**Formation à l’enseignement de l’informatique au collège**

**Informatique**

**Collège**

**TP 3**

**Le jeu du Simon**

|  |  |
| --- | --- |
| Compétences |  |
|  | **Analyser le comportement attendu d’un système réel et décomposer le problème posé en sous problèmes afin de structurer un programme de commande.** |
|  | **Écrire, mettre au point (tester, corriger) et exécuter un programme commandant un système réel et vérifier le comportement attendu.** |
|  | **Écrire un programme dans lequel des actions sont déclenchées par des événements extérieurs**. |
| Connaissances associées | |
|  | Notions d’algorithme et de programme. |
|  | Notion de variable informatique |
|  | Déclenchement d’une action par un événement, séquences d’instructions, boucles instructions conditionnelles. |

|  |
| --- |
| **Objectif du TP**   * Les élèves devront réaliser en plusieurs étapes la programmation du jeu du Simon expliqué dans la suite.   **Objectif pédagogique**   * L’objectif est de découvrir la notion d’entrées et sorties tout ou rien (TOR) ou digitales ainsi que les instructions telles que les boucles et les conditions. |

# Présentation du jeu

Le Simon est un jeu permettant de tester la mémoire. Il s’agit de reproduire une séquence lumineuse sans se tromper.

Simon éclaire une des quatre couleurs des boutons rétroéclairés (rouge, bleu, jaune, vert) et produit un son toujours associé à cette couleur. Il faut alors appuyer sur la touche de la couleur qui vient de s'allumer. Le jeu répète la même couleur puis ajoute au hasard une nouvelle couleur. Il faut alors reproduire cette nouvelle séquence entièrement. Chaque fois que la séquence est reproduite correctement, Simon ajoute une nouvelle couleur. Le score maximal qu’il est possible d’atteindre est une séquence de 200.

