TP04 – BOUCLES SIMPLES ET IMBRIQUEES

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

- Les boucles simples basées sur TANT-QUE, REPETER ou POUR
- Les boucles imbriquées

Comme à chaque séance de travaux pratiques, il est nécessaire d'ouvrir un nouveau projet dans l'environnement de développement Visual Studio Code pour chaque exercice.

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 sont plus exigeants.

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

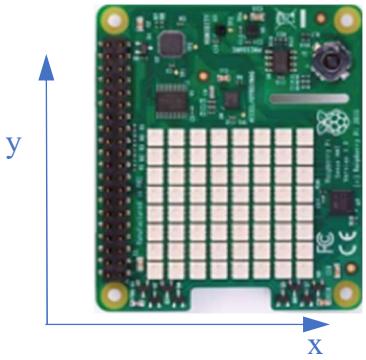
- Pour faire une temporisation de *duree* millisecondes

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

Exercice 1 – Allumer les leds colonne par colonne

Écrire un programme qui allume en VERT chaque pixel de la matrice de leds colonne par colonne (suivant l'axe des ordonnées y), en commençant par le pixel de coordonnées (0,0) (celle en bas à gauche sur l'image ci-dessous).

On utilisera la fonction de temporisation (300 ms) pour que l'on puisse observer que le remplissage se fait bien colonne par colonne.



Exemple :

Le pixel à la coordonnée (0,0) s'allume puis une temporisation de 300ms est lancée.
Le pixel à la coordonnée (0,1) s'allume puis une temporisation de 300ms est lancée.
Le processus continue jusqu'à ce que toute la matrice soit totalement allumée.

Étape 1 – Préparation

Dessiner la vue externe du programme et proposer le résultat visuel attendu.

Combien de boucles le programme devra utiliser ? Justifier.

Pour chaque boucle prévue, indiquer puis justifier la boucle à privilégier.

Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

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

```
getLab.sh  tp4-ex1
```

Une fois le dossier créé, aller dans le **menu Fichier** et sélectionner **Ouvrir le dossier**. Choisir le dossier **tp4-ex1**.

Compléter le code source du fichier **tp4-ex1.cpp** :

- En ajoutant la déclaration de la variable 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.**

Mettez en commentaire la temporisation et exécutez à nouveau le programme.

Qu'observez-vous ? pourquoi ?

Étape 4 – Modification du code et test du programme

Le programme doit exécuter la temporisation dès qu'une colonne est remplie. A quel endroit dans le programme, faut-il mettre la temporisation ?

Vérifier la conformité des tests et **faire valider**.

Exercice 2 – Allumer les leds ligne par ligne

Reprendre le code de l'exercice 1 pour cette fois-ci allumer en BLEU la matrice ligne par ligne (suivant l'axe des abscisses x). Le programme utilisera la temporisation pour permettre l'observation de l'ordre d'allumage des pixels selon la consigne.

Étape 1 – Préparation

Relativement à l'exercice 1, quels sont les changements à apporter sur la vue externe et le jeu de tests.

Quelles sont les modifications (s'il y en a) à apporter aux boucles ?

Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

Commencer par copier la partie du code utile du fichier tp4-ex1.cpp.

À partir du terminal, lancer la commande :

```
getLab.sh  tp4-ex2
```

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

Choisir le dossier **tp4-ex2**

Coller le code du presse-papier vers le fichier **tp4-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.**

Étape 4 – Modification du code et test du programme

Modifier le programme pour que l'ordre d'allumage des pixels commence maintenant à partir des coordonnées (7,0) dans le sens décroissant.

Vérifier la conformité des tests. **Faire Valider.**

Il est possible d'ajouter une variante en alternant le sens d'affichage de chaque ligne.

Exercice 3 – Dégradé de couleurs

Le programme doit allumer les pixels ligne par ligne (avec une temporisation de 300ms entre chaque ligne) avec un changement de couleur à chaque ligne.

Concernant la couleur de la ligne, on utilisera les variables **red**, **green** et **blue** que l'on initialise avec les valeurs suivantes : **31**, **127**, **255**.

Lors d'un changement de ligne, la valeur de **red** sera incrémentée de **32**, la valeur de **green** sera incrémentée de **32** et la valeur de **blue** sera décrémentée de 32.

Étape 1 – Préparation

Dessiner la vue externe du programme.

Donner le type des variables **red**, **green** et **blue** permettant de modifier les couleurs.

Quel est l'intervalle de valeurs que peut prendre ces variables ?

Indiquez la valeur finale des variables **red**, **green** et **blue** pour ce programme. La condition précédente est-elle vérifiée ?

Quelle est la valeur du pas de l'incrément d'une couleur initialisée à 0 qu'il ne faut pas dépasser pour que la valeur de la variable reste dans l'intervalle acceptable ?

Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

À partir du terminal, lancer la commande :

```
getLab.sh  tp4-ex3
```

Editer le code source du fichier **tp4-ex3.cpp** pour répondre au besoin.

Étape 3 – Test du programme

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

Exercice 4 – Alternance horizontale de couleur

Le programme doit alterner la couleur de l'allumage de chaque ligne, en commençant par la ligne du haut.

