TP05 – SOUS-PROGRAMMES I

Le but des exercices de cette séance de travaux pratiques est de se familiariser avec le codage en C++ des éléments suivants :

- Déclaration et Appel de fonctions
- Définition de nouvelles fonctions

On considère que l'ouverture d'un nouveau projet dans l'environnement de développement Visual Studio Code est acquise. En cas de difficulté avec la gestion d'un projet, reprendre le TP01 à partir de l'étape 3 de la partie 1.

Les exercices peuvent être traités de façon indépendante mais il est conseillé de les aborder dans l'ordre car certaines ressources sont réutilisées. Les derniers exercices demandent plus d'initiatives.

À la fin de la séance, il faut avoir validé au moins les exercices 1 et 2.

Éteindre correctement le système cible en fin de séance

À partir du terminal, taper la commande suivante :

```
$ sudo poweroff
```

Liste des fonctions à utiliser

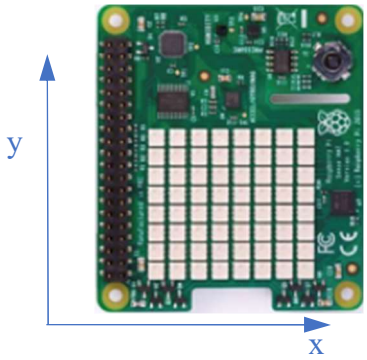
- Pour allumer le pixel aux coordonnées (x ,y) de la couleur formée par les valeurs de R, G et B :

```
senseSetRGBpixel(unsigned int x, unsigned int y, uint8_t R, uint8_t G, uint8_t B)
```

- Pour faire une temporisation de *duree* millisecondes

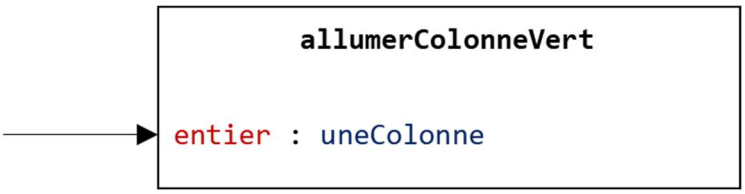
```
void sleep_for(milliseconds(duree));  // Bloque l'exécution pendant 'duree' en ms
```

Exercice 1 – Allumer une colonne de LEDs



Écrire un programme qui allume en VERT les leds d'une colonne spécifiée par son indice (0 à 7). L'indice de la colonne à allumer est fourni par l'utilisateur.

Pour ce faire, on utilisera une fonction déjà fournie `allumerColonneVert` dont la vue externe et le prototype sont les suivants :




```
FONCTION VIDE : allumerColonneVert(entier : uneColonne)
```

Étape 1 – Préparation

- Donner la vue externe du programme principal.
- Donner l'algorithme du programme principal qui demande à l'utilisateur le numéro d'une colonne à allumer en vert, puis appelle la fonction **`allumerColonneVert`** avec le bon paramètre effectif.

Étape 2 – Codage du programme

Ouvrir une session SSH sur la carte cible. À partir du terminal, lancer la commande :

```
getLab.sh tp5-ex1
```

Une fois le dossier créé, aller dans le **menu Fichier** et sélectionner **Ouvrir le dossier**. Choisir le dossier **tp5-ex1**.

Compléter le code source du fichier **tp5-ex1.cpp** :

