

Exercise 4:

```
#include<stdio.h>

void main(){
    int nbs[3], max, min, sum = 0, prod = 1, i=0;

    printf("Entrez trois nombres : ");
    scanf("%i%i%i", nbs, nbs + 1, nbs + 2);
    max = min = nbs[0];

    for (;i<3; i++){
        if(max < nbs[i])
            max = nbs[i];

        if(min > nbs[i])
            min = nbs[i];

        sum += nbs[i];
        prod *= nbs[i];
    }

    float avg = sum / 3.0;

    printf("\nSomme = %i\nProduit = %i\nMoyenne = %1.2f\nMax = %i\nMin = %i\n", sum, prod, avg, max, min);
}
```

```
Windows PowerShell
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> gcc Exo4.c -o test
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
Entrez trois nombres : 23 45 67

Somme = 135
Produit = 69345
Moyenne = 45.00
Max = 67
Min = 23
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> █
```

Exercise 5:

Alternative 1 :

```
#include<stdio.h>

void main(){
    int nbr = 0, i = 4, chiffre[5];

    do{
        printf("Entrez un entier de 5 chiffres : ");
        scanf("%i", &nbr);
    }while (nbr < 9999 || nbr > 99999);

    for(;nbr > 0;i--){
        int q = nbr%10;
        chiffre[i] = q;
        nbr /= 10;
    }

    for(i =0;i<5;i++)
        printf("%i ", chiffre[i]);
}
```

Alternative 2:

```
#include<stdio.h>
#include<string.h>

void main(){
    char nombre[100];

    do{
        printf("Entrez un nombre de 5 chiffres : ");
        scanf("%s", nombre);
    }while (strlen(nombre) > 5 || strlen(nombre) < 5);

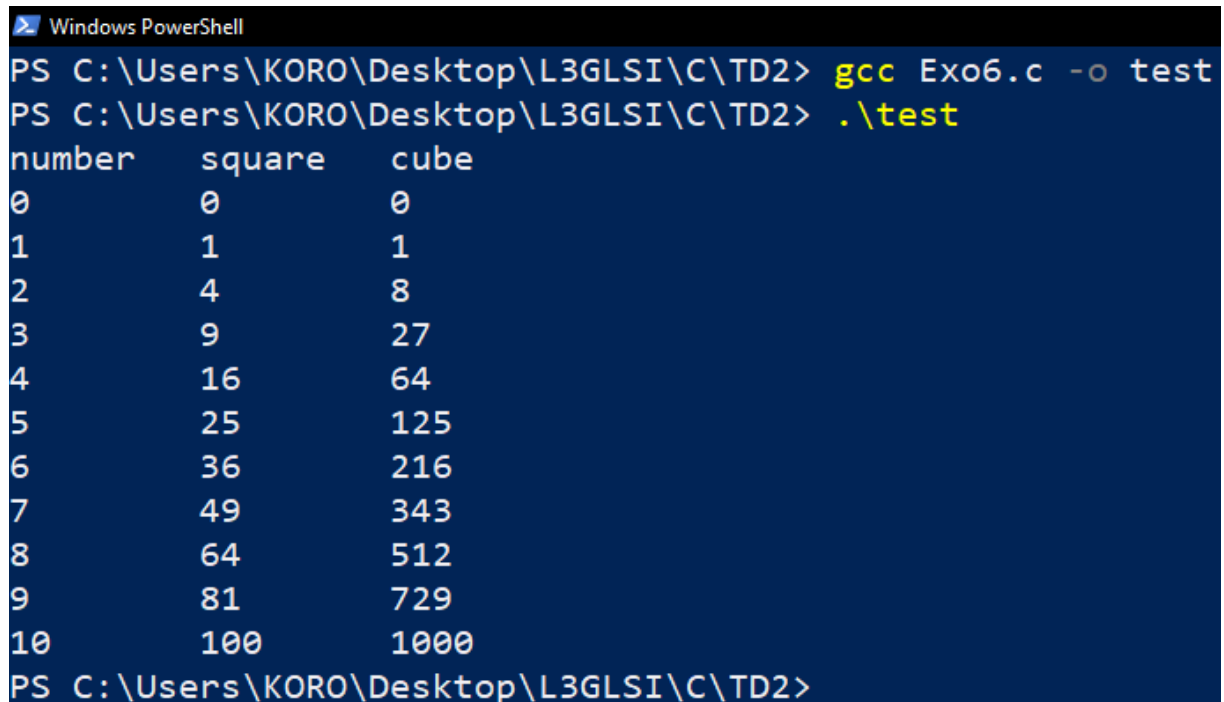
    for(int i = 0; i < strlen(nombre) - 1; i++)
        printf("%i, ", (int)nombre[i] - 48);
    printf("%i\n", (int)nombre[4] - 48);
}
```

```
Windows PowerShell
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> gcc Exo5-1.c -o test
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
Entrez un nombre de 5 chiffres : 00000
0, 0, 0, 0, 0
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
Entrez un nombre de 5 chiffres : 12345
1, 2, 3, 4, 5
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> █
```

