Problem 1

a) Insert Algorithm:

```
function SEARCH_DOUBLY_LINKED_LIST($L$, $x$)
    $e \leftarrow L$.head
    while $e$.key $\neq x$ and $e \neq$ NIL do
        $e \leftarrow e$.next
    end while
    return e
end function
```

Search Algorithm:

```
function INSERT_DOUBLY_LINKED_LIST($L$, $x$)
    $e \leftarrow$ new element
    $e$.key $\leftarrow x$
    $e$.next $\leftarrow L$.head
    $L$.head.prev $\leftarrow e$
    $L$.head $\leftarrow e$
    return $L$
end function
```

b)

```
function DOUBLY_LINKED_LIST_APPEND($A$, $B$)
    $a_i \leftarrow A$.head
    while $a_i$.next $\neq$ NIL do
        $a_i \leftarrow a_i$.next
    end while
    ▷ To keep list B intact, it seems to be not possible to simply link the head of B to the end of A
    $b_i \leftarrow B$.head
    while $b_i$.next $\neq$ NIL do
        $e \leftarrow$ new element
        $e$.key $\leftarrow b_i$.key
        $a_i$.next $\leftarrow e$
        $e$.prev $\leftarrow a_i$
        $a_i \leftarrow a_i$.next
        $b_i \leftarrow b_i$.next
    end while
    $e \leftarrow$ new element
    $e$.key $\leftarrow b_i$.key
    $a_i$.next $\leftarrow e$
    $e$.prev $\leftarrow a_i$
    return $A$
end function
```

The current algorithm has a asymptotic running time in $O(m+n)$, with the number of element of A being m and B being n.

Interestingly, if we don't care if B is intact or not, and meanwhile we

- point list L.head.prev to be the last element of L and in accordance
- point list L.head.prev.next to L.head,

the algorithm could be much more simpler:

```
function DOUBLY_LINKED_LIST_APPEND($A$, $B$)
    $a \leftarrow A$.head.prev
    $a$.next $\leftarrow B$.head                          ▷ A's tail to B's head
    $B$.head.prev.next $\leftarrow A$.head                ▷ B's tail to A's head
    $A$.head.prev $\leftarrow B$.head.prev                ▷ A's head to B's tail
    $B$.head.prev $\leftarrow a$                          ▷ B's head to A's tail
    return $A$
end function
```

This algorithm has an asymptotic running time in $O(1)$.

c)

```
function DOUBLY_LINKED_LIST_ZIP($A$, $B$)
    $i \leftarrow 1$
    $a_i \leftarrow A$.head
    $b_i \leftarrow B$.head
    while $i < n$ do
        ▷ Repointing $a_i$.next, $b_i$.prev, $b_i$.next and $a_{i+1}$.prev
        $a_{i+1} \leftarrow a_i$.next
        $a_i$.next $\leftarrow b_i$
        $b_i$.prev $\leftarrow a_i$
        $b_{i+1} \leftarrow b_i$.next
        $b_i$.next $\leftarrow a_{i+1}$
        $a_{i+1}$.prev $\leftarrow b_i$
        ▷ Initializing the variables for the next iteration
        $a_i \leftarrow a_{i+1}$
        $b_i \leftarrow b_{i+1}$
        $i \leftarrow i + 1$
    end while
    $a_i$.next $\leftarrow b_i$
    $b_i$.prev $\leftarrow a_i$
    return $A$
end function
```