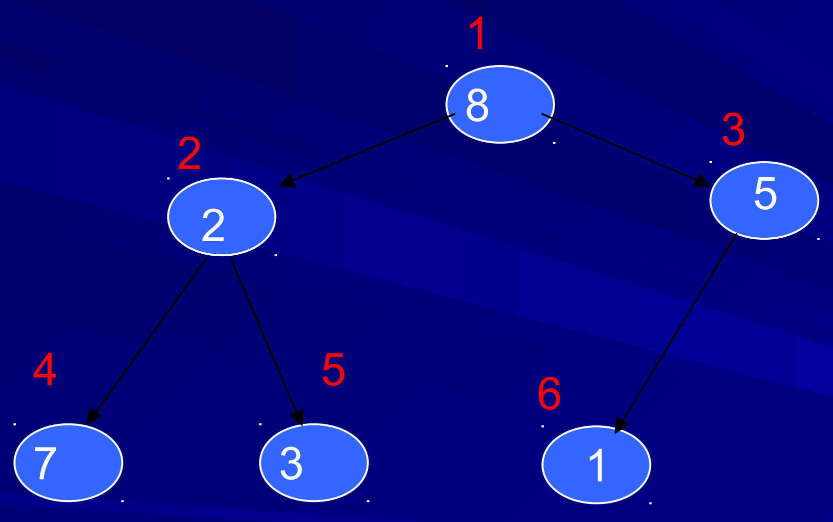
Extract-Max(H)

- Exchange i with child with largest key



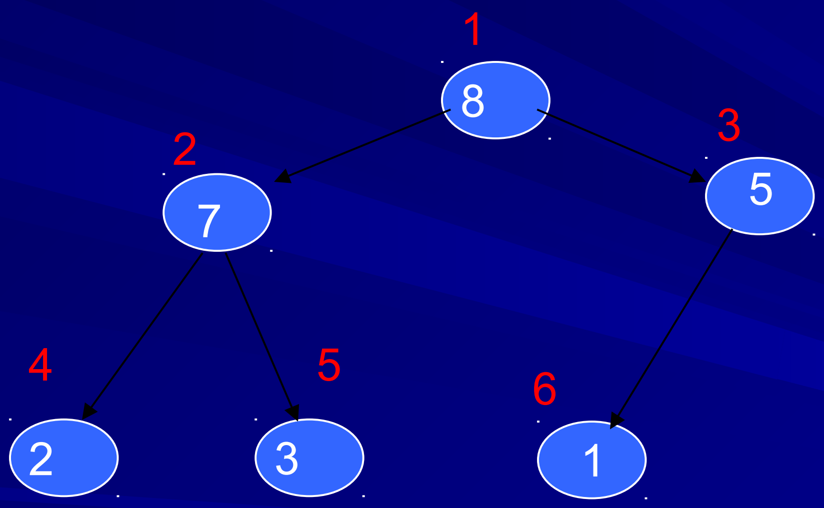
Extract-Max(H)

- Exchange i with child with largest key



Extract-Max(H)

- Exchange i with child with largest key



Quiz 4 part 1

Permutations

- A ***Permutation*** on a set is a ordered arrangement.
- Example S={1,2,3}. There are 6 permutations
  - 1,2,3
  - 1,3,2
  - 2,1,3
  - 2,3,1
  - 3,1,2
  - 3,2,1

Permutations

- If |S|=n, then there are n! permutations of S.
- Let S={1,2,3,4}.
  - The 2-permuations of S are:
    - 1,2 1,3, 1,4 2,1 2,3 2,4 3,1 3,2 3,4
    - 4,1 4,2 4,3
    - 12 total permutations 4*3

Permutations

- The number of r-permutations of a set with n distinct elements,
  - $_{n}P_{r} = n(n-1)(n-2)\ldots(n-r+1) = n!/r!$
- How many ways can you select a first place, second place and third place winner from a group of 30 people?
  - $_{30}P_{3} = 30*29*28=24360$