# préparation

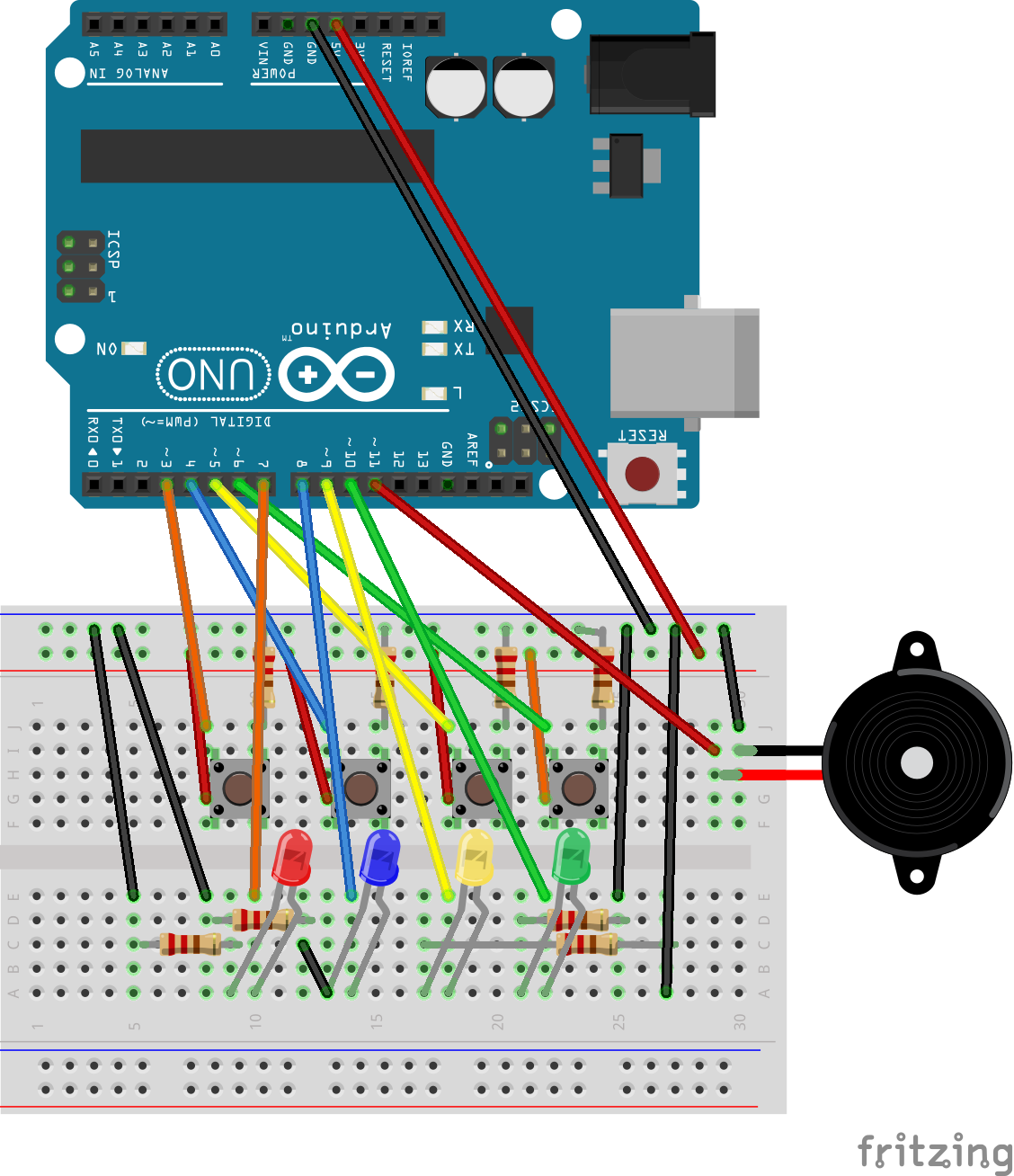
## Matériel et logiciel nécessaires

|  |  |
| --- | --- |
| Pour cette fiche, il vous faut :   * Un ordinateur avec le logiciel Mblock installé ; * Une carte Arduino Uno ; * Un câble USB ; * Une plaque d’essai ; * 4 boutons poussoirs et 4 résistances de 1K ; * 4 leds (une jaune, une rouge, une bleue, une verte) ; * 4 résistances de 200 Ohms ; * Un buzzer ; * Des straps mâles-mâles. |  |

## Câblage

Les leds, boutons poussoir et buzzer sont directement montés sur la plaque d’essai à l’aide des résistances.

Le câblage suivant indique les connections à réaliser.



Le tableau suivant recense les pins digitaux associés aux leds et boutons poussoirs.

|  |  |
| --- | --- |
| Pins Digitaux Arduino | Type |
| 3 | Bouton rouge |
| 4 | Bouton bleu |
| 5 | Bouton jaune |
| 6 | Bouton vert |
| 7 | Led rouge |
| 8 | Led bleue |
| 9 | Led jaune |
| 10 | Led verte |
| 11 | Buzzer |

## Préparation de mBlock

* Brancher la carte Arduino par l’intermédiaire du câble USB.
* Lancer le logiciel mBlock.
* Dans le menu Connecter, choisir Port Série et sélectionner le COM disponible.
* Cliquer ensuite dans Connecter sur Mettre à jour le microprogramme pour travailler directement en liaison avec la carte Arduino (mode On line).
* À la fin du téléversement, cliquer sur Fermer.

# Activité 1 – Découverte des sorties digitales

|  |  |  |
| --- | --- | --- |
| **Activité:**   |  |  | | --- | --- | | * Créer le programme suivant en faisant glisser depuis Pilotage les blocs Arduino (verts) et depuis Contrôle ou Événements les autres blocs. * Cliquer sur  dans la scène pour lancer le programme en mode On-line. Le stopper en cliquant sur .   + Quel est le but du programme ?   + À quoi correspondent le niveau haut et le niveau bas ?   + Comment appelle-t-on ce type d’informations ? * Modifier le programme pour faire clignoter plus rapidement la led. |  | |

|  |
| --- |
| **Activité :**   * Ajouter de nouvelles instructions pour faire clignoter les 4 leds en même temps. |

|  |
| --- |
| **Activité :**   * Modifier les instructions pour que les leds s’allument successivement (rouge puis bleu puis jaune puis vert). On réalise ainsi un chenillard. |

|  |
| --- |
| **Activité :**   * Modifier une instruction pour que le chenillard n’exécute que trois fois la suite de couleurs |

# Activité 2 – Découverte des entrées digitales

|  |  |  |
| --- | --- | --- |
| **Activité:**   |  |  | | --- | --- | | * Réaliser le programme suivant en faisant glisser depuis Pilotage les blocs Arduino (verts), depuis Apparence le bloc dire Bonjour (violet) et depuis Contrôle ou Événements les autres blocs. * Cliquer sur  dans la scène pour lancer le programme en mode On-line. Le stopper en cliquant sur . Appuyer sur le bouton rouge puis le relâcher.   + Qu’observe-t-on ?   + Quelles sont les valeurs disponibles lors de l’appui ou non sur le bouton poussoir. * Comment appelle-t-on ce type d’information ? |  | |

Nous allons maintenant réaliser un programme permettant de contrôler l’allumage des leds en fonction de l’appui sur le bouton poussoir.

|  |  |  |
| --- | --- | --- |
| **Activité :**   |  |  | | --- | --- | | * Modifier le programme pour prendre en compte une instruction de type Condition (dans Contrôle). * Tester le programme. * Modifier le programme pour faire la même chose sur chaque led de couleur (l’appui sur le bouton rouge allume la led rouge, l’appui sur le bouton jaune allume la led jaune). |  | |

## Le petit plus

|  |  |  |
| --- | --- | --- |
| **Activité :**   |  |  | | --- | --- | | * Ajouter la commande de Pilotage pour jouer un son sur le buzzer branché sur la broche 11. Tester différentes notes ou différentes fréquences. |  | |