Les valeurs des variables de la couleur de la 1ère ligne sont les suivantes :

```
red=0, green= 255, blue = 125.
```

Les valeurs des variables de la couleur de la 2ème ligne sont les suivantes :

```
red=255, green= 125, blue = 0.
```

On recommence cette alternance à partir de la 3ème ligne.

Étape 1 – Préparation

Dessiner la vue externe du programme.

Comment déterminer la couleur à utiliser en fonction du numéro de ligne ?

Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

À partir du terminal, lancer la commande :

```
getLab.sh  tp4-ex3
```

Editer le code source du fichier **tp4-ex4.cpp** pour répondre au besoin.

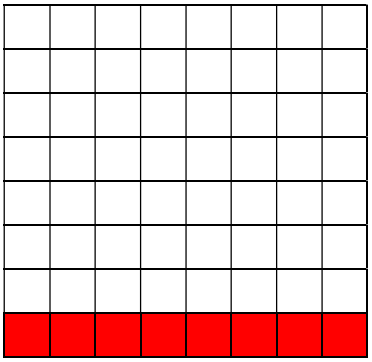
Étape 3 – Test du programme

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

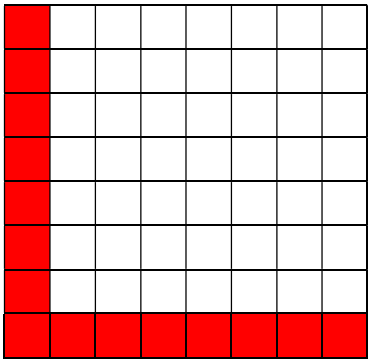
Exercice 5 – Remplissage en forme de L

On souhaite maintenant que la matrice du senseHat s'allume progressivement en forme de lettre L. Chaque L sera allumé avec une couleur différente que vous choisirez. Un exemple d'exécution du programme est donné ci-après :

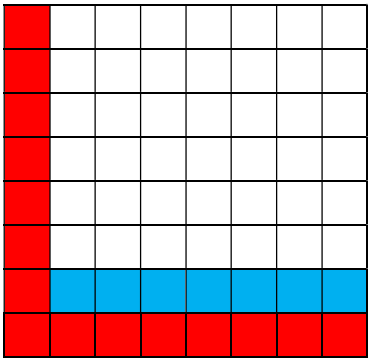
1. L'allumage commence au point (0,0)



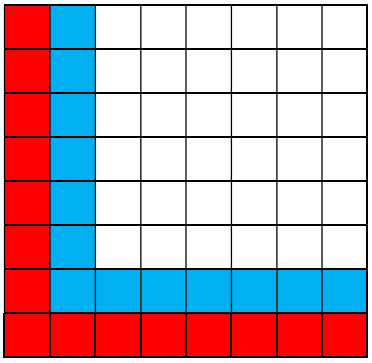
2. Une fois la ligne allumée, on allume la colonne



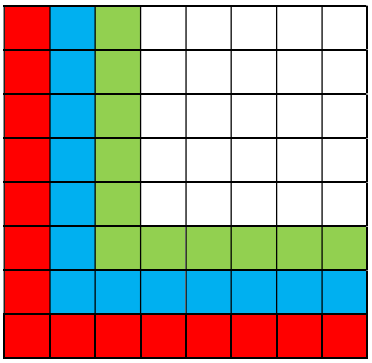
3. Au bout de 200ms, la ligne suivante s'allume avec une couleur différente



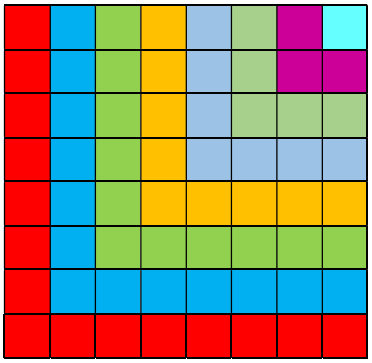
4. La colonne s'allume



5. Le processus continue avec une couleur différente à chaque



6) … jusqu'au remplissage total



Étape 1 – Préparation (Optionnel)

Etudier d'abord la variation des coordonnées pour former un L, puis, ensuite, pour enchaîner plusieurs L.

Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

À partir du terminal, lancer la commande :

```
getLab.sh  tp4-ex5
```

Editer le code source du fichier **tp3-ex5.cpp** pour répondre au besoin.

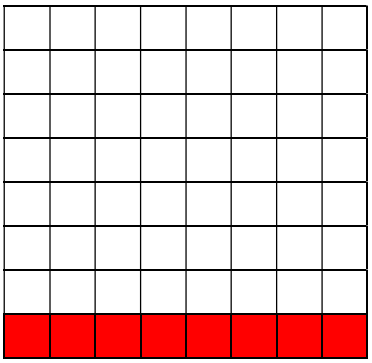
Étape 3 – Test du programme

Vérifier le fonctionnement demandé à l'étape 1. **Faire Valider.**

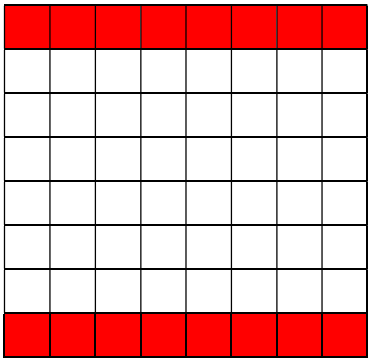
Exercice 6 – Remplissage en carré (BONUS)

On souhaite maintenant que la matrice du senseHat s'allume par des carrés successifs.