Permutations

- How many permutation of the letters ABCDEFG contain
  - the string BCD
    - 120
  - The string CFGA
    - 24
  - The strings BA and GF
    - 120
  - The strings ABC and DE
    - 24
  - The strings ABC and CDE
    - 6
  - The strings CBA and BED
    - 0

Combinations

- For combinations, order does not matter.
- For S={1,2,3,4} the 3-combinations are
  - {1,2,3}
  - {1,2,4}
  - {1,3,4}
  - {2,3,4}
- So ${}_4C_3=4$

Combinations

- **The number of r-combinations of a set with n elements, 0<=r<=n is**
  - $_{n}C_{r}$ = n! / [r! (n-r)!]
  - $_{4}C_{3}$ = 4!/ (3! 1!)= 24/6=4

Combinations

- How many 5-card hards are there from a deck of 52 cards?
  - $_{52}C_{5}$ = 2,598,960
- How many bit strings of length eight contain exactly three 1's?
  - $_{8}C_{3}$ = 56

Combinations

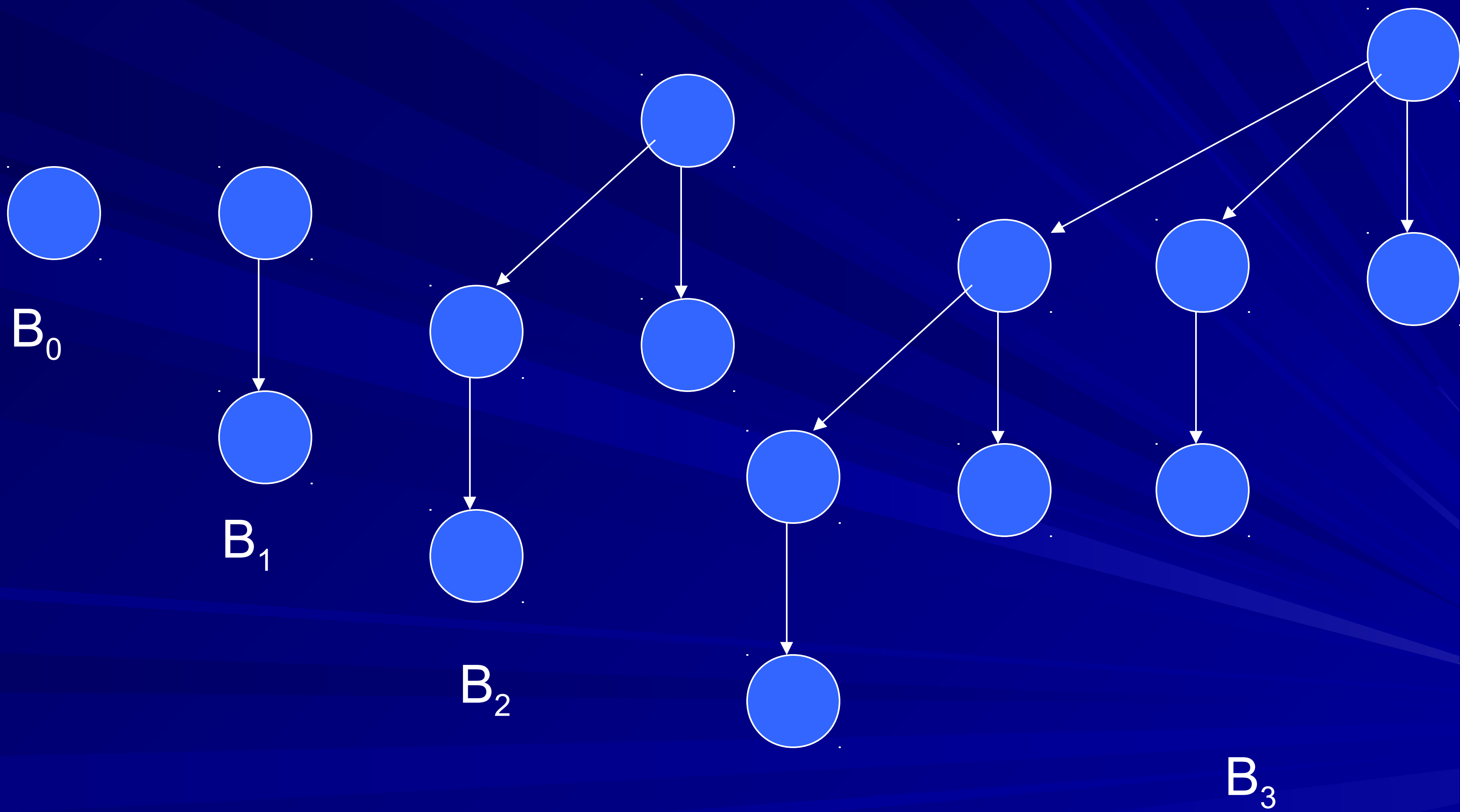
- How many bit strings of length ten contain
  - exactly four 1s?
    - ${}_{10}C_{4}=210$
  - at most four 1s
    - 386
  - at least four 1s
    - 848
  - an equal number of 0s and 1s?
    - ${}_{10}C_{5} = 252$

