1:اكتب برنامج لايجاد مفكوك العدد

```
program factoralnumber;
var i,f:integer;
begin
readln(i);
f:=1;
for i:=1 to i do
f:=f*i;
writeln(f);
end.
```

2:اكتب برنامج لجمع عددين ثلاث مرات

بحيث تظهر النيجة مباشرة بعد ضغط انتر لكل عملية

```
program hussien;
var a,b,c,i:integer;
begin
for i:=1 to 3 do
begin
readln(a,b);
c:=a+b;
writeln(c);
end;
end.
```

3:برنامج ينتج الشكل الاتي من مصفوفة (5*5)

```
  *
 ***
*****
*****
```

```
program hussien;
var i,j:integer;
d:array[1..5,1..5]of integer;
begin
for i:=1 to 5 do
begin
writeln;
for j:=1 to 5 do
begin
if((j=3)or(i=4)or(i=5)) then
write('*')
else if(i=3)and(j mod 2=0) then
write('*')
else
write(' ');
end;
```

```
end;
end
```

4:واطبع الشكل الاتي

```
*     *
 *   *
 * * *
*     *
 * * *
```

```
program hussien;
var i,j:integer;
d:array[1..5,1..5]of integer;
begin
for i:=1 to 5 do
begin
writeln;
for j:=1 to 5 do
begin
if((i=j)and(i+j<7)) then
write('*')
else if((i+j=6)and(i<4)) then
write('*')
else if(i=3)and(j mod 2=0) then
write('*')
else if((i=5)and(j mod 2=0)) then
write('*')
else if(((i=4)and(i+j=5))or((i=4)and(i+j=9))or((i=5)and(i+j=8)) )then
write('*')
else
write(' ');
end;
end;
end.
```

5:اطبع القطر الرئيسي والقطر الثانوي لمصفوفة(5*5)

```
program hussien;
var i,j:integer;
d:array[1..5,1..5]of integer;
begin
for i:=1 to 5 do
begin
writeln;
for j:=1 to 5 do
begin
if((j=i)or(i+j=6)) then
write('*')
else
write(' ');
end;
end;
end.
```

* اضرب مادام الحديد حاميا

* اذا ضربت فأوجع فان الملامة واحدة

Hussien Ahmmed.T

6:إضافة عناصر الى (stack) بأستخدام(record)

```
program stackaddtion;
var
stack:record
i,top,m,x,g,w,r,k,j,h:integer;
stack:array[1..6] of integer;
item:array[1..10] of integer;
end;
procedure pushstak(top,i,h:integer);
begin
if stack.top=6 then
writeln('error...stack is full')
else
begin
stack.m:=1;
writeln('enter location new item');
readln(stack.g);
writeln('enter new item');
readln(stack.r);
if stack.g>stack.top then
begin
stack.top:=stack.top+1;
stack.stack[stack.top]:=stack.r;
end
else
begin
for stack.k:=stack.top downto 1 do
begin
stack.item[stack.m]:=stack.stack[stack.k];
stack.stack[stack.k]:=0;
stack.m:=stack.m+1;
if stack.k=stack.g then
begin
stack.item[stack.m]:=stack.r;
stack.m:=stack.m+1;
end;
end;
stack.m:=stack.m-1;
for stack.w:=1 to (stack.top+1)  do
begin
stack.stack[stack.w]:=stack.item[stack.m];
stack.item[stack.m]:=0;
stack.m:=stack.m-1;
end;
stack.top:=stack.top+1;
end;
end;
for stack.k:=stack.top downto 1 do
writeln('stack[',stack.k,']=', stack.stack[stack.k]);
writeln('top=',stack.top);
end;
begin
writeln('who many item you want to push');
readln(stack.i);
for stack.i:=1 to stack.i do
pushstak(stack.top,stack.i,stack.h);
end.
```

ملاحظة:**تكون الأضافة في أي موقع تريد أن تضع فيه العنصر الجديد داخل هذا السجل(هذا هو مبدأ عمل هذا البرنامج)**

