```
Program Ex_3;
type AdresaCandidat=^Candidat;
Candidat=record
NumePrenume:string;
NotaMedie:real;
Urm:AdresaCandidat;
end;
var p,c,u,p1:AdresaCandidat;
n:integer;

procedure creare;
var i:integer;
begin
write('n='); readln(n);
new(c);
readln(c^.NumePrenume);
readln(c^.NotaMedie);
c^.Urm:=nil;
p:=c; u:=c;
for i:=2 to n do begin
new(c);
readln(c^.NumePrenume);
readln(c^.NotaMedie);
c^.Urm:=nil;
u^.Urm:=c;
u:=c;
end;
end;


procedure afisare;
begin
c:=p;
while c<>nil do begin
writeln(c^.NumePrenume);
writeln(c^.NotaMedie);
c:=c^.Urm;
end;
end;

procedure candidati;
begin
c:=p;
writeln('Medie peste 7.5 au urmatorii candidati');
while c<>nil do begin
if c^.NotaMedie>7.5 then  begin write(c^.NumePrenume); writeln(' ', c^.NotaMedie);
end;
c:=c^.Urm;
end;
end;

procedure Excludere;
label 1;
var q:AdresaCandidat;
Nume:string;
begin
writeln('Ce candidat isi retrage acetele?');
```

```
readln(Nume);
c:=p;
q:=c;
while c<>nil do begin
if c^.NumePrenume=Nume then goto 1;
q:=c;
c:=c^.Urm;
end;
1:if c=nil then writeln('Nu exista numele') else begin
if c=p then p:=c^.urm else q^.urm:=c^.urm;
end;
end;

procedure Includere;
label 1;
var q:AdresaCandidat;
Nume:string;
begin
new(q);
writeln('Ce candidat isi depune actele?');
readln(q^.NumePrenume);
readln(q^.NotaMedie);
writeln('Dupa cine va fi in lista?');
readln(Nume);
c:=p;
while c<>nil do
begin
if c^.NumePrenume=Nume then goto 1;
c:=c^.Urm;
end;
1:if c=nil then begin
writeln('Nu exista numele'); dispose(q); end
else begin
q^.Urm:=c^.Urm;
c^.Urm:=q;
end;
end;

procedure creare2;
label 1,2;
var q:AdresaCandidat;
begin
new(q);
c:=p;
writeln('Lista 2');
while c<>nil do begin
if c^.NotaMedie>9 then p1:=c; break;
c:=c^.Urm;
end;
if p1<>nil then begin
c:=p1^.Urm;
while c<>nil do begin
if c^.NotaMedie>9 then begin
q^.urm:=c;
q:=c;
end;
end;
```

```
end;
end;


procedure afisare2;
begin
c:=p1;
while c<>nil do begin
writeln(c^.NumePrenume);
writeln(c^.NotaMedie);
c:=c^.Urm;
end;
end;


procedure Excludere2;
label 1;
var q:AdresaCandidat;
begin
c:=p;
q:=c;
while c<>nil do begin
if c^.NotaMedie<6 then goto 1;
q:=c;
c:=c^.Urm;
end;
1:if c=nil then writeln('Nimeni nu are mai putin de 6') else begin
if c=p then p:=c^.urm else q^.urm:=c^.urm;
end;
end;


begin
creare; afisare; candidati; excludere; includere; afisare; creare2; afisare2; excludere2; afisare;
end.
```

```
Program ex_4;

type List=^ListItem;

    ListItem = record

     data : string;

     value : real;

     order : integer;

     urm : List;

    end;


list_item = record

  name : string;
  base : List;
  count : integer;
end;

database = array[1..100] of list_item;



var a:database;
    r,b,v:List;
    c:integer;
    ans:string;



procedure create_list();
  var ans, name:string;
      i:integer;
  begin
    inc(c);
    writeln('List Name : '); readln(name);
    while ans<>'EXIT' do begin
      writeln();
      writeln('   Type EXIT to end list creation');
      writeln('PRESS ENTER TO CONTINUE');
      readln(ans);

      if ans<>'EXIT' then begin
        new(r);
        inc(i);
        writeln(i,'| Data :'); readln(r^.data);
        writeln(i,'| Value :'); readln(r^.value);
        r^.order:=i;
```

```
       if i=1 then b:=r else v^.urm:=r;
       v:=r;
     end else begin
       a[c].base:=b;;
       a[c].name:=name;
       a[c].count:=i;
     end; {SAVE LIST DATA}
   end;
 end;

procedure concat();
  var one,two,i,save:integer;
      name:string;
      q,base,top:List;
  begin
    writeln('1st List Database ID : '); readln(one);
    writeln('2nd List Database ID : '); readln(two);

    inc(c);
    writeln('New list name : ');
    readln(name);
    a[c].name:=name;

    b:=a[one].base;
    r:=b;
    while r<>nil do begin
      new(q);
      inc(i);
      q^.order:=i;
      q^.data:=r^.data;
      q^.value:=r^.value;
      if i=1 then a[c].base:=q else begin
       top^.urm:=q;
      end;
      top:=q;
      r:=r^.urm;
   end;

     save:=i;

   b:=a[two].base;
   r:=b;
   while r<>nil do begin
     new(q);
     inc(i);
     q^.order:=i;
     q^.data:=r^.data;
     q^.value:=r^.value;
     top^.urm:=q;
     top:=q;
     r:=r^.urm;
   end;
 end;


procedure slice();
 var s,i,j,p,cut:integer;
     slicing:boolean;
     name:string;
     q,v:List;
 begin
   writeln('Database ID of the Sliced list : ');
```

```
readln(s);


writeln();


r:=a[s].base;
while r<>nil do begin
  inc(i);
  writeln(i,'.');
  writeln('   data :',r^.data);
  writeln('   value :',r^.value);
  r:=r^.urm;
end;


writeln();
writeln();
writeln('NO. of the element the Slice starts from : ');
readln(cut);


writeln();
writeln('Name of The New List 1 : ');
readln(name);
inc(c);
a[c].name:=name;


writeln();
writeln('Name of The New List 2 : ');
readln(name);
inc(c);
a[c].name:=name;