Quiz 4 part 2

Binomial Heaps

- A collection of binomial trees.
  - Each tree has the min heap property
- B_0 is a single node
- B_k consists of two binomial B_{k-1} linked together
- B_k has 2^k nodes
- The height of B_k is k.
- There are exactly (K C i)=K!/(K-i)!i! nodes at depth i, for i=0,….k
- The root has k children, called the rank.

Binomial Heap



Binomial Heaps

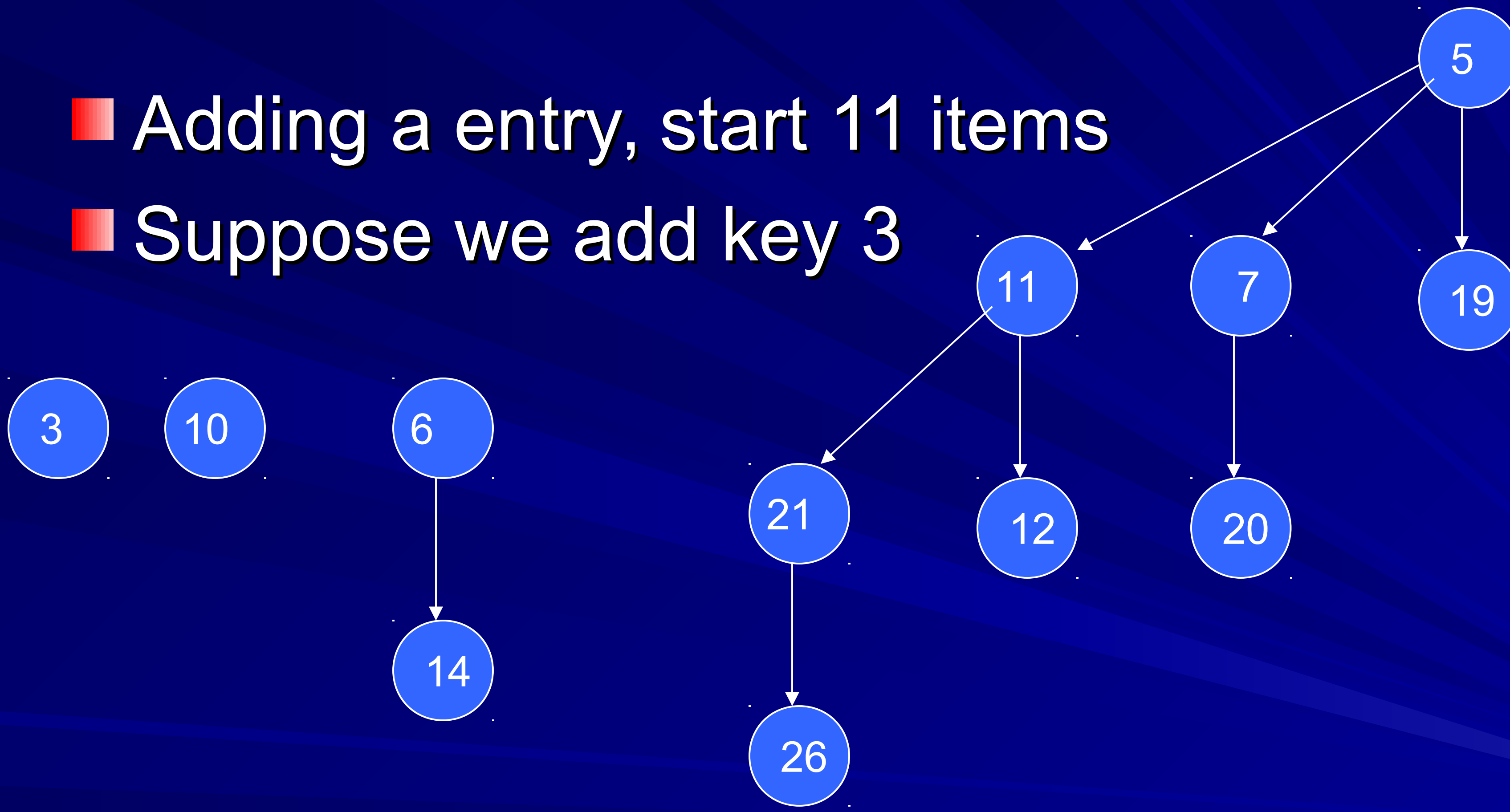
- A binomial heap has at most floor(lg n)+1 trees.
- For example when n=11
  - 1011 binary
  - We have a B_3 B_1 and B_0.
  - Has B_3 has 8 nodes, B_1 has 2, and B_0 has 1
  - Each tree has the min heap property

