## Temă pentru acasă. Editarea şi rularea programelor

Editaţi şi rulaţi următorul program:

```
#include<iostream>
using namespace std;
const int dim=100;
typedef int registru[dim];

void initializeaza(registru w){     //w=1
    w[0]=1;
    for(int i=1; i<dim; i++) w[i]=0;
    return;
}
void dubleaza(registru w){
    int t=0;        // transportul initial e nul
    for(int i=0; i<dim; i++){
        int aux=t+2*w[i];
        w[i]=aux%10;//noul w[i]=ultima cifra a lui aux
        t=aux/10;   //noul t=prima cifra a lui aux
    }
    return;
}
void afiseaza(registru w){
    int i;
    for(i=dim-1; i>=0 && w[i]==0; i--)
        ;           //sarim peste zerourile din fata
    if(i<0) cout<<0;
    else for( ; i>=0; i--){
        cout<<w[i]; //afisam incepand cu prima cifra nenula
        if(i%3==0) cout<<' ';
    }
    cout<<endl;
    return;
}
int main(){
    int n=80;
    registru v;
    initializeaza(v);
    for(int i=1;i<=n;i++){
        cout<<"doi la puterea "<<i<<" face ";
        dubleaza(v);
        afiseaza(v);
    }
    return 0;
}
```

Atenţie, nu veţi putea folosi *copy&paste* cu tastele `<ctrl-C, ctrl-V>` deoarece în acest fişier pdf textul sursă al programului a fost înglobat ca imagine (captură de ecran cu `<alt print-screen>`). Scopul temei este să exersaţi redactarea textelor cu editorul mediului de dezvoltare MS Visual Studio.

Rezultatul rulării trebuie să arate astfel:

doi la puterea 1 face 2
doi la puterea 2 face 4
doi la puterea 3 face 8
doi la puterea 4 face 16
doi la puterea 5 face 32
doi la puterea 6 face 64
doi la puterea 7 face 128
doi la puterea 8 face 256
doi la puterea 9 face 512
doi la puterea 10 face 1 024
doi la puterea 11 face 2 048
doi la puterea 12 face 4 096
doi la puterea 13 face 8 192
doi la puterea 14 face 16 384
doi la puterea 15 face 32 768
doi la puterea 16 face 65 536
doi la puterea 17 face 131 072
doi la puterea 18 face 262 144
doi la puterea 19 face 524 288
doi la puterea 20 face 1 048 576
doi la puterea 21 face 2 097 152
doi la puterea 22 face 4 194 304
doi la puterea 23 face 8 388 608
doi la puterea 24 face 16 777 216
doi la puterea 25 face 33 554 432
doi la puterea 26 face 67 108 864
doi la puterea 27 face 134 217 728
doi la puterea 28 face 268 435 456
doi la puterea 29 face 536 870 912
doi la puterea 30 face 1 073 741 824
doi la puterea 31 face 2 147 483 648
doi la puterea 32 face 4 294 967 296
doi la puterea 33 face 8 589 934 592
doi la puterea 34 face 17 179 869 184
doi la puterea 35 face 34 359 738 368
doi la puterea 36 face 68 719 476 736
doi la puterea 37 face 137 438 953 472
doi la puterea 38 face 274 877 906 944
doi la puterea 39 face 549 755 813 888
doi la puterea 40 face 1 099 511 627 776
doi la puterea 41 face 2 199 023 255 552
doi la puterea 42 face 4 398 046 511 104
doi la puterea 43 face 8 796 093 022 208
doi la puterea 44 face 17 592 186 044 416
doi la puterea 45 face 35 184 372 088 832
doi la puterea 46 face 70 368 744 177 664
doi la puterea 47 face 140 737 488 355 328
doi la puterea 48 face 281 474 976 710 656
doi la puterea 49 face 562 949 953 421 312
doi la puterea 50 face 1 125 899 906 842 624
doi la puterea 51 face 2 251 799 813 685 248
doi la puterea 52 face 4 503 599 627 370 496

```
doi la puterea 53 face 9 007 199 254 740 992
doi la puterea 54 face 18 014 398 509 481 984
doi la puterea 55 face 36 028 797 018 963 968
doi la puterea 56 face 72 057 594 037 927 936
doi la puterea 57 face 144 115 188 075 855 872
doi la puterea 58 face 288 230 376 151 711 744
doi la puterea 59 face 576 460 752 303 423 488
doi la puterea 60 face 1 152 921 504 606 846 976
doi la puterea 61 face 2 305 843 009 213 693 952
doi la puterea 62 face 4 611 686 018 427 387 904
doi la puterea 63 face 9 223 372 036 854 775 808
doi la puterea 64 face 18 446 744 073 709 551 616
doi la puterea 65 face 36 893 488 147 419 103 232
doi la puterea 66 face 73 786 976 294 838 206 464
doi la puterea 67 face 147 573 952 589 676 412 928
doi la puterea 68 face 295 147 905 179 352 825 856
doi la puterea 69 face 590 295 810 358 705 651 712
doi la puterea 70 face 1 180 591 620 717 411 303 424
doi la puterea 71 face 2 361 183 241 434 822 606 848
doi la puterea 72 face 4 722 366 482 869 645 213 696
doi la puterea 73 face 9 444 732 965 739 290 427 392
doi la puterea 74 face 18 889 465 931 478 580 854 784
doi la puterea 75 face 37 778 931 862 957 161 709 568
doi la puterea 76 face 75 557 863 725 914 323 419 136
doi la puterea 77 face 151 115 727 451 828 646 838 272
doi la puterea 78 face 302 231 454 903 657 293 676 544
doi la puterea 79 face 604 462 909 807 314 587 353 088
doi la puterea 80 face 1 208 925 819 614 629 174 706 176
Press any key to continue . . .
```

Observaţie. Datorită limitărilor inerente privind mărimea zonei de memorie alocată datelor de tip numeric, numărul de cifre al acestor date este relativ mic. Pe compilatorul MS Visual Studio C++ 2008 sau 2010, cel mai mare număr întreg care poate fi reprezentat direct, ca valoare a unei singure variabile de tip întreg (`unsigned int`, mai precis) este $2^{32} - 1$, adică 4294967295, un număr întreg cu 10 cifre. Pentru calculul cu *numere mari* (numere cu multe cifre, peste precizia compilatorului) programatorul trebuie să îşi definească propriile *structuri de date* în care să memoreze aceste numere şi să elaboreze funcţii de operare cu acestea.

In programul de mai sus numerele sunt memorate cifră cu cifră în *tablouri* (matrici linie) de 100 de elemente de tip `int` iar funcţia `void dubleaza(registru w)` aplică algoritmul înmulţirii cu 2 pentru a dubla numărul stocat în tabloul `w`.

Dificultatea programului este de nivel mediu, începătorii trebuie numai să-l editeze! Cei care se consideră avansaţi, pot încerca să implementeze cele patru operaţii aritmetice (adunarea, scăderea, înmulţirea şi împărţirea) aplicată acestor numere. Iată, de exemplu, funcţia care adună primele două argumente şi depune rezultatul în al treilea:

```
void aduna(registru a, registru b, registru c){
    int i,t,aux;
    t=0;            // transportul initial e nul
    for(i=0;i<dim;i++){
        aux=t+a[i]+b[i];
        c[i]=aux%10;//noul c[i]=ultima cifra a lui aux
        t=aux/10;   //noul t=prima cifra a lui aux
    }
    return;
}
```

In final o temă pentru avansaţi: încercaţi să calculaţi suma

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

cu acest program, pentru valori ale lui $n$ cât mai mari ($n \geq 10$).