- En ajoutant le code C++ de la fonction **`allumerColonneVert`** dans la partie Sous-Programmes dans le fichier **tp5-ex1.cpp** (Ne pas oublier de copier la constante `MAX` aussi).

```
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
// Début constantes
unsigned int MAX = 8;
```

```
// Fin constantes
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
// Début sous-programmes
void allumerColonneVert(unsigned int uneColonne)
{
      unsigned int lig;

      for (lig = 0; lig < MAX; lig = lig + 1)
      {
            senseSetRGBpixel(uneColonne, lig, 0, 255, 0);
      }
}
// Fin sous-programmes
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

- En ajoutant les déclarations de variables nécessaires dans la zone Début/Fin variable.
- En ajoutant la suite du code dans la zone Début/Fin Instructions.

Étape 3 – Test du programme et modification du code

Vérifier la conformité des tests prévus dans l'étape 1. **Faire Valider.**

Exercice 2 – Allumer une colonne de LEDs avec une couleur

Reprendre le code de l'exercice 1 pour cette fois-ci allumer une colonne avec une couleur donnée. L'indice de la colonne et la couleur seront spécifiés par l'utilisateur.

Étape 1 – Préparation

- Donner la vue externe et le prototype de la nouvelle fonction **`allumerColonneCouleur`** qui prend en entrée l'indice d'une colonne (**`uneColonne`**), ainsi que l'indice d'une couleur (**`uneCouleur`**), puis allume la colonne spécifiée avec la couleur choisie.
  - On utilisera cette correspondance pour les couleurs :
    - `1 <-> Rouge`
    - `2 <-> Vert`
    - `3 <-> Bleu`
    - `Autre valeur <-> Blanc`
- Donner la définition de la fonction **`allumerColonneCouleur`** en notation algo.
- Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

Commencer par copier la partie du code utile du fichier **tp5-ex1.cpp**.

À partir du terminal, lancer la commande :

```
getLab.sh tp5-ex2
```

Une fois le dossier créé, aller dans le **menu Fichier** et sélectionner **Ouvrir le dossier**.

Choisir le dossier **tp5-ex2**

Coller le code du presse-papier vers le fichier **tp5-ex2.cpp**

Apporter les modifications nécessaires pour répondre au nouveau besoin.

Étape 3 – Test du programme et modification du code

Vérifier la conformité des tests prévus dans l'étape 1. **Faire Valider.**

Exercice 3 – Validation des saisies

Le programme doit maintenant s'assurer que l'utilisateur entre des valeurs valides pour le choix des numéros de la colonne et de la couleur. Si l'utilisateur entre un numéro non valide (en dehors de [0-7] pour les colonnes, et en dehors de [1-3] pour les couleurs), un message d'erreur s'affiche invitant l'utilisateur à saisir une nouvelle fois le numéro attendu.

Pour ce faire, on définira deux nouvelles fonctions : **`saisieNumColonne`** et **`saisieNumCouleur`**.

Étape 1 – Préparation

- Donner la vue externe et le prototype de chaque nouvelle fonction.
- Donner la définition de chaque fonction en notation algo.
- Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

À partir du terminal, lancer la commande :

```
getLab.sh tp5-ex3
```

Editer le code source du fichier **tp5-ex3.cpp** pour répondre au besoin.

Étape 3 – Test du programme

Vérifier le fonctionnement du programme. **Faire Valider.**

Exercice 4 – Carré en couleur

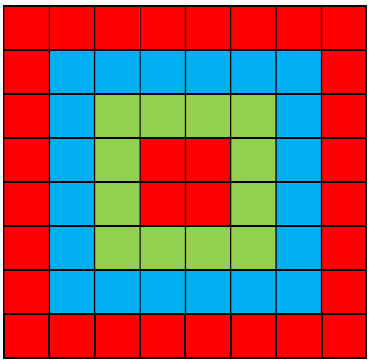
Le programme doit afficher une succession de carrés de taille différente afin d'allumer toute la matrice de leds. On alternera les couleurs des carrés (Rouge>Vert>Bleu>Rouge>…).

Pour ce faire, on propose d'adapter la fonction **`allumerColonneCouleur`** pour prendre en entrée un pixel de départ (**`unX1`**, **`unY1`**) et le nombre de pixels à allumer (**`unNbPixels1`**) avec la couleur choisie (**`uneCouleur1`**).

Dans le même style, faire une fonction **`allumerLigneCouleur`** qui prend en entrée un pixel de départ (**`unX2`**, **`unY2`**) et le nombre de pixels à allumer (**`unNbPixels2`**) avec la couleur choisie (**`uneCouleur2`**).

Ensuite, on propose de définir une nouvelle fonction **`allumerCarreCouleur`** qui, en utilisant les deux fonctions précédentes, allume des leds formant un carré dont :

- Le coin supérieur gauche est spécifié par les paramètres d'entrée **`unX`** et **`unY`**.
- La longueur du côté est **`uneLongueur`**
- La couleur est **`uneCouleur`**



Étape 1 – Préparation (optionnel)

- Donner la vue externe et le prototype de chaque fonction
- Donner la définition de chaque fonction en notation algo.
- Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

À partir du terminal, lancer la commande :

