1. Here is a Huffman tree for the given frequencies:

```
100% 63% 29% 16%
\_ \_ \_ \_
37% 34% 13% 1
\_ \_ \_ \_
18% 16% 7% 3%
\_ \_ \_
19% 10% 4%
\_ \_
16% 49%
\_ \_
18% 10%
\_ \_
18% 10%
\_ \_
18% 10%
\_ \_
18% 10%
```

The corresponding code is:

- ♠: 000
- ♦: 001
- ♣: 010
- ♠: 011
- ♣: 1000
- ♦: 1001
- ♠: 1010
- ♣: 1011
- ♦: 1111
- ♠: 1110
- ♣: 1111

\[ E[\text{no. of bits persymbol}] = 3 \times 0.27 + 4 \times 0.66 + 6 \times 0.07 = 3.87 \]

Huffman code is more efficient than fixed length code by

\[ \frac{4 - 3.87}{4} \times 100 = 3.25\% \]
Base case: 1 node

- no. of full nodes = 0
- no. of leaves = 1

Inductive step:

The recursive description of a binary tree is:

\[ T \]

where \( T_2 \) does not have to exist

Inductive hypothesis -

Assume statement is true in \( T_1 \) and \( T_2 \) (if it exists), i.e. \( FN_i = L_i - 1 \) for \( i = 1 \) and \( i = 2 \) if \( T_2 \) exists,

where \( FN_i \) = no. of full nodes in \( T_i \) and
\( L_i \) = no. of leaves in \( T_i \)

Case 1 - \( r \) has 1 child (\( T_2 \) does not exist)

Let \( FN = \) no. of full nodes in \( T \)
\( L = \) no. of leaves in \( T \)

Since \( r \) is not a full node in this case,
\( FN = FN_1 \)
\( L = L_1 \)

Since \( FN_1 = L_1 - 1 \), \( FN = L - 1 \)
Case 2 - r has 2 children (T2 does exist)

in this case, r is a full node but is still not a leaf

\[ FN = FN_1 + FN_2 + 1 \quad \text{and} \]
\[ L = L_1 + L_2 \]

using inductive hypothesis,

\[ FN = (L_1 - 1) + (L_2 - 1) + 1 \]
\[ = L_1 + L_2 - 1 = L - 1 \quad \text{QED} \]

\[ \begin{array}{c}
\text{Before delete} \\
\begin{array}{c}
\circ \\
\circ \\
\circ
\end{array}
\end{array} \quad \begin{array}{c}
\text{After delete} \\
\begin{array}{c}
\circ \\
\circ
\end{array}
\end{array} \]

After delete, l becomes the left subtree of m

since m > x and everything in l is ≤ x, this maintains the BST property

also, everything in r stays on the same side of p as before the delete, so that maintains the BST property
if the height of $L$ is $h_1$ and the height of $R$ is $h_2$, then paths below $p$ have length at most $\max(h_1, h_2)$ before the delete.

After the delete, some paths below $p$ can have length $h_1 + h_2$ for the strategy in this problem (Sec. 5.7 path lengths would not change after delete).

If the tree was reasonably well balanced before the delete so that $h_1$ and $h_2$ are similar, then $h_1 + h_2 \gg \max(h_1, h_2)$.

We prefer the solution with shorter path lengths.

\[ \therefore \text{we prefer the strategy in Sec. 5.7} \]