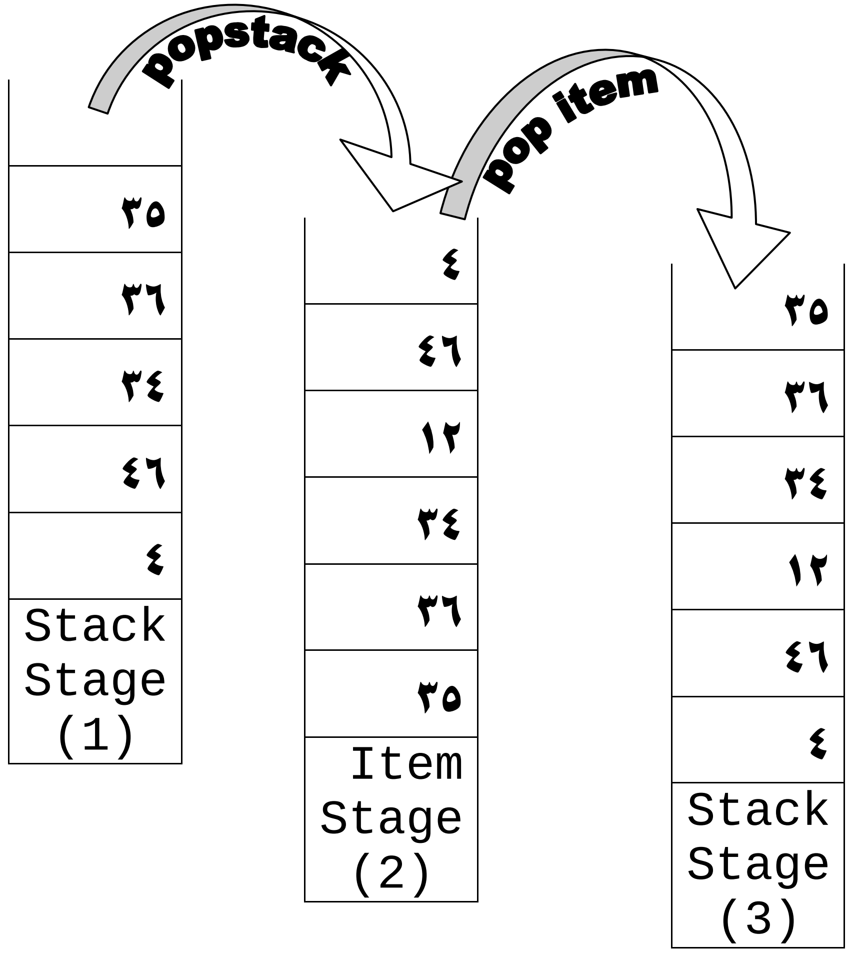
7: مشروع لخوارزميات(stack)

```
program stackoperation;
var
i,top,m,x,g,w,search,r,s,k,j,h:integer;
stack:array[1..6] of integer;
item:array[1..10] of integer;
check:char;
procedure pushstack( var i,x,g,r,top:integer);
begin
if top=6 then
writeln('error...stack is full')
else
begin
m:=1;
writeln('enter location new item');
readln(g);
writeln('enter new item');
readln(r);
if g>top then
begin
top:=top+1;
stack[top]:=r;
end
else
begin
for k:=top downto 1 do
begin
item[m]:=stack[k];
stack[k]:=0;
m:=m+1;
if k=g then
begin
item[m]:=r;
m:=m+1;
end;
end;
m:=m-1;
for w:=1 to (top+1)  do
begin
stack[w]:=item[m];
item[m]:=0;
m:=m-1;
end;
top:=top+1;
end;
end;
for k:=top downto 1 do
writeln('stack[',k,']=', stack[k]);
writeln('top=',top);
end;
procedure popstack(var g,k,h,top,s:integer);
begin
if top=0 then
writeln('error..stack is empty')
else
begin
write('who location you want to delete:');
readln(h);
if h>=top then
begin
stack[top]:=0;
top:=top-1;
end
else
begin
m:=1;
```

```
for i:=top downto 1 do
begin
if i=h then
stack[i]:=0
else
begin
item[m]:=stack[i];
stack[i]:=0;
m:=m+1;
end;
end;
m:=m-1;
s:=1;
for i:=m downto 1 do
begin
stack[s]:=item[i];
item[i]:=0;
s:=s+1;
writeln('s=',s);
end;
top:=s-1;
end;
writeln('the contain of stack is');
for j:=top downto 1 do
writeln('stack[',j,']=', stack[j]);
writeln('top=',top);
end;
end;
procedure checkfullstack(chek:char;top:integer);
begin
if top=6 then
writeln('stack is full')
else
writeln('stack is not full');
end;
procedure checkemptystack(chek:char;top:integer);
begin
if top=0 then
writeln('stack is empty')
else
writeln('stack is not empty');
end;
procedure clearstack(h,top:integer);
begin
if h=0 then
begin
for j:=top downto 1 do
stack[j]:=0;
writeln('stack=',stack[j]);
end
else
begin
for j:=top downto 1 do
writeln('stack[',j,']=', stack[j]);
writeln('top=',top);
end;
end;
begin
writeln('who many item you want to enter in this stack');
readln(x);
writeln('enter the element');
for top:=1 to x do
begin
write('stack[',top,']=');
readln(stack[top]);
end;
writeln('who do you want do in this stack press');
writeln('(1)push ,(2)pop ,(3)check stack ,(4)clearstack');
readln(search);
if search=1 then
```

```
begin
writeln('who many item you want to push');
readln(x);
for i:=1 to x do
pushstack(i,x,g,r,top);
end;
if search=2 then
begin
writeln('who many item you want to pop');
readln(g);
for k:=1 to g do
popstack(g,k,h,top,s);
end;
if search=3 then
begin
write('do you want to check full stack(y,n): ');
readln(check);
if check='y' then
checkfullstack(check,top);
write('do you want to check empty stack(y,n): ');
readln(check);
if check='y' then
checkemptystack(check,top)
end;
if search=4 then
begin
writeln('press(0) to clear  stack other to show final result');
readln(h);
clearstack(h,top);
end;
end.
```