Exercise 6:

```
void main(){
    printf("number    square    cube\n");

    for (int i=0; i < 11; i++)
        if(i < 3)
            printf("%i %8i %8i\n", i, i * i, i * i * i);
        else if (i == 3)
            printf("%i %8i %9i\n", i, i * i, i * i * i);
        else if (i == 4)
            printf("%i %9i %8i\n", i, i * i, i * i * i);
        else if (i > 4 && i < 10)
            printf("%i %9i %9i\n", i, i * i, i * i * i);
        else if (i == 10)
            printf("%i %9i %9i\n", i, i * i, i * i * i);
}
```



Windows PowerShell

```
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> gcc Exo6.c -o test
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
```

number	square	cube
0	0	0
1	1	1
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

```
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2>
```

Exercise 7:

```
#include<stdio.h>

void main(){
    int nbs[10], max[2], min[2], i = 0;

    printf("Entrez 10 entiers : ");

    for(;i < 10;)
        scanf("%i", nbs + i++);

    max[0] = min[0] = nbs[0];

    for(i = 0;i < 10;i++)
        if(max[0] < nbs[i])
            max[0] = nbs[i];
        else if(min[0] > nbs[i])
            min[0] = nbs[i];

    max[1] = min[0];
    min[1] = max[0];

    for(--i;i >= 0; i--)    if(max[1] < nbs[i] && nbs[i] != max[0]) max[1] = nbs[i];

    for(++i; i < 10; i++)    if(min[1] > nbs[i]) if(nbs[i] != min[0]) min[1] = nbs[i];

    printf("Plus grand : %i\nSecond plus grand : %i\nPlus petit : %i\nSecond plus petit : %i\n", max[0], max[1], min[0], min[1]);
}
```

