2. (7, I); (8, G)

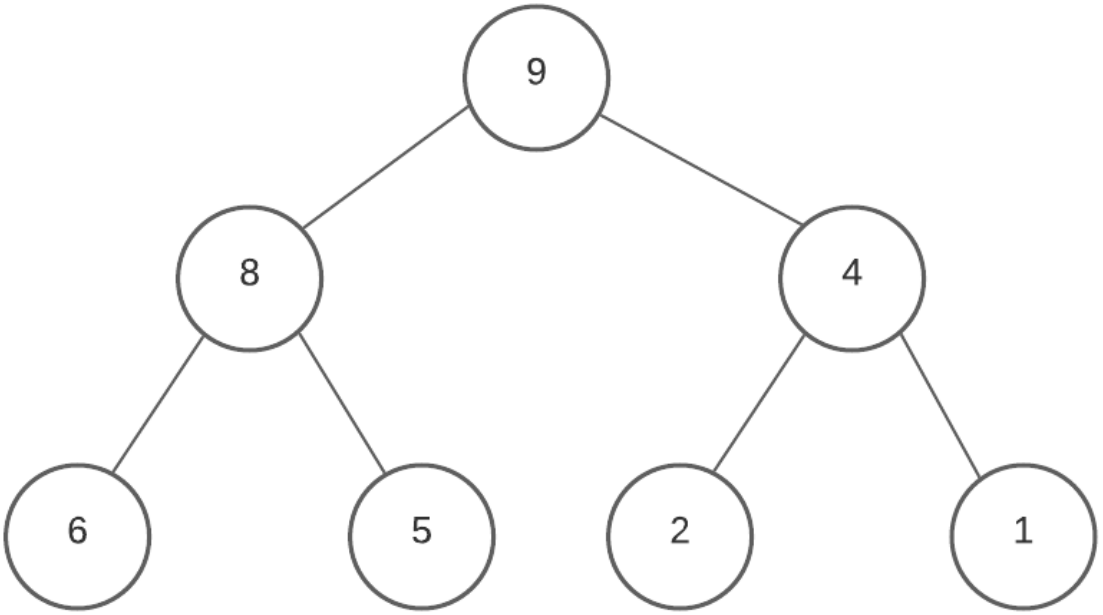
3. Deve-se usar a fila de prioridades por ser um simulador. Portanto o voo será retirado com o menor time-stamp e assim consecutivamente.

4. Pode estar armazenada na raiz da árvore.

~~5. Sim, pois para uma árvore ser um heap, ela deve ser uma árvore binária completa. Então a árvore T deve ser considerada um heap por ser completa.~~

~~6. Não se considera pois, sem criar o filho da esquerda antes, não será possível ter o filho da direita.~~

