1) besuche

```
void besuche(*NODE_PTR node)
{
    if (tree != NULL)
    {
        besuche (node->left);
        insert_element_at_back (list, node->data);
        besuche (node->right);
    }
```

2) ELEM* (HEAD* head, void* data)

insertFront

```
{
    ELEM* temp = (ELEM*) malloc (sizeof (ELEM*));
    if (temp)
    {
        temp->data = data;
        temp->next = head->first;
        head->first = temp;
        if (head->len == 0)
        {
            head->last = temp;
        }
        head->len++;
        return temp;
    }
} return NULL;
```

```
void* removeLast(HEAD* list)
{
    void* temp = NULL;
    ELEM* tempList = NULL;

    if (list)
    {
        tempList = list -> last
        temp = tempList -> data;

        list -> last = tempList -> prev;
        free(list -> last);
    }
    return temp;
}

ELEM* exFirst(HEAD* list)
{
    if(list -> len > 0)
    {
        return (list -> first);
    }
    return NULL;
}

int queueLen(HEAD* list)
{
    if (list)
    {
        return (list -> len);
    }
    return -1;
```