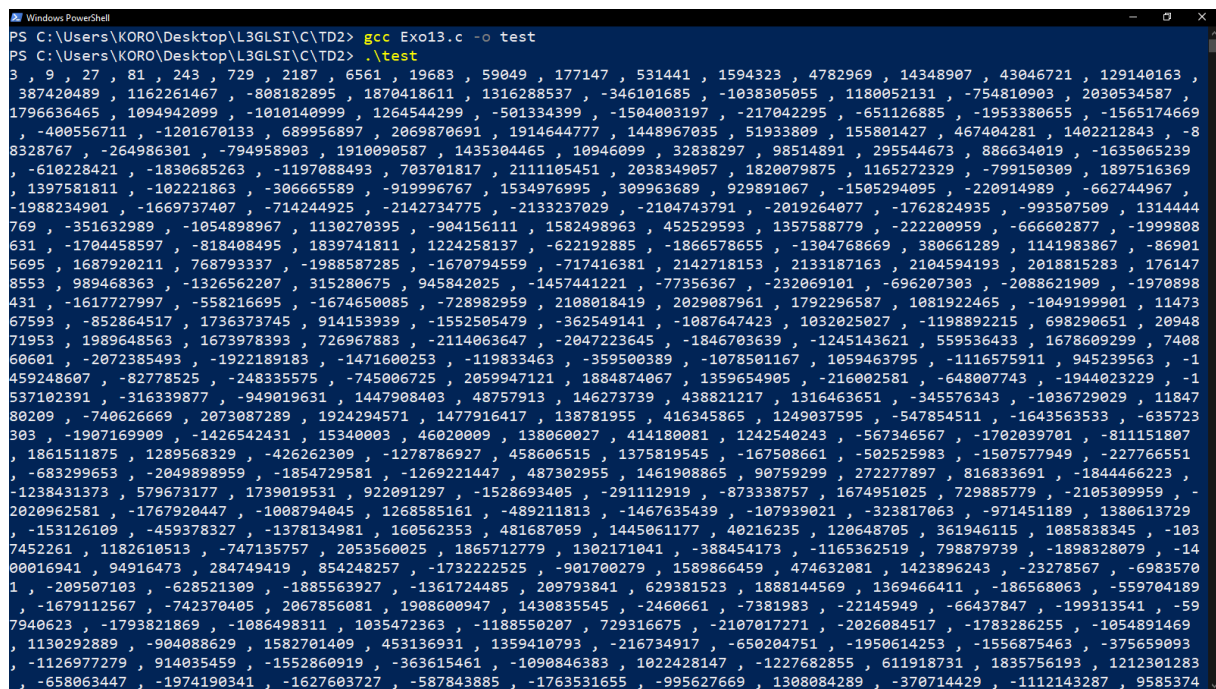
```
Windows PowerShell
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> gcc Exo7.c -o test
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
Entrez 10 entiers : 00 43 34 235 534 348 395 34 35 13
Plus grand : 534
Second plus grand : 395
Plus petit : 0
Second plus petit : 13
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
Entrez 10 entiers : 0 1 2 3 4 5 6 7 8 9
Plus grand : 9
Second plus grand : 8
Plus petit : 0
Second plus petit : 1
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2>
```