r:=a[s].base;
```

```
    while r<>nil do begin
     if r^.order=cut then slicing:=true;


     if slicing<>true then begin
       new(q);
       inc(j);
       q^.data:=r^.data;
       q^.value:=r^.value;
       if j=1 then a[c-1].base:=q else v^.urm:=q;
       v:=q;
      end else begin
       new(q);
       inc(p);
       q^.data:=r^.data;
       q^.value:=r^.value;
       if p=1 then a[c].base:=q else v^.urm:=q;
       v:=q;
      end;
      r:=r^.urm;
    end;
  end;


procedure database_display();
  var i:integer;
  begin
    writeln();
    writeln('----LISTS IN THE DATABASE----');
    writeln();
    for i:=1 to c do begin
     writeln(i,'|| ',a[i].name);
    end;
```

```
 end;

procedure list_display();
 var n,i:integer;
 begin
  writeln();
  writeln('ID of the List in the Database : ');
  readln(n);

  writeln();
  writeln();
  writeln('<>-----|',a[n].name,'|-----<>');
  writeln();

  r:=a[n].base;
  while r<>nil do begin
   inc(i);
   writeln(i,'#');
   writeln('   data : ',r^.data);
   writeln('   value : ',r^.value);
   r:=r^.urm;
  end;
 end;

 procedure show();
 var i,n,cnt:integer;
     save:integer;
     min:real;
    ans:string;
     b:List;
  begin
```

```
     writeln();
    writeln('List Database ID : ');
    readln(n);
    writeln();
   writeln('Criteria of display');
    writeln('V - Value');
    writeln('D - Data (aplhabetically)');
    readln(ans);

     if ans = 'V' then begin
       for i:=1 to a[n].count do begin
      r:=a[n].base;
         while r<>nil do begin
          if r^.order >= i then begin
             if r^.order=i then min:=r^.value;

             if r^.value < min then min:=r^.value;
          end;
         writeln(min);
          r:=r^.urm
         end;
       end;
        writeln(min);

     end else if ans = 'D' then begin
     end;
   end;

procedure by_value(n:integer);
 var min:real;
     val:real;
     dat:string;
     save,i:integer;
     box:List;
 begin

for i:=1 to a[n].count do begin      r:=a[n].base;
     min:=r^.value;
    while r<>nil do begin
      if r^.order >= i then begin
         if min > r^.value then min:=r^.value;
      end else save:=r^.order;
       r:=r^.urm;
     end;

     inc(save);
     r:=a[n].base;
    while r<>nil do begin
      if r^.order=save then box:=r; {IF the item order no. = min's position}
       r:=r^.urm;
     end;



     r:=a[n].base;
     while r<>nil do begin
      if r^.value=min then begin
       dat:=box^.data;
      val:=box^.value;

         writeln(box^.value,'<=>',r^.value);

        box^.data:=r^.data;
```

```
         box^.value:=r^.value;


         r^.value:=val;
         r^.data:=dat;
       end;
       r:=r^.urm;
     end;
 end;end;

procedure show();
  var n,i:integer;
      lim:real;
  begin
    writeln('Database List ID : ');
    readln(n);
    writeln('Show values over :');
    readln(lim);

    r:=a[n].base;
    while r<>nil do begin

      if r^.value > lim then begin
        inc(i);
        writeln(i,'#');
        writeln('     data: ',r^.data);
        writeln('     value:',r^.value);
      end;
      r:=r^.urm;
    end;
  end;
```

```
procedure by_value();
 var n:integer;
     val:real;
     dat:string;
     good:boolean;
 begin
   writeln('Database List ID : ');
   readln(n);

   while good<>true do begin

   b:=a[n].base;
   r:=b;
   while r<>nil do begin
     if r=b then begin
       v:=r;
       r:=r^.urm;
     end else if r^.value < v^.value then begin
       dat:=r^.data;
       val:=r^.value;

       r^.value:=v^.value;
       r^.data:=v^.data;

       v^.data:=dat;
       v^.value:=val
     end;
   end;

   r:=a[n].base;
   while r<>nil do begin
```

```
      if r=b then begin
        v:=r;
        good:=true;
        r:=r^.urm;
      end else begin
        if r^.value < v^.value then good:=false;
        r:=r^.urm;
      end;
    end;
  end;
  writeln(good);
end;

procedure menu();
  begin
    writeln();
    writeln('--->PRESS ENTER TO CONTINUE<---');
    readln();
    writeln();
    writeln('<||--------- MENU ---------||>');
    writeln();
    writeln('C - Create List');
    writeln('B - Display Database Lists');
    writeln('D - Display Specific List');
    writeln('K - Concatenate 2 Lists');
    writeln('M - Display Selected List Items');
    writeln('R - Sort list items by value');
    writeln('S - Slice List');
    writeln('E - EXIT');
    writeln();
    readln(ans);
    if ans = 'C' then create_list() else
      if ans = 'B' then database_display() else
        if ans = 'D' then list_display() else
          if ans = 'S' then slice() else
            if ans = 'K' then concat() else
              if ans = 'M' then show()
            else if ans = 'R' then by_value();

  end;

begin
  while ans<>'E' do menu();
end.
```