Note:the come is digraim for work program in add



Add item (12) in location(3)

<u>Fig(1) add stack</u>

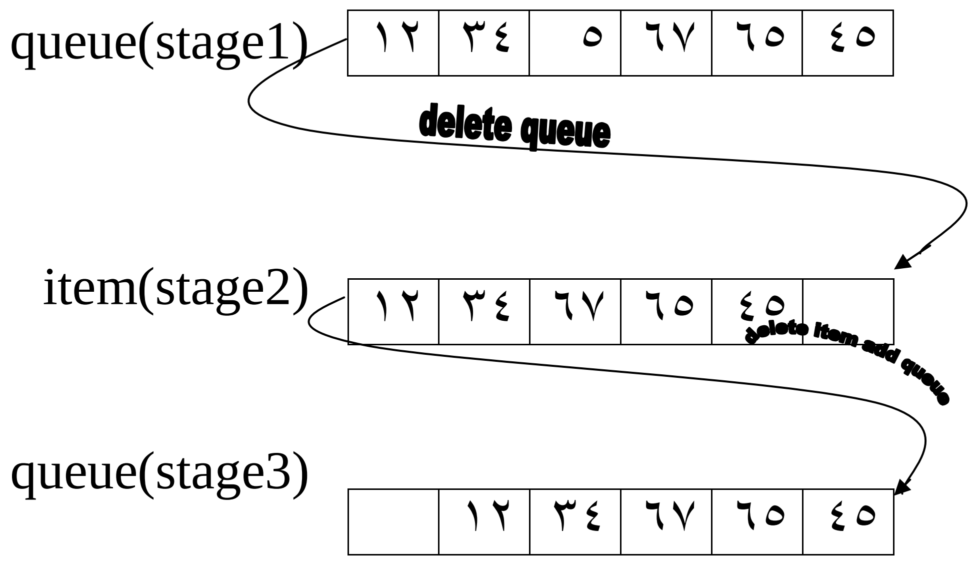
8: مشروع لخوارزميات (Queue)

```
program queueoperation;
var
i,rear,m,front,x,g,w,r,k,j,h:integer;
queue:array[1..6] of integer;
item:array[1..10] of integer;
check:char;
procedure addqueue( var i,x,g,r,rear,front:integer);
begin
if rear=6 then
writeln('error...queue is full')
else
begin
m:=1;
writeln('enter location new item');
readln(g);
writeln('enter new item');
readln(r);
if g>rear then
begin
rear:=rear+1;
queue[rear]:=r;
if front=0 then
front:=1;
end
else
begin
for k:=front to rear do
begin
if k=g then
begin
item[m]:=r;
m:=m+1;
end;
item[m]:=queue[k];
queue[k]:=0;
m:=m+1;
end;
for w:=front to (rear+1) do
begin
queue[w]:=item[w];
item[m]:=0;
end;
rear:=w;
end;
if front=0 then
front:=1;
end;
for k:=rear downto front do
writeln('queue[',k,']=', queue[k]);
writeln('rear=',rear,'front=',front);
end;
procedure deletequeue(var g,k,h,rear,front:integer);
begin
if front=0 then
writeln('error..queue is empty')
else
begin
if front=rear then
begin
rear:=0;
front:=0;
queue[rear]:=0;
```

```
writeln('queue[rear]=',queue[rear]);
writeln('rear=',rear,'front=',front);
writeln('error..queue is empty');
end
else
begin
write('who location you want to delete:');
readln(h);
m:=1;
for i:=front to rear do
begin
if i=h then
queue[i]:=0
else
begin
item[m]:=queue[i];
queue[i]:=0;
m:=m+1;
end;
end;
m:=1;
for i:=(front+1) to (rear+1) do
begin
queue[i]:=item[m];
m:=m+1;
end;
front:=front+1;
writeln('the contain of queue is');
for j:=rear downto front do
writeln('queue[',j,']=', queue[j]);
writeln('rear=',rear,'front=',front);
end;
end;
end;
procedure checkfullqueue(chek:char;rear:integer);
begin
if rear=6 then
writeln('queue is full')
else
writeln('queue is not full');
end;
procedure checkemptyqueue(chek:char;front:integer);
begin
if front=0 then
writeln('queue is empty')
else
writeln('queue is not empty');
end;
procedure clearqueue(h,rear,front:integer);
begin
if h=0 then
begin
for j:=front to rear do
queue[j]:=0;
writeln('queue=',queue[j]);
end
else
begin
for j:=rear downto front do
writeln('queue[',j,']=', queue[j]);
writeln('rear=',rear,'front=',front);
end;
end;
begin
```

```
writeln('who many item you want to enter in this queue');
readln(x);
writeln('enter the element');
for rear:=1 to x do
begin
write('queue[',rear,']=');
readln(queue[rear]);
end;
if rear<>0 then
front:=1
else
front:=0;
writeln('who many item you want to add');
readln(x);
for i:=1 to x do
addqueue(i,x,g,r,rear,front);
writeln('who many item you want to detet');
readln(g);
for k:=1 to g do
deletequeue(g,k,h,rear,front);
write('do you want to check full queue(y,n): ');
readln(check);
if check='y' then
checkfullqueue(check,rear);
write('do you want to check empty queue(y,n): ');
readln(check);
if check='y' then
checkemptyqueue(check,front);
writeln('press(0) to clear  queue other to show final result');
readln(h);
clearqueue(h,rear,front);
end.
```

