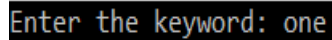


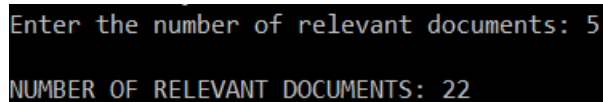
CSE2025 Data Structures
Project #2
Merve Rana Kızıl - 150119285

- 1) The keyword is taken from the user as can be seen in the screenshot given below.



```
Enter the keyword: one
```

- 2) Number of relevant documents is taken from the user. Total number of relevant documents is found. The screenshot is given below.



```
Enter the number of relevant documents: 5
NUMBER OF RELEVANT DOCUMENTS: 22
```

- 3) Enqueue implementation is given below. This function inserts the given node into the binomial heap. It creates a one-node binomial heap and unites it with the given binomial heap.

```
// Inserts the given node into the binomial heap
Node* insertIntoBinomialHeap(Node* BH, Node* node) {
    Node* BH1 = newBinomialHeap();
    node->parent = NULL;
    node->child = NULL;
    node->sibling = NULL;
    node->degree = 0;
    BH1 = node;
    // Free the temporary binary heap BH1
    BH = binomialHeapUnion(BH, BH1);
    return BH;
}
```

Deque implementation is given below. This function finds the node that holds the maximum key in the root list of the given binomial heap and removes that node from the root list. After that, it reverses the order of the linked list of the removed node's children. Then, it unites the given binomial tree, which one of its root nodes is removed, and the binomial tree that is obtained after reversing the linked list of the removed node's children.

```
// Extracts the node with maximum key
Node* extractMax(Node* BH){

    if (BH == NULL)
        return NULL;
```

```

Node *prevMaxRoot = NULL;
Node *maxRoot = BH;

// Find the node with maximum key
int max = maxRoot->number_of_words;
Node *current = BH;

while (current->sibling != NULL) {
    if ((current->sibling)->number_of_words > max) {
        max = (current->sibling)->number_of_words;
        prevMaxRoot = current;
        maxRoot = current->sibling;
    }
    current = current->sibling;
}

if(maxRoot->number_of_words > 0){

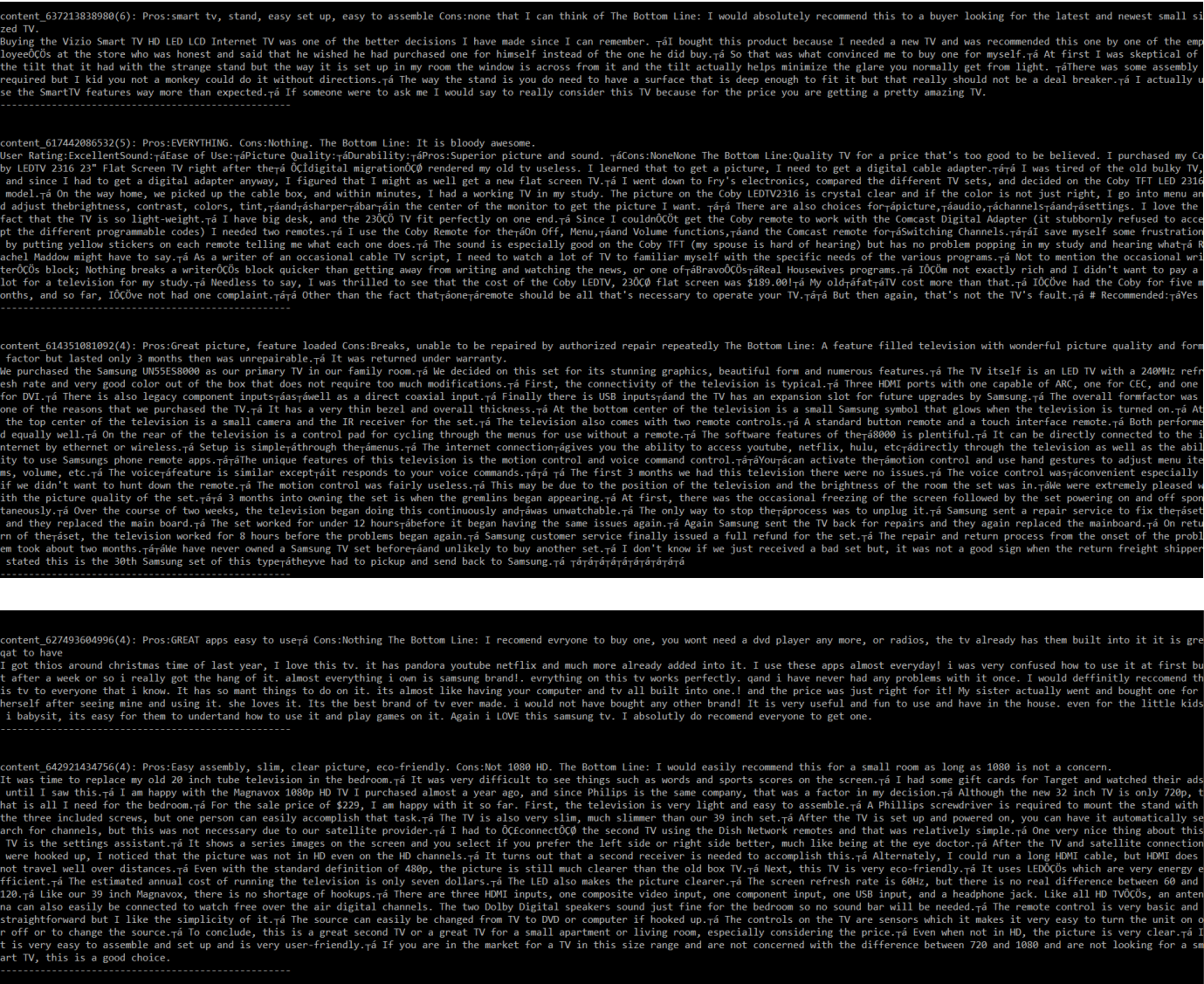
    printf("\n%s(%d): ", maxRoot->filename, maxRoot->number_of_words);
    readWords(maxRoot->filename, "", 1);
    printf("-----\n\n");
}

// If there is only one node in the binomial heap
if (prevMaxRoot == NULL && maxRoot->sibling == NULL)
    BH = NULL;
else if (prevMaxRoot == NULL)
    BH = maxRoot->sibling;
else
    // Remove the node with maximum key from root list
    prevMaxRoot->sibling = maxRoot->sibling;

Node* BH1 = newBinomialHeap();
// Reverse linked list of maxRoot's children
BH1 = reverseList(maxRoot->child);
BH = binomialHeapUnion(BH, BH1);
return BH;
}

```

4) A binomial heap is built using the keyword “one”. The names of the documents, the frequencies of the keywords and the contents of the documents are extracted and printed for the most relevant 5 documents. The screenshot of the output is given below.



5) Root of a maximum binomial tree keeps the maximum value. Therefore, a max-binomial-heap is constructed to retrieve the most relevant documents since it is easier to extract the maximum frequencies from the max-binomial-heap. Using a max-priority-queue is an efficient way to solve this problem.