Exercise 13:

```
#include<stdio.h>

void main(){
    int puissance = 1;

    while (puissance){
        puissance *= 3;
        printf("%i , ", puissance);
    }
}
```



```
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> gcc Exo13.c -o test
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
3 , 9 , 27 , 81 , 243 , 729 , 2187 , 6561 , 19683 , 59049 , 177147 , 531441 , 1594323 , 4782969 , 14348907 , 43046721 , 129140163 ,
387420489 , 1162261467 , -808182895 , 1870418611 , 1316288537 , -346101685 , -1038305055 , 1180052131 , -754810903 , 2030534587 ,
1796636465 , 1094942099 , -1010140999 , 1264544299 , -501334399 , -1504003197 , -217042295 , -651126885 , -1953380655 , -1565174669 ,
-400556711 , -1201670133 , 689956897 , 2069870691 , 1914644777 , 1448967035 , 51933809 , 155801427 , 467404281 , 1402212843 , -8
8328767 , -264986301 , -794958903 , 1910090587 , 1435304465 , 10946099 , 32838297 , 98514891 , 295544673 , 886634019 , -1635065239 ,
-610228421 , -1830685263 , -1197088493 , 703701817 , 2111105451 , 2038349057 , 1820079875 , 1165272329 , -799150309 , 1897516369 ,
1397581811 , -102221863 , -306665589 , -919996767 , 1534976995 , 309963689 , 929891067 , -1505294095 , -220914989 , -662744967 ,
-1988234901 , -1669737407 , -714244925 , -2142734775 , -2133237029 , -2104743791 , -2019264077 , -1762824935 , -993507509 , 1314444
769 , -351632989 , -1054898967 , 1130270395 , -904156111 , 1582498963 , 452529593 , 1357588779 , -222200959 , -666602877 , -1999808
631 , -1704458597 , -818408495 , 1839741811 , 1224258137 , -622192885 , -1866578655 , -1304768669 , 380661289 , 1141983867 , -86901
5695 , 1687920211 , 768793337 , -1988587285 , -1670794559 , -717416381 , 2142718153 , 2133187163 , 2104594193 , 2018815283 , 176147
8553 , 989468363 , -1326562207 , 315280675 , 945842025 , -1457441221 , -77356367 , -232069101 , -696207303 , -2088621909 , -1970898
431 , -1617727997 , -558216695 , -1674650085 , -728982959 , 2108018419 , 2029087961 , 1792296587 , 1081922465 , -1049199901 , 11473
67593 , -852864517 , 1736373745 , 914153939 , -1552505479 , -362549141 , -1087647423 , 1032025027 , -1198892215 , 698290651 , 20948
71953 , 1989648563 , 1673978393 , 726967883 , -2114063647 , -2047223645 , -1846703639 , -1245143621 , 559536433 , 1678609299 , 7408
60601 , -2072385493 , -1922189183 , -1471600253 , -119833463 , -359500389 , -1078501167 , 1059463795 , -1116575911 , 945239563 , -1
459248607 , -82778525 , -248335575 , -745006725 , 2059947121 , 1884874067 , 1359654905 , -216002581 , -648007743 , -1944023229 , -1
537102391 , -316339877 , -949019631 , 1447908403 , 48757913 , 146273739 , 438821217 , 1316463651 , -345576343 , -1036729029 , 11847
80209 , -740626669 , 2073087289 , 1924294571 , 1477916417 , 138781955 , 416345865 , 1249037595 , -547854511 , -1643563533 , -635723
303 , -1907169909 , -1426542431 , 15340003 , 46020009 , 138060027 , 414180081 , 1242540243 , -567346567 , -1702039701 , -811151807
, 1861511875 , 1289568329 , -426262309 , -1278786927 , 458606515 , 1375819545 , -167508661 , -502525983 , -1507577949 , -227766551
, -683299653 , -2049898959 , -1854729581 , -1269221447 , 487302955 , 1461908865 , 90759299 , 272277897 , 816833691 , -1844466223 ,
-1238431373 , 579673177 , 1739019531 , 922091297 , -1528693405 , -291112919 , -873338757 , 1674951025 , 729885779 , -2105309959 , -
2020962581 , -1767920447 , -1008794045 , 1268585161 , -489211813 , -1467635439 , -107939021 , -323817063 , -971451189 , 1380613729
, -153126109 , -459378327 , -1378134981 , 160562353 , 481687059 , 1445061177 , 40216235 , 120648705 , 361946115 , 1085838345 , -103
7452261 , 1182610513 , -747135757 , 2053560025 , 1865712779 , 1302171041 , -388454173 , -1165362519 , 798879739 , -1898328079 , -14
00016941 , 94916473 , 284749419 , 854248257 , -1732222525 , -901700279 , 1589866459 , 474632081 , 1423896243 , -23278567 , -6983570
1 , -209507103 , -628521309 , -1885563927 , -1361724485 , 209793841 , 629381523 , 1888144569 , 1369466411 , -186568063 , -559704189
, -1679112567 , -742370405 , 2067856081 , 1908600947 , 1430835545 , -2460661 , -7381983 , -22145949 , -66437847 , -199313541 , -59
7940623 , -1793821869 , -1086498311 , 1035472363 , -1188550207 , 729316675 , -2107017271 , -2026084517 , -1783286255 , -1054891469
, 1130292889 , -904088629 , 1582701409 , 453136931 , 1359410793 , -216734917 , -650204751 , -1950614253 , -1556875463 , -375659093
, -1126977279 , 914035459 , -1552860919 , -363615461 , -1090846383 , 1022428147 , -1227682855 , 611918731 , 1835756193 , 1212301283
, -658063447 , -1974190341 , -1627603727 , -587843885 , -1763531655 , -995627669 , 1308084289 , -370714429 , -1112143287 , 9585374
```

Le programme produit une boucle “infinie” ne devant afficher que les puissances de 3, mais les types ayant des tailles limitées, il y a un dépassement de capacité d’où les valeurs négatives.