# Activité 3 – Gestion de différents modes – fonctions

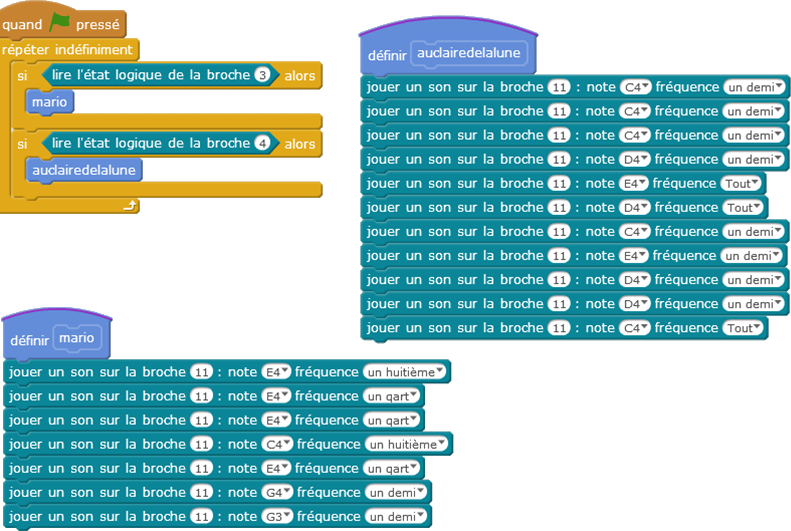
Dans le jeu du Simon, il est possible de sélectionner différents modes de jeu (jeu à deux, répétition d’une suite inventée par le joueur, jeu de rapidité…). Nous allons voir que ce type de comportement nécessite de faire des tests et gérer les événements en fonction de conditions.

|  |  |  |
| --- | --- | --- |
| **Activité :**   |  |  | | --- | --- | | * Créer un programme qui fasse clignoter toutes les leds en même temps. Tester votre programme. |  | |

|  |  |  |
| --- | --- | --- |
| **Activité :**   |  |  | | --- | --- | | * Cliquer dans Blocs et variables sur Créer un bloc. Renseigner le nom blink (clignotement). Un bloc apparaît dans la fenêtre de programmation. Il s'agit de la définition d’un sous-programme ou fonction. * Prendre toutes les instructions mises dans répéter indéfiniment et les faire glisser dans le bloc Définir blink. * Dans répêter indéfiniment faire glisser depuis Blocs le bloc blink. Tester le comportement. |  | |

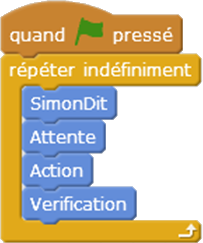
Par cette technique, il est possible de faire plusieurs fonctions et de définir un programme principal qui utilisera ces fonctions.

|  |
| --- |
| **Activité :**   * Créer de 2 à 4 fonctions différentes que vous nommerez comme vous voulez en utilisant les leds ou le buzzer (on peut créer ainsi 4 musiques !). * Dans le programme principal, introduire des conditions en fonction de l’appui sur les boutons pour appeler une des fonctions créées. |



# Activité 4 – Gestion du jeu SIMON

Le programme principal du jeu est défini de la manière suivante.



## Fonction SimonDit

La fonction SimonDit est proposée ci-contre Une variable N est introduite.

|  |  |  |
| --- | --- | --- |
| **Activité :**   |  |  | | --- | --- | | * Réaliser ou ouvrir le fichier JeuSimon   + Quelle valeur peut prendre N au cours de l’exécution du programme.   + Que se passe-t-il si N est égale à 2. |  | |

## Fonction Attente

La fonction Attente permet d’attendre que le joueur appuie sur une des 4 touches.

## Fonction Action

Ensuite la fonction Action est appelée dès qu’une touche est appuyée. Une nouvelle variable APPUI est utilisée.

Lorsqu’une touche est appuyée, la broche Entrée logique correspondante est mise au niveau logique Bas. A l’inverse lorsqu’elle n’est pas appuyée, cette entrée vaut Haut.

|  |
| --- |
| **Activité :**   * Réaliser cette fonction ou ouvrir le fichier JeuSimon   + Indiquer ce qu’il se passe lorsque le bouton rouge est appuyé. |

## Fonction Vérification

Le jeu procède ensuite à une vérification pour savoir si la touche appuyée correspond bien à ce qui était demandé. Si la touche ne correspond pas, toutes les leds s’allument une seconde puis elles s’éteignent pendant une seconde. Si la touche correspond, les leds s’éteignent toutes et la procédure complète peut redémarrer.

|  |  |  |
| --- | --- | --- |
| **Activité :**   |  |  | | --- | --- | | * Ouvrir le fichier JeuSimon   + Compléter le sous-programme Verification en renseignant les éléments vides pour que la spécification précédente soit respectée.   + Tester votre programme |  | |

|  |
| --- |
| **Activité :**   * En conservant les mêmes fonctions, que faut-il modifier pour respecter le jeu complet du Simon, c’est-à-dire répéter une séquence augmentant progressivement. * On pourra réaliser ce programme en introduisant une variable de type Liste dans laquelle on stockera les valeurs à jouer. |