DW Sheet N° 09: Linked list Solution

Exercise 1:

```
Type list =record
    Word : c.c;
    Next : *list
End list
Var head : *list;
Begin
head←NULL;
2. function nbr(var l:*list ):integer ;
Var c: integer; p: *list ;
Begin
p←l;
c←0;
while (p.next<>NULL) do
c←c+1;
p←p.next;
end while
return c;
end function
```

Exercise 02 :

```
 procedure concat(var l1,l2:*list )
var p,q:*list ;
begin
if  l1= NULL then
l1←l2;
else
p←l1;
while (p.next <>NULL) do
p←p.next;
end while
p.next←l2;
end procedure
procedure concat (var l1,l2:liste)
begin
if  l1= NULL then
l1←l2;
else
p←l2;
while(p!=null) do
inserttail(l1,p.data);
p←p.next;
endif
end.
```

Exercise 03:

```
procedure inserthead(var head:*list, int v)
var  p,q: *list ;
Begin
Allocate (q);
q.data←v;
If head = NULL then
Head ←q;
q.next←NULL
else
q.next ←head;
head←q;
end if
end
procedure reverse(l1:*list,l2:*list)
var p:*list ;
Begin
p←l1;
while (p <>NULL) do
inserthead(l2,p.data);
p←p.next;
end while
l1←l2;
end procedure.
Ou
Procedure reverselist(var l:*list)
Begin
If(l.next != null) then
reverselis(l->next) ;
Write(‘’ <-‘’,l.data);
Else
Write(“null”);
Endif
End.
```

Exercise 04:

```
Procedure delet(int v ):
Var p,q,head :*list;
Begin
p←head;
while(p.data<>v )and (p.next <>null) do
q←p;
p←p.next
end while
if (p.next= null) then
write(“ the is not this value “);
else
q.next←p.next;
free(p);
end
```

Exercise 05:

```
procedure inserpos(int v,m);
insertail(int v);
procedure insersorted(int m);
var q,p:*list;
Begin
```

```
allocate(q);
q.data←m;
p←head;
while(p.next!=null) and (p.data<m) and
(p.next.data>m) do
p←p.next
end while
if p.next =null then insertail(m);
else
insertpos(p.data,m);
end if
end.
```

Exercise 06 : concatenation between two linked list

```
Type Liste = record
   Val : integer ;
   next : * Liste ;
End
Procedure concate(var L1, L2 :*liste);
Var p1,p2 : *Liste;
Begin
p1←L1;
p2←L2;
while ((p1<>NULL ) and (p2<>NULL) )do
if (p1->val >p2->val) then
inserposavant (L1,p2->val,p1->val);
p2←p2->next;
else
p1←p1->next;
end if
end while
end
inserposavant (var L: *Liste , x:
integer,v :integer);
var *r,*p,*q: liste;
    Begin
    if (head = NULL) then
    return;
    else
    p=L
    while(p <>NULL) do
    if(p->val <>v) then
    r←p;
    p←p->next;
    else ;
    Allocate (q);
    q.val←x;
    r->next=q;
    q->next=p;
    endif
    endwhile
end.
```

Exercise 07:

Procedure delete(var L:*liste, x: integer);

```
Var r, p,q: *Liste;
Begin
p←L ;
q←p ;
if(L=NULL )then return ;
else
while (p<>NULL) do
if(p->val<> x) then
q←p;
p←p->next;
else
r←p;
q->next←p->next;
p←p->next;
free ( r );
else
endif
endwhile.
Endprocedure.
```

Exercise 08:

```
    Type element = record
                val: integer ;
                 next:*element;
                 previous: *element;
    end
    var head
    begin head←NULL;
    procedure display( var L:*element);
    var p:*element;
    begin
    p←L;
    while (p->next <> NULL ) do
    write (p->val, ”->”);
    p←p->next;
    endwhile
    while(p->previous<> NULL ) do
    write (p->val,”<-“);
    p←p->previous;
    endwhile.
```

Exercise 09:

```
Procedure inserthead(var *l:element,
x:integr;);
Var q:*element;
Begin
Alloacte (q);
q.>val←x;
if (L== null) then
q->next ←null;
q->previous←  null;
L ←  q;
Else
q->next←L;
q->previous←  null;
L->previous← q;
L ←  q;
Endif
End.
Procedure deltehead(var *l:element,x:integr;);
Var q:*element;
Begin
q←L;
if(L=null) then return;
else
L←L->next;
L->previous←  null;
q->next←  null;
Free(q);
Endif
End
Procedure inserttail(var *l:element,x:integr;);
Var q:*element;
Begin
Alloacte (q);
q.>val←x;
if (L== null) then
q->next ←null;
q->previous←  null;
L ←  q;
Else
p←L;
while(P->next<>NULL) then
p←p->next;
endwhile
p->next←q;
q->previous←p;
q->next←NULL;
endif
end
procedure deletetail(var *l:element,x:integr;);
Var q:*element;
Begin
Begin
p←L;
if(L=null) then return;
else
while(p->next<>NULL) then
p←p->next;
endwhile
p->previous->next ← null;
p->previous ←  null;
Free(p);
Endif
End.
```

Exercise 10:

```
    Type element = record
                val: integer ;
                 next:*element;
                 previous: *element;
    end
    var head
    begin head←NULL;
    procedure delete( var L:*element,);
    var p:*element;
    begin
    p←L;
    if (p= NULL ) then
    return;
    else
    while (p <> NULL ) do
    if(p->val <>x) then
    p←p->next;
    else
    q←p;
    p←p->next;
    p->previous←q->previous;
    q->previous->next←q->next;
    free(q );
    endif
    end while
end.
```

Exercise 11: cours

```
  Procedure inserheadc(var l:*element  ,
   value: integer)
  Var p,q:*node;
  Begin
  Allocate(q);
  q->data←value ;
  if(l=NULL) then
  l←q;
  q->next←l;
  else
  p←l ;
  while (p.next<>l) do
  p←p.next ;
  endwhile
  q->next←l;
```

```
  p->next←q;
  l←q;
  end if
  end procedure;
```

Delete a head from a list:

```
Procedure deleteheadc (var L: *element );
Var p,q: *element;
Begin
q←L;
p←L;
 repeat
P ←P->next ;
Until  (P->next = L)
L← q^.next
p^.next← L
free (q) ;
end.
```

procedure addtailc(Var L:element , X:integer)

```
var q, p :*element ;
Begin
Allocate (q)
q->data←X
p←L
repeat
p←p->next
 until  (p->next = L)
p->next←q ;
q->next←L
end
```

procedure deletetailc(var L:*element);

```
var  p, q:*element;
Begin
If(l= null) then return;
else
p←L;
repeat
q←p;
P ←P->next ;
Until  (P->next = L)
q->next←L;
free (p) ;
end.
```

Exercise 12:

Procedure singly(var L:*liste);

```
Var p:*liste;
Begin
P←L;
While(p->next <>L) do
p←p->next;
endwhile
p->next←NULL;
end.
```

Exercise 13:

```
Procedure concatec(var L1,L2:*liste);
Var p1,p2 : *liste ;
Begin
p1←L1;
p2←L2;
if(L1= null ) then L1←L2;
Else
while (p1->next <>L1) do
p1←p1->next;
End while
p1->next←L2;
while (p1->next <>L2) do
p1←p1->next;
Endwhile
p1->next←L1;
endif
end
```