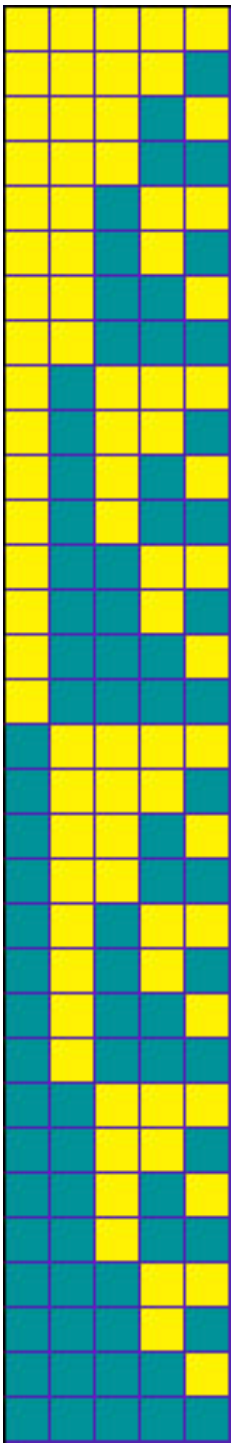
Temă pentru acasă. Baze de numeraţie



Problema 1. Desenul alăturat reprezintă rezultatul numărării în baza 2 pe un registru de lungime 5 folosind ca cifre culorile galben şi albastru. Desenul a fost generat automat de un program realizat în Visual C++ utilizând, în mod esenţial, *algoritmul numărării* în baza 2.

Incercaţi să reprezentaţi (cu creioane colorate!) rezultatul numărării în baza 3 pe un registru de lungime 4, folosind culorile roşu, galben şi albastru.

Problema 2. *Algoritmul numărării* în baza b pe un registru de lungime n rezolvă următoarea problemă: se consideră un *alfabet* format dintr-o mulţime finită şi ordonată de $b \geq 2$ elemente (un b-uplu) şi un număr natural $n \geq 1$. Elementele alfabetului se numesc *cifre* iar prima cifră este numită *zero*. Se cere să se genereze toate cele b^n n-uple distincte care se pot forma cu cifrele considerate.

Algoritmul este binecunoscut: folosim un registru cu n locaţii, toate ocupate iniţial cu cifra zero (*registrul nul*), pe care îl *incrementăm* în mod repetat până când se revine la registrul nul. Fiecare incrementare a registrului se execută *avansând* pe rând locaţiile sale, de la dreapta la stânga, atât timp cât există *transport*. Avansarea unei locaţii este *fără transport* când cifra curentă este înlocuită cu succesoarea ei în alfabet, şi *cu transport* când se înlocuieşte ultima cifră din alfabet cu cifra zero.

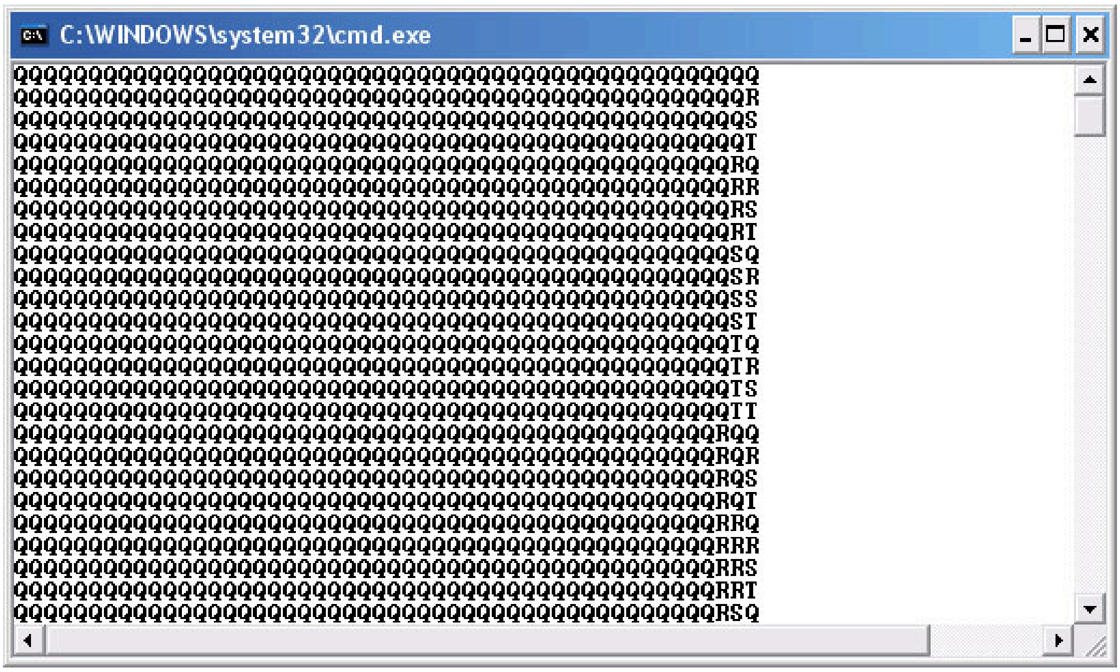
Problemă: descrieţi în pseudo-cod algoritmul numărării în baza 4 pe un registru de lungime oarecare, folosind alfabetul (Q, R, S, T).

