Made Naradeva Handika Prameswara / 10303230010 1

Soal 1

a.
```
type akun <
        username  : string
        password  : string
>
type infotype : akun
type address  : pointer to elmList
type elmList <
        info : infotype
        next : address
        prev : address
>
type List <
        first : address
        last  : address
>
procedure createList (in/out [redacted] l : List)
[redacted]
Function newElm ( akun : infotype ) -> [redacted] address
Procedure insertLast (in/out l : List , in p : address )
Function findAkun (l : List , username : string) -> address
Procedure signUp (in/out l : List , in akun : infotype)
Procedure deleteFirst (in/out L : List , out p : address)
Procedure deleteAfter (in q : address , out p : address)
Procedure deleteLast (in/out l : List , out p : address)
Procedure removeAkun (in username : string , in/out l : List)
```

b. (i)
```
Function newElm (akun : infotype) -> address
  kamus
  p : address
  Algoritma
    allocate(p)
    p -> next = NIL
    p -> prev = NIL
    p -> info = akun
    return p
endfunction
```

(ii) Procedure insertLast (in/out L : List, inp : address)

Kamus

Algoritma

```
if L.first == NIL AND L.last == NIL then
    L.first = p
    L.last = p
else
    L.last -> next = p
    p -> prev = L.last
    L.last = p
endif
```

Endprogram

(iii) Function findAkun (L : List, username : string) -> address

Kamus

```
p : address
```

Algoritma

```
if L.first == NIL then
    return NIL
else
    p = L.first
    while p != NIL do
        if p -> info.username == username then
            return p
        endif
        p = p -> next
    endwhile
    return NIL
endif
```

endfunction

Made Naradeva Hendika Pramesta / 1302222300101

(iv) Procedure signUp(in/out l:List, in akun : infotype)

Kamus

p : address

newElm (akun : infotype) → address

insertLast (in/out l:List, in p:address)

findAkun (l:List, username :String) → address

Algoritma

p = findAkun (l, akun.username)

if p ≠ NIL then

output("Account has been registered")

else

p = newElm (akun)

insertLast (l, p)

endif

endprogram

C.(i) Procedure deleteFirst (in/out l : List, out p : address)

Kamus

Algoritma

p = l.first

l.first = p→next

p→next = Nil

l.first→prev = NIL

endprocedure

(ii) Procedure deleteAfter (in q : address, out p : address)

Kamus

Algoritma

p = q→next

q→next = p→next

p→next→prev = q

p→next = NIL

p→prev = NIL

endprocedure

(iii) Procedure deleteLast (in/out l : List, out p: address)

Kamus

Algoritma

```
p = l.last
l.last = p->prev
l.last->next = NIL
p->prev = NIL
```

endprocedure

(iv) Procedure removeAkun (in username : string, in/out l : List)

Kamus

```
p : address
findAkun (l : last, username : String) -> address
deleteFirst (in/out l : List, out p : address)
deleteAfter (in q : address, out p : address)
deleteLast (in/out l : List, out p : address)
```

Algoritma

```
if l.first == NIL then
    output ("List is empty")
else
    p = findAkun (l, username)
    if p == NIL then
        output ("Account has not found")
    else
        if p == l.first then
            deleteFirst (l, p)
        else if p == l.last then
            deleteLast (l, p)
        else
            deleteAfter (p->prev, p)
        endif
        deallocate (p)
    endif
endif
```

endprocedure