Exercise 2: Huffman codes

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table 2** | α | β | γ | δ | **∈** | ζ | η | θ |
| Frequency | 10 | 400 | 80 | 50 | 20 | 300 | 100 | 40 |
| Fixed-length | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| Variable-length | 100100 | 0 | 1010 | 1000 | 100101 | 11 | 1011 | 10011 |

110100 =

1. How many bits do we need to store the file if using a fixed-length code?
   * + **3 bits/character\*1000 characters so we would need 3000 bits to store the file using the fixed-length code**
2. Show the code built by the Huffman algorithm, both as a tree and as a list (character,codeword)

* Refer to **Table 2** to see the list

**1000**

/0 \1

**β:**400 **600**

/0 \1

**300** **ζ**:300

/0 \1

**120** **180**

/0 \1 /0 \1

**δ**:50 **70** **γ**:80 **η**:100

/0 \1

**30** **θ**:40

/0 \1

**α**:10 **∈**:20

1. (10\*6+400\*1+80\*4+50\*4+20\*6+300\*2+100\*4+40\*5) **= 2300 bits**
2. Show the code built by the Huffman algorithm, both as a tree and as a list (charac-ter,codeword)

**1000**

/0 \1

**500** **500**

/0 \1 /0 \1

**250 250 250 250**

/0 \1 /0 \1 /0 \1 /0 \1

**γ:125 δ:125 α:125 β:125 η:125 θ:125 ∈:125 ζ:125**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table 3** | α | β | γ | δ | **∈** | ζ | η | θ |
| Frequency | 125 | 125 | 125 | 125 | 125 | 125 | 125 | 125 |
| Variable-length | 010 | 011 | 000 | 001 | 110 | 111 | 100 | 101 |

1. 8(125\*3) = **3,000 bits**
2. We can assume that the worst case for this algorithm is when the frequency is the same for all of the characters because it creates a balance tree. Total bits require = size of the tree \* no of characters