```
char letters[] = { 'A', 'B','C', 'D', 'E', 'F' };

double freq[] = { 0.1, 0.3, 0.1, 0.2, 0.15, 0.15 };
```

```
int size = 6;
Node<char>* elements[6];
for (int i = 0; i < size; ++i) {
        elements[i] = new Node<char>;
        elements[i]->value = letters[i];
        elements[i]->frequency = freq[i];
        elements[i]->left = nullptr;
        elements[i]->right = nullptr;
}
```

```
int index1;
int index2;
double freq1;
double freq2;

while (size > 1) {
        index1 = index2 = 0;
        freq1 = freq2 = numeric_limits<double>::max();
        for (int i = 0; i < size; ++i)  {
                if (elements[i]->frequency < freq1)  {
                        freq2 = freq1;
                        index2 = index1;
                        freq1 = elements[i]->frequency;
                        index1 = i;
                }
                else if ((elements[i]->frequency < freq2)
                                && (i != index1)) {
                        freq2 = elements[i]->frequency;
                        index2 = i;
                }
```

```
        }
        Node<T>* newNode;
        newNode = new Node<char>;
        newNode->left = elements[index1];
        newNode->right = elements[index2];
        newNode->frequency = freq1 + freq2;

        elements[min(index1,index2)] = newNode;
        for (int i = max(index1,index2); i < size - 1; ++i) {
                elements[i] = elements[i + 1];
        }

        --size;
}
```

```
void printArray(int path[], int pathLen, Node<char>* leaf)
{
	std::cout << leaf>value << "     ";
	for (int i = 0; i < pathLen; i++)  {
		std::cout << path[i] << " ";
	}
	std::cout << std::endl;
}
```