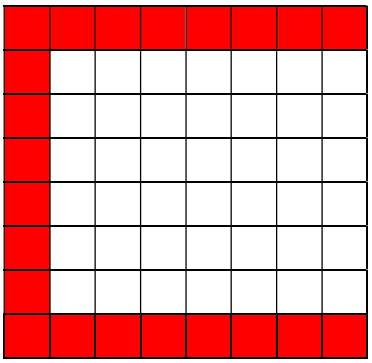
1. L'allumage commence au point (0,0)



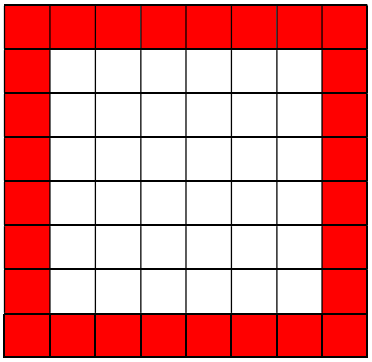
2. Une fois la ligne allumée, on allume la ligne du haut



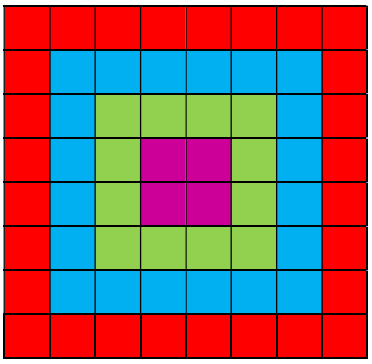
3. On allume ensuite la colonne de gauche.



4. puis celle de droite :



5. On recommence le processus jusqu'au remplissage de la matrice :



Étape 1 – Préparation (optionnel)

Comment varient les coordonnées des lignes et des colonnes,

- pour former un carré
- pour passer d'un carré à un autre.

Faire l'algorithme de ce programme.

Étape 2 – Codage du programme

À partir du terminal, lancer la commande :

```
getLab.sh  tp4-ex6
```

Editer le code source du fichier **tp4-ex6.cpp** pour répondre au besoin.

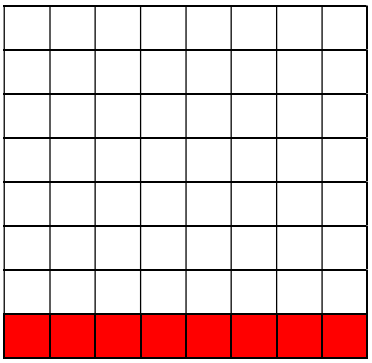
Étape 3 – Test du programme

Vérifier le fonctionnement demandé à l'étape 1. **Faire Valider.**

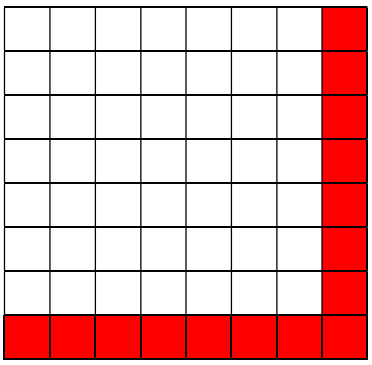
Étape 4 – Modification du code et test du programme

Modifier le programme pour qu'un carré se forme de la façon suivante :

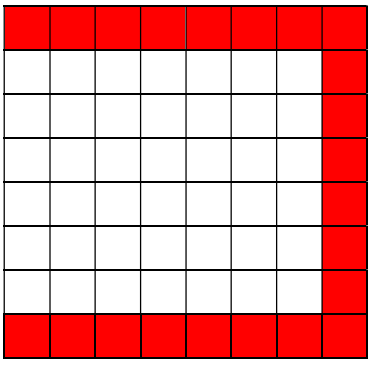
1) L'allumage commence au point (0,0)



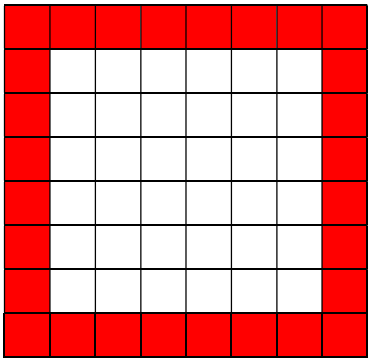
2) Une fois la ligne du bas allumée, on allume la colonne de droite



3) puis la ligne du haut.



4) et enfin la colonne de gauche :



Vérifier le fonctionnement puis **Faire Valider.**

Fin de séance

Éteindre le système cible

À partir du terminal, taper la commande suivante : `sudo  poweroff`