**TP N°2**

**Exercice 1 (15 minutes) :** on veut contrôler un servomoteur degré par degré pour éviter de produire un choc à l’instrument relié au servo. Ecrivez un programme qui lit le degré souhaité au clavier et teste si le degré est compris entre 0-180° avant de faire fonctionner le servo (tourner). Utilisez la fonction d’affichage pour afficher chaque degré (chaque pas) au lieu de contrôler réellement le servo.



**Figure 1.** Photo réelle d’un servomoteur Towerpro SG92R.

**Exercice 2 (20 minutes) :** supposant on a un microcontrôleur dont 8 I/O ports (entrée/sortie) représentés par la variable PORTA, où tous les ports sont configurés en entrée (reçoivent des signaux externes). On veut lire l’état du troisième port (un bouton poussoir relié), et s’il est pressé (état 1) alors on allume une LED si elle est éteinte ou bien l’inverse dans le cas contraire. Pour éteindre ou allumer la LED on utilise la fonction d’affichage dédiée au C++.

***Remarque :*** *la variable PORTA doit être déclarée comme une variable global (unsigned char).*

**Exercice 3 (20 minutes) :** écrivez un programme qui représente un nombre entier sous la forme binaire en utilisant les boucles, les opérateurs binaires et la fonction d’affichage. On doit représenter le nombre dans la même ligne comme l’exemple ci-dessous.

Entrez le numéro : 398

Représentation décimale est 398

Représentation binaire est 00000000000000000000000110001110

**Exercice 4 (20 minutes) :** écrivez un programme qui remplit une matrice (tableau 2D) de taille 4x4 avec des nombres aléatoires en utilisant la fonction **rand()**. Ensuite, on extrait la diagonale et on la copie dans une autre matrice de même taille en remplissant les autres cases par des 0.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 1 | 0 | 30 |  | 3 | 0 | 0 | 0 |
| 17 | 6 | 9 | 12 | 0 | 6 | 0 | 0 |
| 29 | 7 | 13 | 63 | 0 | 0 | 13 | 0 |
| 30 | 4 | 10 | 98 | 0 | 0 | 0 | 98 |

**Exercice 5 (15 minutes) :** écrivez un programme qui remplit quatre vecteurs (tableaux 1D) de taille 4 avec des nombres aléatoires en utilisant la fonction **rand()**. Ensuite, le programme concatène les quatre vecteurs pour construire une matrice de taille 4x4. Essayez d’utiliser deux boucles au maximum pour la construction de la matrice.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **V1** | **V2** | **V3** | **V4** |  |  | | | |
| 3 | 1 | 0 | 30 |  | 3 | 1 | 0 | 30 |
| 17 | 6 | 9 | 12 | 17 | 6 | 9 | 12 |
| 29 | 7 | 13 | 63 | 29 | 7 | 13 | 63 |
| 30 | 4 | 10 | 98 | 30 | 4 | 10 | 98 |

**Exercice 6 (30 minutes) :** on veut créer un algorithme de cryptographie pour sécuriser les données transmises via le moyen radio (notamment le kit radio nRF24L01+) afin de les rendre difficiles à lire en cas d’interception du signal. On a des textes courts de 280 caractères au maximum (typiquement des messages Twitter), où pour chaque deux bytes successifs on fait un croisement d’un ensemble de bits (entre 1-7). Le résultat final est enregistré dans un tableau de 281 bytes, où le dernier byte contient le nombre de bits de croisement.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Byte 1** | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Byte 2** | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |

**Après croisement ils deviennent**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Byte 1** | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Byte 2** | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |



**Figure 2.** Photo réelle du kit radio nRF24L01+.

***Remarque :*** *chaque étudiant est noté entre A-D sur la participation et le dynamisme durant la séance du TP. Cette note sera accumulée avec l’assiduité et le test final.*