~~7.~~



Prefixado: 9, 8, 6, 5, 4, 2, 1

Interfixado: 6, 8, 5, 9, 2, 4, 1

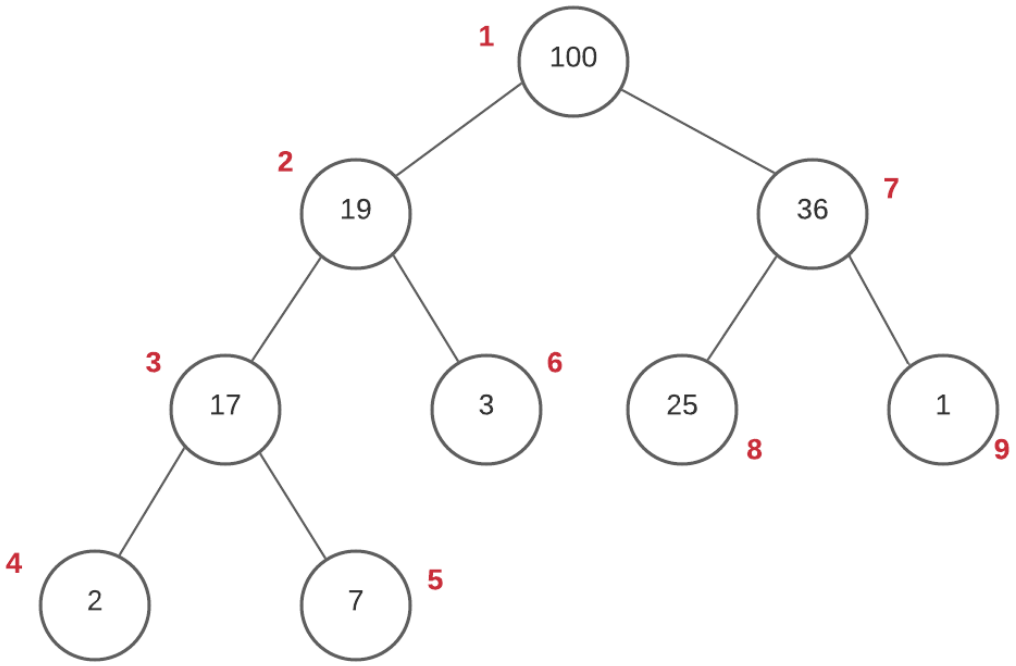
Pós-fixado: 6, 5, 8, 2, 4, 1, 9

~~8. Prefixado: 1, 2, 4, 8, 9, 5, 10, 11, 3, 6, 12, 13, 7, 14, 15~~

~~Interfixado: 8, 4, 9, 2, 10, 5, 11, 1, 12, 6, 13, 3, 14, 7, 15~~

~~Pós-fixado: 8, 9, 4, 10, 11, 5, 2, 12, 13, 6, 14, 15, 7, 3, 1~~

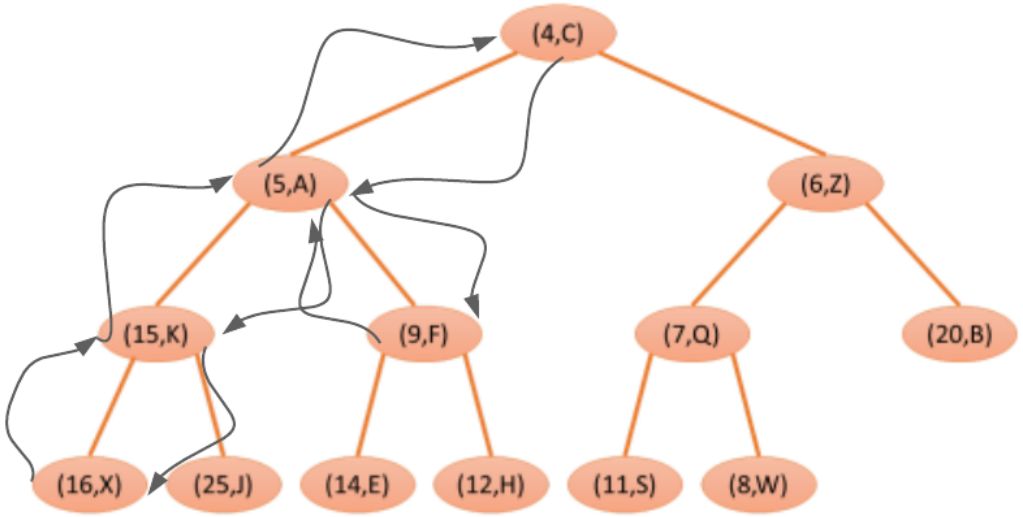
9.



100 -> 19 -> 17-> 2-> 7 -> 3 -> 36 -> 25 -> 1

10. Um exemplo de heap que prova que ela está errada é o CompleteBinaryTree. O heap que foi fornecido para provar como errado é a imagem do heap que é representado pela lista de [x, 1,5,2,8,9,1,6]. esta pilha não irá produzir chaves em ordem crescente quando um caminhamento pós fixado é usado.

11.



12.

13.

14. O hashing duplo consegue tolerar um fator de carga superior a 1 e o que não consegue tolerar o fator de carga é o endereçamento aberto.

15. Se usarmos a soma dos componentes, a função hash seria x0 + x1 + x2 + x3 + x4 mod N.

16.
11-entry hash table: h(i) = (3i+5) mod 11

empty hash table:

[], [], [], [], [], [], [], [], [], [], []

hash key 12:

[], [], [], [], [], [], [], [], [12], [], []

hash key 44

[], [], [], [], [], [44], [], [], [12], [], []

hash key 13

[13], [], [], [], [], [44], [], [], [12], [], []

hash key 88

[13], [], [], [], [], [[44, 88]], [], [], [12], [], []

hash key 23

[13], [], [], [], [], [[44, 88]], [], [], [12, 23], [], []

hash key 94

[13], [94], [], [], [], [[44, 88]], [], [], [[12, 23]], [], []

hash key 11

[13], [94], [], [], [], [[44, 88, 11]], [], [], [[12, 23]], [], []

hash key 39

[13], [[94, 39]], [], [], [], [[44, 88, 11]], [], [], [[12, 23]], [], []

hash key 20

[13], [[94, 39]], [], [], [], [[44, 88, 11]], [], [], [[12, 23]], [], [20]

hash key 16

[13], [[94, 39]], [], [], [], [[44, 88, 11]], [], [], [[12, 23]], [16], [20]

hash key 5

[13], [[94, 39]], [], [], [], [[44, 88, 11]], [], [], [[12, 23]], [[16, 5]], [20]

17.
11-entry hash table: h(i) = (3i+5) mod 11

empty hash table:

[], [], [], [], [], [], [], [], [], []

hash key 12:

[], [], [], [], [], [], [], [], [12], [], []

hash key 44

[], [], [], [], [], [44], [], [], [12], [], []

hash key 13

[13], [], [], [], [], [44], [], [], [12], [], []

hash key 88

[13], [], [], [], [], [44], [88], [], [12], [], []

hash key 23

[13], [], [], [], [], [44], [88], [], [12], [23], []

hash key 94

[13], [94], [], [], [], [44], [88], [], [12], [23], []

hash key 11

[13], [94], [], [], [], [44], [88], [11], [12], [23], []

hash key 39

[13], [94], [39], [], [], [44], [88], [11], [12], [23], []

hash key 20

[13], [94], [39], [], [], [44], [88], [11], [12], [23], [20]

hash key 16

[13], [94], [39], [16], [], [44], [88], [11], [12], [23], [20]

hash key 5

[13], [94], [39], [16], [5], [44], [88], [11], [12], [23], [20]

