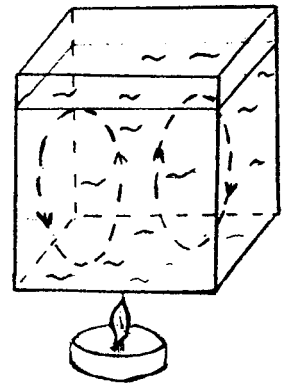
13. Hőáramlás - függőleges síkú kísérlet

Az ábrán látható kísérletet az írásvetítő elé helyezzük. Ha nincs megfelelő téglatest alakú üvegedény, akkor egy Berzéliusz-pohár is megteszi, csak ebben az esetben a kép nem elég éles és a fűrészpor mozgása csak a fényes csíkban látható ami a vízlencsének köszönhető. (8-as ábra)



8. ábra

14. Mágneses erővonalak

Az írásvetítő üveglapjára helyezzük a mágnest, rá egy minél vékonyabb üveglapot és beállítjuk az írásvetítőt, hogy a kép minél élesebb legyen. Az üveglapra vasreszeléket szórunk és kissé megkopogtatjuk.

15. Mágnesek kölcsönhatása

Helyezzük a két mágnest az írásvetítőre. Közelítve egyiket a másikhoz észrevehetjük a vonzást vagy a taszítást, aszerint, hogy milyen pólussal közelítettük egyiket a másik felé. *Hátránya, hogy feketén látszanak a mágnesek a vetítővásznon, és nem lehet látni a pólusokat. Vigyázni kell, hogy miként tartjuk a kezünket a kísérlet során, hogy ne takarja a mágneseket.*

Cseh Gyopár
Kolozsvár

Példa egy rekurzív algoritmusra

Hogyan juthat el egy egér a legrövidebb úton a sajthoz, ha mindketten egy labirintusban vannak? Az alábbi program csak a legrövidebb út hosszát adja meg.

A megoldás alapötlete: megnézzük, hogy a négy szomszédos mező melyikéről lehet legrövidebb úton célba érni.

A program véletlenszerűen generál labirintust, egér- és sajtpozíciót.

Egy lehetséges eredmény:

Labirintus méretei (mxn):

```
m=4 n=6
0 0 1 0 0 0
s 0 0 0 1 1
1 0 1 0 e 0
1 0 0 0 0 0
```

Legrövidebb út: 5

```
program labirint; { Egér a labirintusban. Legrövidebb út a sajtig.
                    C. H. A. Kostner: Programozás felülnézetben
                    Műszaki, Bp. 1988 alapján }
uses crt;
const vegtelen = MaxInt;
   foglalt = 1;
   szabad  = 0;
   sajt    = 2;
   maxindex  = 50;
```

```
type labirintus = array[ 1..maxindex,1..maxindex] of byte;

var a  : labirintus;   { a[ i,j] = 0 szabad, 1 foglalt (fal v. eger),
                              2 sajt, haladni csak vízszintesen
                              vagy függôlegesen lehet }
  m,n,i,j,k,l : byte;
  ut : integer;

  procedure invers (x:char); { alapszín és rajzolószín beállítása }
  begin
    textcolor(white);
    textbackground(black);
    write(' ');
    textcolor(black);
    textbackground(white);
    write(x);
    textcolor(white);
    textbackground(black);
  end;

  function odamehetek (x,y:byte): boolean;  { megvizsgálja, hogy
                                              egy x,y helyre mehet-e}

  begin
     if (x>0) and (x<m+1) and (y>0) and (y<m+1)
       and ((a[ x,y] =szabad) or (a[ x,y] =sajt))
     then odamehetek := true
     else odamehetek := false;
  end;

  function legrovidebb (x,y: byte) : integer;  { a négy szomszédos
                              négyzet melyikébôl lehet legrövidebb
                              úton a sajthoz jutni }
  var min : integer;

     function minimum (a,b:integer):integer;
     begin
       if a<b then minimum := a
           else minimum := b;
     end;

  begin
    if odamehetek(x,y)
    then if a[ x,y] = sajt then legrovidebb := 0
    else
      begin
        a[ x,y] := foglalt;
        min := minimum (legrovidebb (x-1,y), legrovidebb (x+1,y));
        min := minimum (min, legrovidebb (x,y-1));
        min := minimum (min, legrovidebb (x,y+1));
        legrovidebb := min + 1;
        a[ x,y] := szabad;
      end
    else legrovidebb := vegtelen;
  end;

BEGIN
  textcolor(white);
  textbackground(black);
  Clrscr;
  randomize;
  writeln (' Labirintus méretei (mxn):' );
  repeat write (' m=' ); readln (m) until m in[ 1..maxindex] ;
  repeat write (' n=' ); readln (n) until n in[ 1..maxindex] ;
```

```
  for i := 1 to m do     { labirintus generálása véletlenszerűen }
   for j := 1 to n do
   begin
     a[ i,j] := random (3);
     if a[ i,j] = 2 then a[ i,j] :=0;
   end;
a[ random(m)+1,random(n)+1] := sajt;                    { sajt }
  repeat
    k := random(m)+1;                                   { egér }
    l := random(n)+1;
  until a[ k,l] =szabad;

  for i:= 1 to m do                  { a labirintus kirajzolása }
  begin
    for j:=1 to n do
      if (i=k) and (j=1) then invers (' e' )
                else if a[ i,j] =sajt then invers (' s' )
                          else write (a[ i,j] :2);
    writeln;
  end;

  ut := legrovidebb (k,l);
  if ut < vegtelen then writeln (' Legrövidebb út: ' , ut )
           else writeln (' Nincs út!' );
  readln;
END.
```

Borzási Péter

Variációk színes, pezsgő löttyökre: a peroxo-dikromátok világa

Kilencedikes koromban szerettem meg a redoxi folyamatokat. "Kedvenc" vegyületeim közé tartozott a hidrogén-peroxid. Gyakran szórakoztam azzal, hogy minden oxidálhatót feloxidáltam vele.

Egyszer egy érdekes kérdés jutott eszembe. A $K_2Cr_2O_7$-ban a króm maximális oxidációs állapotban van, tehát tovább nem oxidálható, csak redukálódni képes. Ha tehát tömény H_2O_2 oldattal (perhidrol) reagálna, akkor a perhidrolból O_2 fejlődne, a Cr pedig redukálódna. De vajon mivé? Cr^{+3}-má, Cr^{+2}-vé, vagy esetleg mássá?

Kíváncsian készítettem el az oldatokat, s mikor összeöntöttem, nagyon furcsa dolgokat tapasztaltam. Az történt, hogy a két oldat koncentrációjának függvényében az összeöntés után rögtön vagy bizonyos idő után – pár másodperctől negyed óráig – az oldat olyan lett mint a Coca-Cola: barnás színben pezsegni kezdett. A pezsgést okozó gázfejlődés (O_2) időtartama szintén a koncentrációtól függött. Egy idő után kezdett lelassulni és az oldat fokozatosan visszanyerte eredeti narancssárga, dikromátra jellemző színét.

Nagyon elcsodálkoztam és többször is elvégeztem a kísérletet, különféle magyarázatokat keresve és elméleteket állítva fel. Izgalmam csak fokozódott, amikor H_2SO_4-val savanyított, ill. NaOH-dal lúgosított oldatokkal is elvégeztem a kísérletet, mert a következőket tapasztaltam:

— savas közegben gyönyörű szép kék oldat keletkezik, ami a pezsgés megszűntével halványzöldre vált (Cr^{3+})