Insert

- Adding a entry, start 11 items
- Suppose we add 3

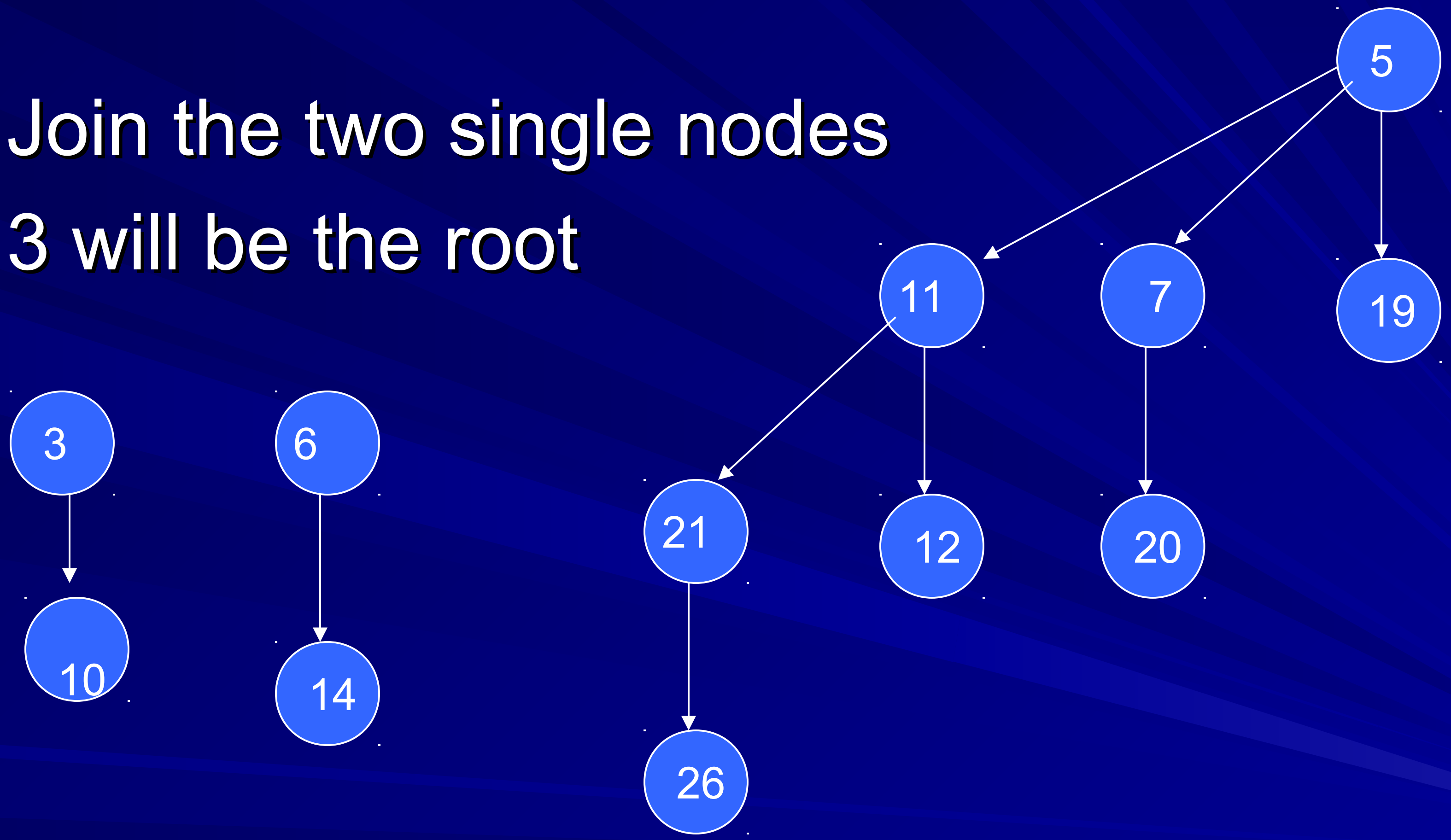
Insert

- Adding a entry, start 11 items
- Suppose we add key 3



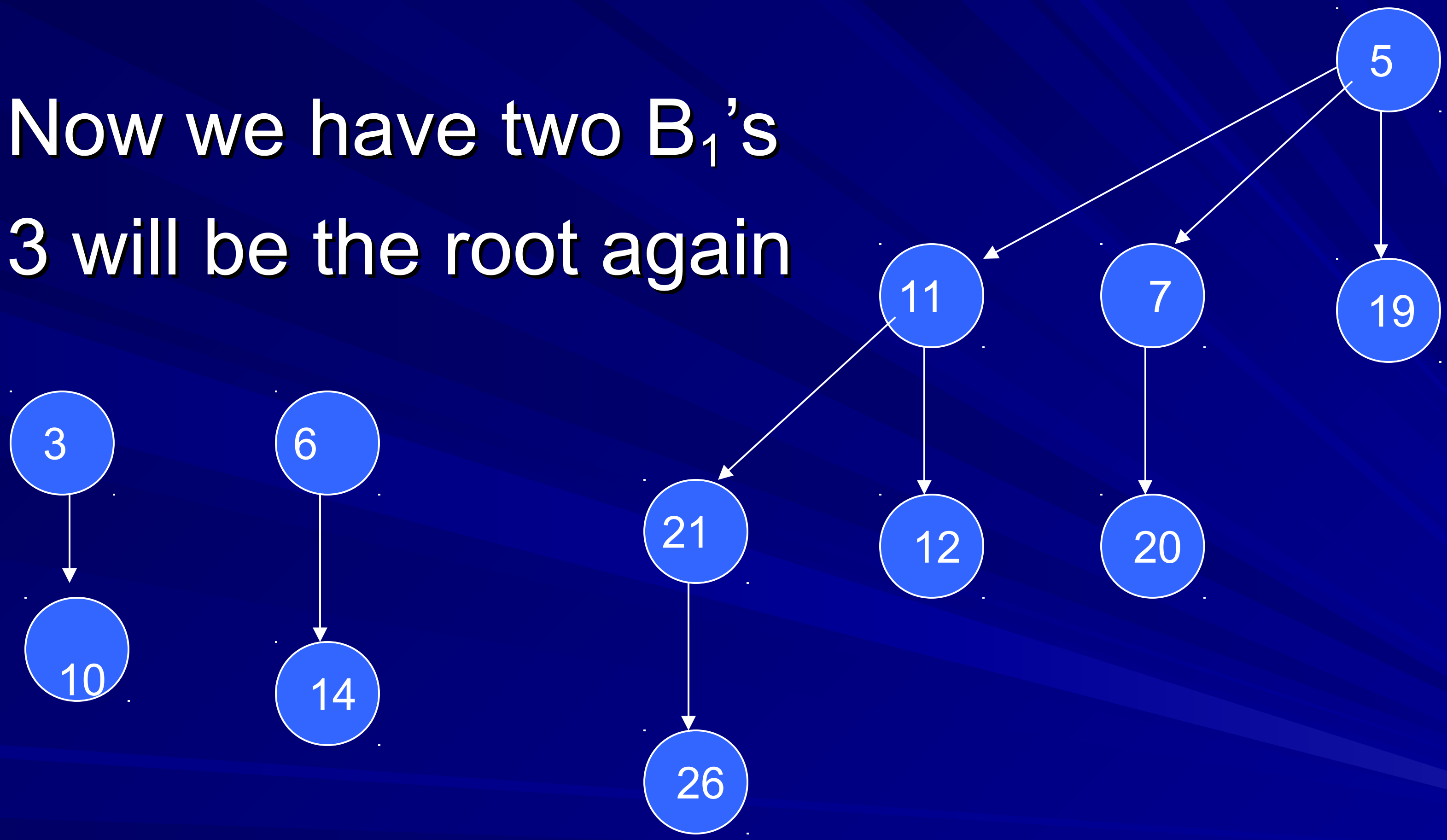
Insert

- Join the two single nodes
- 3 will be the root



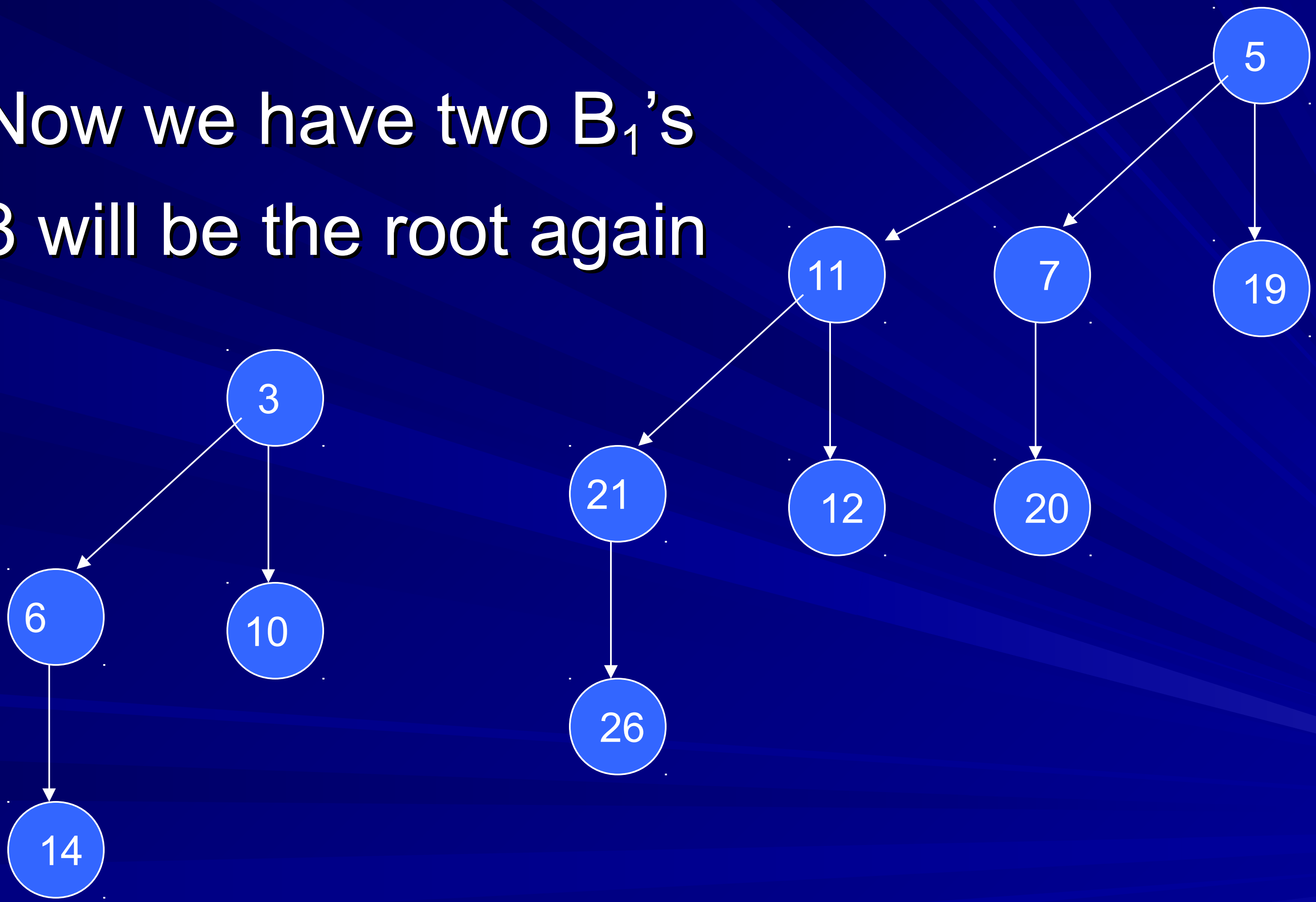
Insert

- Now we have two B_1's
- 3 will be the root again



Insert

