

**Project #2**

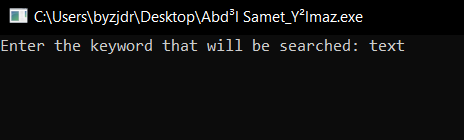
NAME ID

Abdül Samet Yılmaz 150118059

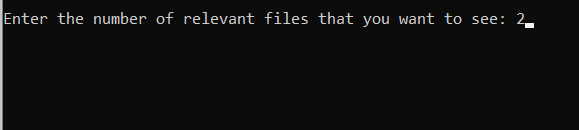
CSE2025 Data Structures, Fall 2020

Date Submitted: January 24, 2021

**1)**

****

**2)**

****

**3)**

**void enqueue(BH\_Header \*hdr, int key, char fN[40]) {**

**BH\_Node \*node = B0\_Create(key, fN);**

**BH\_Header \*temp = BH\_Create();**

**temp->head = node;**

**hdr->head = BH\_Merge(hdr, temp);**

**free(temp);**

**}**

**void dequeue(BH\_Header \*heap, BH\_Node \*head, BH\_Node \* prev) {**

**if( head == heap->head )**

**heap->head = head->sibling;**

**else**

**prev->sibling = head->sibling;**

**BH\_Node \*newHead = NULL;**

**BH\_Node \*chld = head->child;**

**while (chld != NULL)**

**{**

**BH\_Node \*next = chld->sibling;**

**chld->sibling = newHead;**

**chld->parent = NULL;**

**newHead = chld;**

**chld = next;**

**}**

**BH\_Header \*temp = BH\_Create();**

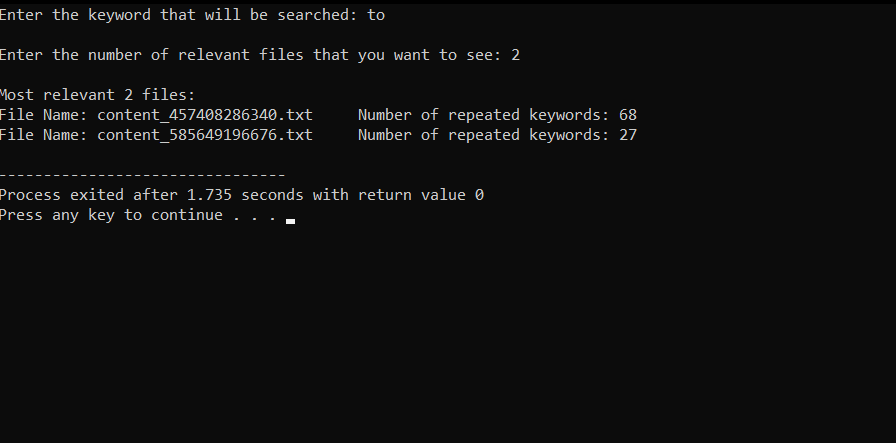
**temp->head = newHead;**

**heap->head = BH\_Merge(heap, temp);**

**free( temp );**

**}**

**4)**

****

**5)**

Using priority queues for this kind of problems is very effective, because it makes much more easier to find relevant files since we give files keys which is kind of relevancy points.  
In conclusion priority queues are very useful for this type of works.