Huffman Tree – encode – decode

# Text which should be compressed:

**hi Hannah**

**ACHTUNG: Ich schreibe jetzt ein Leerzeichen als Underline** ‘\_’

1. **Collect all letters from a text**

Allocation of 5 h\_chars  
['\_', 'a', 'h', 'i', 'n']

1 2 3 1 2

1. **Queue nodes and create a tree**

* Insertion Sort of nodes  
    
  ['i', '\_', 'n', 'a', 'h']  
   1 1 2 2 3

Dequeue 2 Elemente + neuer Knoten + Verbindungen   
 --'i\_'--

| |

'i' '\_'

* Enqueue('i\_')  
    
  ['i\_', 'n', 'a', 'h']

2 2 2 3   
Dequeue 2 Elemente + neuer Knoten + Verbindungen   
 --'i\_n'--

| |  
 --'i\_'-- 'n'

| |

'i' '\_'

* Enqueue('i\_n')  
    
  ['a', 'h', 'i\_n']

2 3 4

* Dequeue 2 Elemente + neuer Knoten + Verbindungen  
    
   --'i\_n'-- --'ah--

| | | |   
 --'i\_'-- 'n' 'a' 'h'

| |

'i' '\_'

* Enqueue('ah')  
  ['i\_n', 'ah']

4 5

* Dequeue 2 Elemente + neuer Knoten + Verbindungen  
     
   ------'i\_nah'-------   
   0| 1|   
   --'i\_n'-- --'ah'--

0| 1| 0| 1|   
 --'i\_'-- 'n' 'a' 'h'

0| 1|

'i' '\_'

10 a

11 h

01 n

000 I

001 \_

==> 11101101 (hahn)

==> 111001011011 (hannah)