Diagram for work stage detete



Delete item (5) from loction (3)

Fig(2) delete queue

Program circular queue

```
program Circular_queue;
const size=6;
var
cq:array[1..size] of integer;
item:integer;
front,rear,choice,item1,i,l,m :integer;
procedure addcq(var
front,rear:integer;item:integer);
begin
if rear=size then
rear:=1
else
rear:=rear+1;
if rear=front then
writeln('error..the co is full')
else
cq[rear]:=item;
end;
procedure deletecq(var
front,rear:integer;item:integer);
begin
if rear=front then
writeln('error..the cq is empty')
else
begin
if front = size then
front:=1
else
front:=front+1;
item:=cq[front];
cq[front]:=0;
end
end;
begin {main program}
for i:=1 to size do
cq[1]:=0;
rear:=1;
front:=1;
repeat
writeln('program circular queue operation');
writeln('________________________________');
writeln('1-insertion operation');
writeln('2-delete operation');
writeln('3-Display the content op the CQ');
writeln('4-exit');
writeln('select your choice');
readln(choice);
case choice of
1:begin
writeln('Hwomany element you like to enter?');
readln(m);
for i:=1 to m do
begin
writeln('enter element new element');
readln(item1);
addcq(front,rear,item1)

end;
end;
2:begin
writeln('hwomany element you like to delete?');
readln(l);
for i:=1 to l do
begin
deletecq(front,rear,item1)
end;
end;
```

```
3:begin
writeln('the conten of CQ is');
for i:=1 to size do
writeln('cqueue[',i,']=',cq[i]);
writeln;
writeln('front=',front,'          rear=',rear);
end;
4:end;
until choice=4
end.
```