Problema 3. Se ştie că un octet (registru de 8 biţi) poate fi reprezentat prin două cifre hexazecimale. Să se listeze (manual sau printr-un program C), pe două coloane alătuate, rezultatul numărării pe un octet în binar şi în hexal.

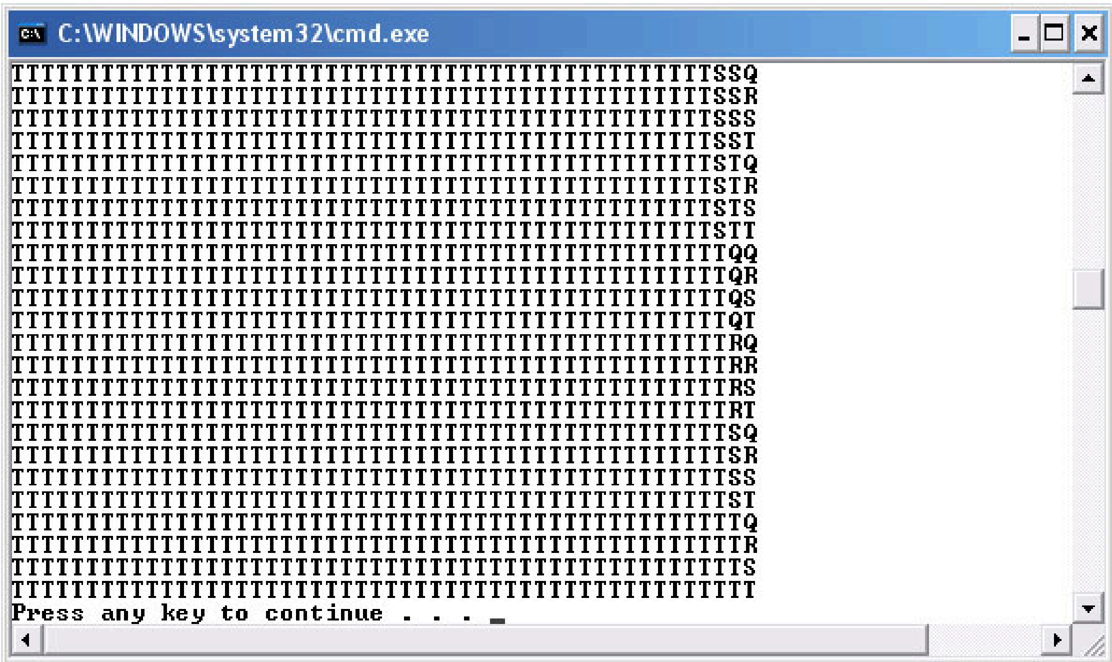
Prezentăm mai jos rezolvarea printr-un program C a problemei 2.

```
/* Numararea in baza 4
cu cifrele {Q, R, S, T} */

#include<iostream>
using namespace std;

const int dim=50;          //lungimea registrului
char avansare(char cifra){
    if(cifra=='Q') return 'R';
    if(cifra=='R') return 'S';
    if(cifra=='S') return 'T';
    return 'Q';            //Q este cifra zero
}

int incrementare(char reg[]){
    int i;
    for(i=0; i<dim; i++){
        reg[i]=avansare(reg[i]);
        if(reg[i]!='Q') break;
    }
    return i;              //i==dim <--> registrul nul
}

void afisare(char reg[]){
    int i;
    for(i=dim-1;i>=0; i--)
        cout<<reg[i];
    cout<<endl;
    return;
}

int main(){
    char registru[dim];
    int i;
    for(i=0;i<dim; i++){   // registrul nul initial
        registru[i]='Q';
    }
    do{                    // numararea propriu-zisa
        afisare(registru);
    }while(incrementare(registru)<dim);
    return 0;
}
```

Iată începutul



şi sfârşitul rulării cu `dim=50` a programului:



Estimaţi cât a durat execuţia de mai sus, ştiind că rularea cu `dim=5` durează exact o secundă.

Problema 4. Completaţi programul pentru calculul sumei

$$2^{2^1} + 2^{2^2} + 2^{2^2} + 2^{2^3} + \cdots + 2^{2^n}$$

prezentat în Tema 1 cu următoarele funcţii:

(a) `void bin2dec(registru d, registru s){...}`
primeşte în registrul sursă `s` un număr scris în baza 10 şi îl scrie în baza 2 în registrul destinaţie `d`;

(b) `void dec2bin(registru d, registru s){...}`
primeşte în registrul sursă `s` un număr scris în baza 2 şi îl scrie în baza 10 în registrul destinaţie `d`;

(c) `void amplifica(registru a, registru b){...}`
calculează produsul dintre numerele `a` şi `b` (în baza 10) şi rezultatul îl depune în `a`.