- Now we have two B_1's
- 3 will be the root again



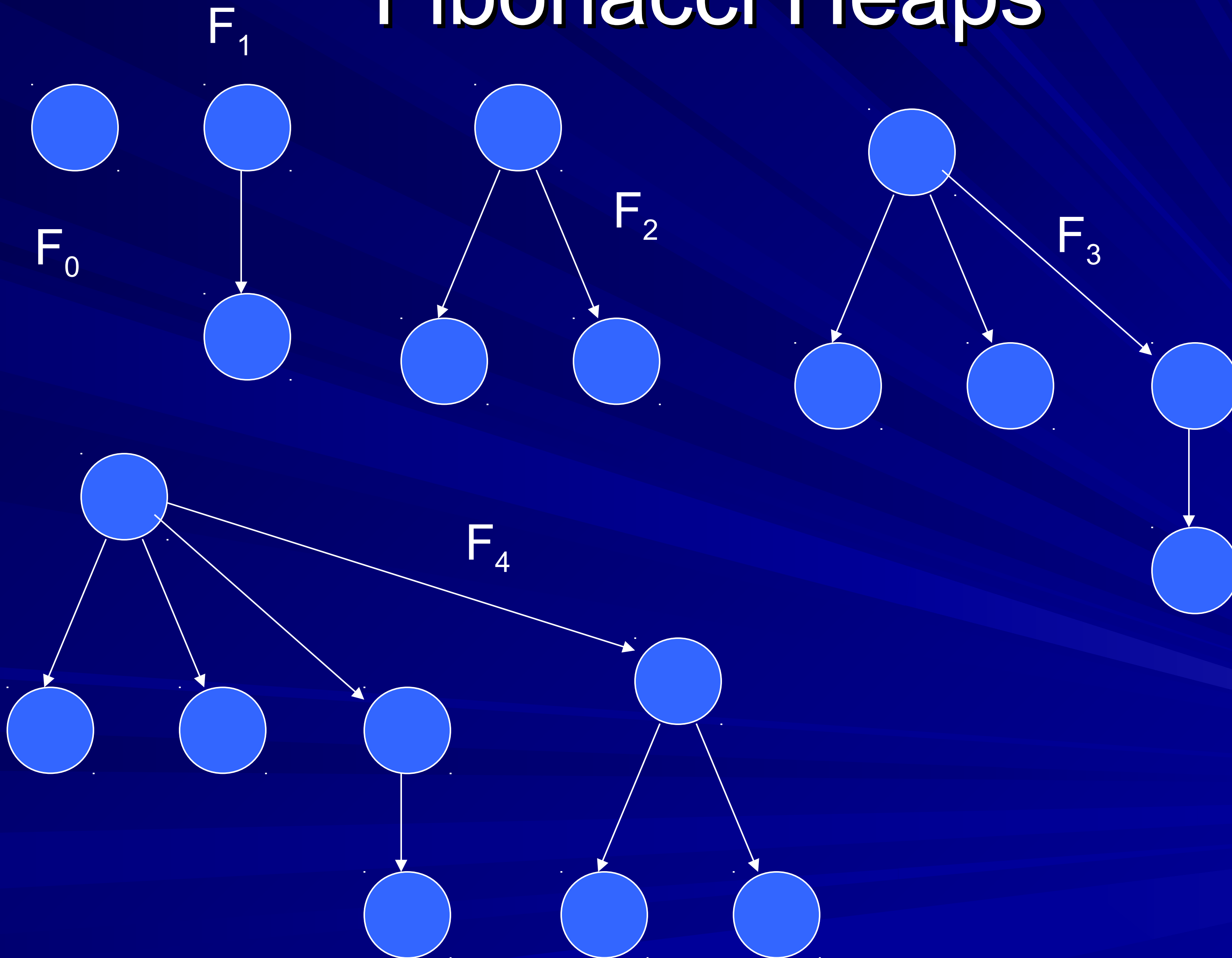
Insert

- Worst-Case O(lg n)
- Amortized cost per insert O(1).
  - 3
  - 1 credit to create a B_0
  - Place 2 credits on B_0
  - When we merge B_i and B_i, each tree has 2 credits.
  - Find the min root (1 comparison)
  - Make the min root point to the other (1)
  - We used 2 credits to make a B_{i+1}