Program double ended queue

```
program DEQ_operation;
var
i,rear,m,front,x,g,w,r,k,j,h,chiceme,chiceme1,xite
m,item:integer;
queue:array[1..6] of integer;
check:char;
procedure addqueue( var
i,x,g,r,rear,front:integer);
begin
if rear=6  then
writeln('error...queue is full')
else
begin
rear:=rear+1;
if rear>0 then
begin
writeln('enter new item');
readln(item);
queue[rear]:=item
end;
if rear<0 then
begin
rear:=abs(rear);
writeln('enter new item');
readln(item);
queue[rear]:=item;
rear:=-rear;
end;
if rear=1 then
front:=1;
end;
end;
procedure deletequeue(var
g,k,h,rear,front:integer);
begin
if front=0 then
writeln('error..queue is empty')
else
begin
if front=rear then
begin
rear:=abs(rear);
queue[rear]:=0;
rear:=0;
front:=0;
writeln('massage..queue become is empty');
end
else
begin
if front<0 then
begin
front:=-front;
queue[front]:=0;
front:=-front;
writeln('top=',front);
front:=front+1;
```

```
end;
if front>0 then
begin
queue[front]:=0;
front:=front+1;
end;

end;
end;
end;
procedure checkfullqueue(chek:char;rear:integer);
begin
if rear=6 then
writeln('queue is full')
else
writeln('queue is not full');
end;
procedure
checkemptyqueue(chek:char;front:integer);
begin
if front=0 then
writeln('queue is empty')
else
writeln('queue is not empty');
end;
procedure clearqueue(h,rear,front,x:integer);
begin
if h=0 then
begin
for j:=x downto 1 do
queue[j]:=0;
writeln('queue=',queue[j]);
end
else
begin
for j:=rear downto front do
writeln('queue[',j,']=', queue[j]);
writeln('rear=',rear,'front=',front);
end;
end;
begin  {main program}
writeln('who many item you want to enter in this
queue');
readln(x);
writeln('enter the element');
for rear:=1 to x do
begin
write('queue[',rear,']=');
readln(queue[rear]);
end;
if rear<>0 then
front:=1
else
front:=0;
repeat
writeln('program double ended queue operation');
writeln('________________________________');
writeln('1-Right to left');
writeln('2-Left to right');
writeln('3-exit');
readln(chiceme);
case chiceme of
1:
begin
repeat
if rear < 0 then
rear:=-rear;
if front<0 then
front:=-front;
writeln;
writeln('program double ended queue operation
right to  left');
```

```
writeln('*****************************************
**********');
writeln('1-insert op');
writeln('2-delete');
writeln('3-check full');
writeln('4-check empty');
writeln('5-by zero');
writeln('6-Display contain queue');
writeln('7-return');
readln(chiceme1);
case chiceme1 of
1:
begin
writeln('who many item you want to add');
readln(x);
for i:=1 to x do
addqueue(i,x,g,r,rear,front);
end;
2:begin
writeln('who many item you want to detet');
readln(g);
for k:=1 to g do
deletequeue(g,k,h,rear,front);
end;
3:begin
checkfullqueue(check,rear);
end;
4:begin
checkemptyqueue(check,front) ;
end;
5: begin
clearqueue(h,rear,front,x);
end;
6:begin
for i:= front to rear do
writeln('queue[',i,']=',queue[i]);
writeln('front=',front,'
rear=',rear);
end;
7:end;
until chiceme1=7
end;  {ending right to left}
2: {left to right}
begin
xitem:=front;
front:=-rear;
rear:=-xitem;
repeat
writeln;
writeln('program double ended queue operation left
to  right');
writeln('*****************************************
**********');
writeln('1-insert op');
writeln('2-delete');
writeln('3-check full');
writeln('4-check empty');
writeln('5-by zero');
writeln('6-display contain queue');
writeln('7-return');
readln(chiceme1);
case chiceme1 of
1:
begin
writeln('who many item you want to add');
readln(x);
for i:=1 to x do
begin
addqueue(i,x,g,r,rear,front);
if rear>0 then
rear:=-rear;
end;
```

```
end;
2:begin
writeln('who many item you want to detet');
readln(g);
for k:=1 to g do
begin
deletequeue(g,k,h,rear,front);
if front>0 then
front:=-front;
end;
end;
3:begin
if front<0 then
front:=-front;
if rear<0 then
rear:=-rear;

checkfullqueue(check,rear);
end;
4:begin
if front<0 then
front:=-front;
if rear<0 then
rear:=-rear;
checkemptyqueue(check,front) ;
end;
5: begin
if front<0 then
front:=-front;
if rear<0 then
rear:=-rear;
clearqueue(h,rear,front,x);
end;
6:begin
if front<0 then
front:=abs(front);
if rear<0 then
rear:=abs(rear);
for i:= 1 to 6 do
writeln('queue[',i,']=',queue[i]);
front:= 6-front+1;
rear:=6-rear+1;
if front>6 then
begin
front:=0;
rear:=0;
end;
writeln('front=',front,'
rear=',rear);
end;
7:end;
until chiceme1=7
end;  {ending left to right}
3:end;
until chiceme=3
end.
```