```
getLab.sh tp5-ex4
```

Editer le code source du fichier **tp5-ex4.cpp** pour répondre au besoin.

On veillera à insérer une temporisation de 300ms entre chaque affichage d'un carré.

Étape 3 – Test des sous-programmes (tests unitaires)

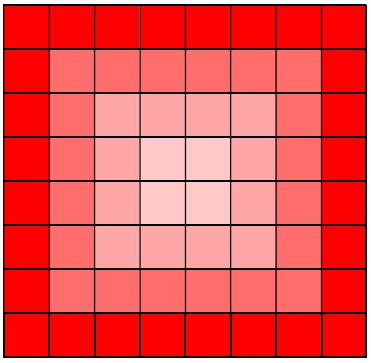
- Vérifier le fonctionnement de la fonction **`allumerColonneCouleur`**.
- Vérifier le fonctionnement de la fonction **`allumerLigneCouleur`**.
- Vérifier le fonctionnement de la fonction **`allumerCarreCouleur`**.
- **Faire Valider.**

Étape 4 – Test du programme

Vérifier le fonctionnement du programme complet. **Faire Valider.**

Étape 5 – Modification du programme

- Modifier les sous-programmes précédents de telle manière à accepter des couleurs exprimées par leurs composantes Rouge-Vert-Bleu.
  - `allumerColonneCouleurRVB`
  - `allumerLigneCouleurRVB`
  - `allumerCarreCouleurRVB`
- Proposer une modification du programme principal pour afficher des carrés successifs avec un dégradé de couleurs (cf. TP4)
- Vérifier le fonctionnement du programme complet. **Faire Valider.**



Exercice 5 – Ecran de veille

Le programme doit afficher un « écran de veille » animé à l'aide d'un affichage successif de carrés dont la taille, la couleur, et la position changent d'un affichage à un autre.
Pour ce faire, on utilisera les fonctions suivantes :

- **allumerCarreCouleurRVB** de l'exercice 4

```
void allumerCarreCouleurRVB ( unsigned int unX,
                              unsigned int unY,
                              unsigned int uneLongueur,
                              unsigned int unRouge,
                              unsigned int unVert,
                              unsigned int unBleu);
```

- Fonction de temporisation ;

```
void sleep_for(milliseconds(duree));
  // Bloque l'exécution pendant 'duree' en ms
```

- Fonction d'effacement de la matrice des leds :

```
void senseClear();  // éteint toutes les leds
```

- Fonction pour la génération de nombres aléatoires

```
int rand(void);  //génère un nombre aléatoire entre 0 et 32767
```

Exemple : Pour générer un nombre aléatoire entre 0 et 9, on exécute l'instruction suivante :

```
nombreAleatoire = rand() % 10 ;
```

Étape 1 – Préparation (optionnel)

- Rappeler la vue externe et le prototype de chaque fonction
- Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

À partir du terminal, lancer la commande :

```
getLab.sh tp5-ex5
```

Editer le code source du fichier **tp5-ex5.cpp** pour répondre au besoin.

Étape 5 – Test du programme

Vérifier le fonctionnement du programme complet. **Faire Valider.**