  - B_{i+1} has 2 credits

Running Times

Proc	Binary Heap (worst-case)	Binomial (worst-case)	Binomial (amortized)	Fibonacci (amortized)
MakeHeap	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$
Insert	$\Theta(\lg n)$	$O(\lg n)$	$\Theta(1)$	$\Theta(1)$
Minimum	$\Theta(\lg n)$	$O(\lg n)$	$\Theta(1)$	$\Theta(1)$
Extract-Min	$\Theta(\lg n)$	$O(\lg n)$	$O(\lg n)$	$O(\lg n)$
Union (Merge)	$\Theta(n)$	$O(\lg n)$	$\Theta(1)$ lazy $O(\lg n)$ eager	$\Theta(1)$
Decrease Key	$\Theta(\lg n)$	$\Theta(\lg n)$	$\Theta(1)$	$\Theta(1)$
Delete	$\Theta(\lg n)$	$\Theta(\lg n)$	$O(\lg n)$	$O(\lg n)$

Fibonacci Heaps



Fibonacci Heaps

- Rank of the root of F_i is i
- The size of F_k>= ϕ^k, i.e. exponential
  - ϕ=(1+sqrt(5))/2
- Used for Dijkstra’s shortest path and Prims MST
- Low amortized cost for Decrease-Key is why Fibonacci heaps are used for to solve many graph problems

Other Advanced Data Structures

- Binary Search Trees
  - Balanced trees (B-tree)
  - Splay trees, dynamic trees, persistent trees
    - Balance happens automatically
    - Invented by Sleator and Tarjan
  - We will discuss Search Trees later.