By:

1.Hussien aHmmed TalaB

Program of linker list operation it do:

<u>('1=creat linker list');</u>
<u>('2=display list');</u>
<u>('3=add new item before element you chose it');</u>
<u>('4==add new item after element you chose it');</u>
<u>('5==detete item cetain value');</u>
<u>('6==detete by give position');</u>
<u>('7==adding at ending linker list')</u>

```
program linker_list;
type element=integer;
 ptr=^node;
node=record
data:element;
link:ptr
end;
var p2,p,start,p3:ptr;
var i,n,a,k:integer;
var item,item1:element;
procedure acreatlinkerlist(start:ptr);
begin
write('hwo many iten yoy enter: ');
read(n);
for i:=1 to n do
begin
readln(p^.data);
if i<>n then
new(p2)
else
p2:=nil;
p^.link:=p2;
P:=p2;
end;
end;

procedure DETETEVVALUE(start:ptr;item:element);
var q:ptr;
begin
p:=start;

while p^.data<> item do
begin
q:=p;
p:=p^.link
end;
q^.link:=p^.link;
dispose(p);
end;
procedure ADDBEFOR(start:ptr;item:element);
var p3:ptr;
begin
new(p3);
p3^.data:=p^.data;
p^.data:=item;
p3^.link:=p^.link;
p^.link:=p3;
end;
```

```
procedure ADDAFTER(start:ptr;item:element);
var p3:ptr;
begin
new(p3);
p3^.data:=item;
p3^.link:=p^.link;
p^.link:=p3;
end;
procedure showlist(start:ptr);
begin
p:=start;
repeat
writeln('link.data=',p^.data);
p:=p^.link;
until p=nil
end;
procedure DETETEonec(p:ptr);
var q:ptr;
begin
q:=p^.link;
p^.data:=q^.data;
p^.link:=q^.link;
dispose(q);

end;

begin {main program}
a:=5;
start:=p;
repeat
writeln;
writeln('chose your selection please............');
writeln('1=creat linker list');
writeln('2=display list');
writeln('3=add new item before element you chose it');
writeln('4==add new item after element you chose it');
writeln('5==detete item cetain value');
writeln('6==detete by give position');
writeln('7==adding at ending linker list');
writeln('9=exit');
readln(a);
case a of
1:
acreatlinkerlist(start);
2:
showlist(start);
3:
begin
writeln('how the element you want to store new item befor it: ');
readln(item1);
p:=start;
while p^.data <> item1 do
p:=p^.link;
writeln('HOW MANY ITEM YOU LIKE TO ENTER');
readln(k);
for i:=1 to k do
begin
writeln('enter the element');
readln(item);
ADDBEFOR(start,item);
```

```
end;
end;
4:
begin
writeln('how the element you want to store new item after it: ');
readln(item1);
p:=start;
while p^.data <> item1 do
p:=p^.link;
writeln('HOW MANY ITEM YOU LIKE TO ENTER');
readln(k);
for i:=1 to k do
begin
writeln('enter the element');
readln(item);
ADDAFTER(start,item);
end;
end;
5:
begin
writeln('GIVE THE VALUE OF ELEMENT YOU WANT TO DETETE IT');
readln(item);
DETETEVVALUE(start,item);
end;
6:
begin
writeln('how position the element');
readln(k);
for i:=1 to k do
begin
p2:=p;
p:=p^.link;
end;
writeln('whomant item you like to detete');
readln(k);
for i:=1 to k do
DETETEonec(p);
end;
7:
begin
while p^.link <> nil do
p:=p^.link;
writeln('HOW MANY ITEM YOU LIKE TO ENTER');
readln(k);
for i:=1 to k do
begin
writeln('enter the element');
readln(item);
ADDAFTER(start,item);
end;
end;
9:end;
until a=9
end.
```