Exercise 14:

Alternative 1:

```
#include<stdio.h>

int crypting(int n){
    int chiffres[4], i = 4;

    for(;n;i--){
        chiffres[i-1] = n % 10;
        n /= 10;
    }

    for(; i < 4;i++)    chiffres[i] = (chiffres[i] + 7) % 10;

    for(i = 0; i < 2;i++){
        int ech = chiffres[i];
        chiffres[i] = chiffres[i+2];
        chiffres[i+2] = ech;
    }

    for(i = 0; i < 4; i++)    n = n * 10 + chiffres[i];

    return n;
}

int decrypting(int n){
    int chiffres[4], i=4;

    for(;n;i--){
        chiffres[i-1] = n % 10;
        n /= 10;
    }

    for(; i < 4;i++)    chiffres[i] += 10 - 7;

    for(i = 0; i < 2;){
        int ech = chiffres[i];
        chiffres[i] = chiffres[i+2];
        chiffres[i++ +2] = ech;
    }

    for(i = 0; i < 4; i++)    if(chiffres[i] >= 9)        chiffres[i] %= 10;

    for(i = 0; i < 4; i++)    n = n * 10 + chiffres[i];

    return n;
}
```

```

}
void main(){
    int nombre, i;

    do{
        printf("Entrez un entier de 4 chiffres : ");
        scanf("%i", &nombre);
    }while (nombre < 999 || nombre > 9999);

    i = crypting(nombre);
    printf("%i\n", i);

    i = decrypting(i);
    printf("%i\n", i);
}

```

Alternative 2:

```

#include<stdio.h>
#include<string.h>

void crypting(int * toEncrypt){
    for(int i = 3; i >= 0; i--)    toEncrypt[i] = (toEncrypt[i] + 7) % 10;

    for(int i = 3; i > 1; i--){
        int ech = toEncrypt[i];
        toEncrypt[i] = toEncrypt[i-2];
        toEncrypt[i-2] = ech;
    }
}

void decrypting(int * crypted){

    for(int i = 3; i >= 0; i--) crypted[i] += 10 - 7;

    for(int i = 0; i < 2; i++){
        int ech = crypted[i];
        crypted[i] = crypted[i+2];
        crypted[i+2] = ech;
    }

    for(int i = 0; i < 4; i++)    if(crypted[i] >= 10) crypted[i] -= 10;
}

void main(){
    char nombre[100];
    int chiffres[4], i = 0;

```

```

do{
    printf("Entrez un nombre de 4 chiffres : ");
    scanf("%s", nombre);
}while (strlen(nombre) > 4 || strlen(nombre) < 4);

for(;i < strlen(nombre);i++) chiffres[i] = (int)nombre[i] - 48;

crypting(chiffres);

for(i = 0; i < 4 ; i++) printf("%i", chiffres[i]);

printf("\n");
decrypting(chiffres);

for(i = 0; i < 4 ; i++) printf("%i", chiffres[i]);
}

```

```

Windows PowerShell
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> gcc Exo14-1.c -o test
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
Entrez un nombre de 4 chiffres : 0000
7777
0000
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
Entrez un nombre de 4 chiffres : 1234
0189
1234
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2> .\test
Entrez un nombre de 4 chiffres : 54873
Entrez un nombre de 4 chiffres : 123
Entrez un nombre de 4 chiffres : 9357
2460
9357
PS C:\Users\KORO\Desktop\L3GLSI\C\TD2>

```