18.
11-entry hash table: h(i) = (3i+5) mod 11

Quadratic Probing: h(i) = (3i+5) + j^2 mod 11 for j = 0,1,2...

empty hash table:

[], [], [], [], [], [], [], [], [], [], []

hash key 12 (j = 0)

[], [], [], [], [], [], [], [], [12], [], []

hash key 44 (j = 0)

[], [], [], [], [], [44], [], [], [12], [], []

hash key 13 (j = 0)

[13], [], [], [], [], [44], [], [], [12], [], []

hash key 88 (j = 1)

[13], [], [], [], [], [44], [88], [], [12], [], []

hash key 23 (j = 1)

[13], [], [], [], [], [44], [88], [], [12], [23], []

hash key 94 (j = 0)

[13], [94], [], [], [], [44], [88], [], [12], [23], []

hash key 11 (j = 3)

[13], [94], [], [11], [], [44], [88], [], [12], [23], []

hash key 39 (j = 1)

[13], [94], [39], [11], [], [44], [88], [], [12], [23], []

hash key 20 (j = 0)

[13], [94], [39], [11], [], [44], [88], [], [12], [23], [20]

hash key 16 (j = 3)

[13], [94], [39], [11], [], [44], [88], [16], [12], [23], [20]

hash key 5 (j = ?)

all attempts for j = 1 to 10, mod 11 does not equal the empty bucket number 4.

19.
11 entry hash table: h(i) = [(3i+5) + j x [7 - (k mod 7)]] mod 11, for j = 0,1,2...

empty hash table:

[], [], [], [], [], [], [], [], [], [], []

hash key 12:

[], [], [], [], [], [], [], [], [12], [], []

hash key 44

[], [], [], [], [], [44], [], [], [12], [], []

hash key 13

[13], [], [], [], [], [44], [], [], [12], [], []

hash key 88

h(i) = [(3i+5) + (j x [7 - (k mod 7)])] mod 11, for j = 3
h(88) = [(3(88)+5) + (3 x [7 - (88 mod 7)])] mod 11
h(88) = [269 + (3 x [7 - (88 mod 7)])] mod 11
h(88) = [269 + (3 x [7 - 4])] mod 11
h(88) = [269 + 9] mod 11
h(88) = 278 mod 11 = 3
[13], [], [], [88], [], [44], [], [], [12], [], []

hash key 23

h(i) = [(3i+5) + (j x [7 - (k mod 7)])] mod 11, for j = 1
h(23) = [(3(23) + 5) + (1 x [7 - (23 mod 7)])] mod 11
h(23) = [74 + (1 x [7 - (23 mod 7)])] mod 11
h(23) = [74 + (1 x [7 - 2])] mod 11
h(23) = [74 + 5)] mod 11
h(23) = 79 mod 11 = 2
[13], [], [23], [88], [], [44], [], [], [12], [], []

hash key 94

[13], [94], [23], [88], [], [44], [], [], [12], [], []

hash key 11

h(i) = [(3i+5) + (j x [7 - (k mod 7)])] mod 11, for j = 4
h(11) = [(3(11) + 5) + (4 x [7 - (11 mod 7)])] mod 11
h(11) = [38 + (4 x [7 - (11 mod 7)])] mod 11
h(11) = [38 + (4 x [7 - 4])] mod 11
h(11) = [38 + (4 x 3)] mod 11
h(11) = 50 mod 11 = 6
[13], [94], [23], [88], [], [44], [11], [], [12], [], []

hash key 39

h(i) = [(3i+5) + (j x [7 - (k mod 7)])] mod 11, for j = 1
h(39) = [(3(39) + 5) + (1 x [7 - (39 mod 7)])] mod 11
h(39) = [122 + (1 x [7 - (39 mod 7)])] mod 11
h(39) = [122 + (1 x [7 - 4])] mod 11
h(39) = [122 + (1 x 3)] mod 11
h(39) = 125 mod 11 = 4
[13], [94], [23], [88], [39], [44], [11], [], [12], [], []

hash key 20

[13], [94], [23], [88], [39], [44], [11], [], [12], [], [20]

hash key 16

[13], [94], [23], [88], [39], [44], [11], [], [12], [16], [20]

hash key 5

h(i) = [(3i+5) + (j x [7 - (k mod 7)])] mod 11, for j = 10
h(i) = [(3(5) + 5) + (10 x [7 - (5 mod 7)])] mod 11

h(i) = [20 + (10 x [7 - (5 mod 7)])] mod 11
h(i) = [20 + (10 x [7 - 5])] mod 11
h(i) = [20 + (10 x 2)] mod 11
h(i) = 40 mod 11 = 7
[13], [94], [23], [88], [39], [44], [11], [5], [12], [16], [20]

20.

21. A principal diferença entre os dois é que o TAD mapa deve ter chaves únicas, já o dicionário pode ter múltiplas entradas com a mesma chave.

22.

23.