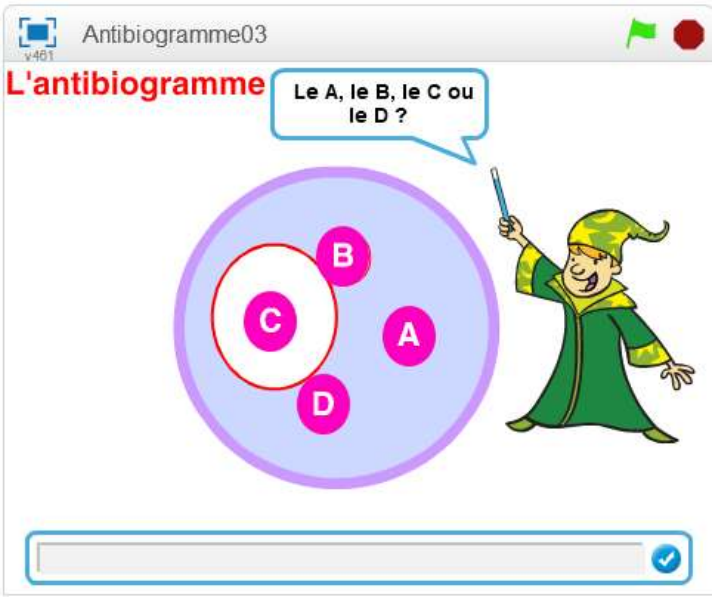
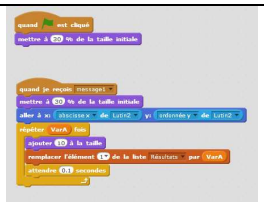
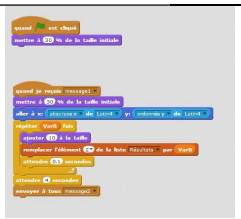
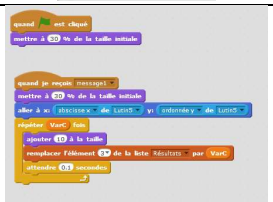
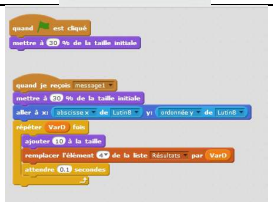
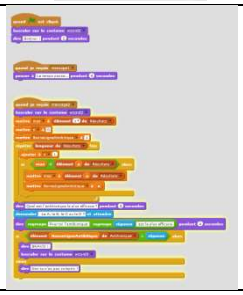
Analyse du code pour le projet scratch : « antibiogramme »

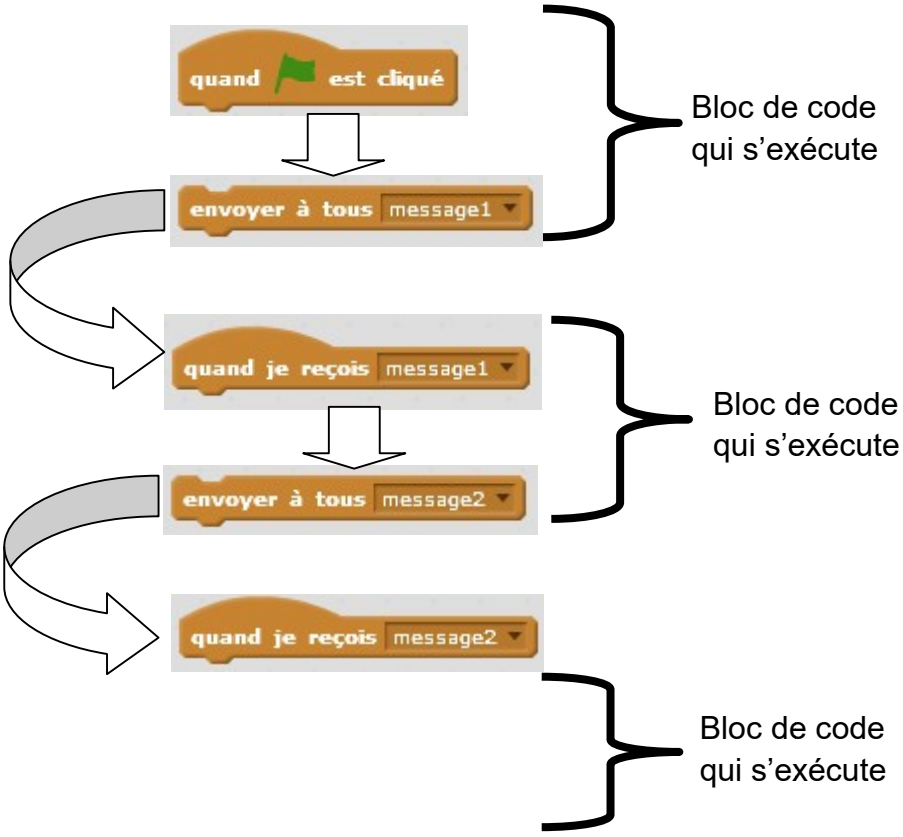


https://scratch.mit.edu/projects/270819514/

1. Tous les codes de l'application en un coup d'œil :

Arrière plan	Lutin 1	Lutin 2	Lutin 3	Lutin 4
				
Lutin 5	**Lutin 6**	**Lutin 7**	**Lutin 8**	**Wizard2**
				

2. Logique générale du programme :

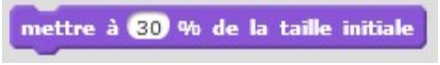


Une information importante à prendre en compte dans scratch, le code contenu dans un « lutin » ou dans « l'arrière plan », ne peut pas contenir « un bout de programme » qui modifie l'action ou le fonctionnement d'un autre lutin autre que lui-même. Pour palier à ce « problème », il faut envoyer un message au lutin que l'on souhaite « contrôler ». Dans ce projet, j'ai laissé « message1 » et « message2 », mais on peut renommer les messages et être plus précis, afin de rendre le code encore plus compréhensible.

3. Décryptage étape par étapes du code de ce projet scratch

Le programme commence quand on clique sur le drapeau vert en haut à droite de la fenêtre.
Généralement, on met cette commande dans le « script » de tous les lutins et parfois dans l'arrière plan comme ici. D'une manière générale cela nous permet de dire si le lutin est « visible » ou « caché », sa position « X » et « Y », si on veut mettre les lutins au « premier plan » par rapport à d'autres lutins…

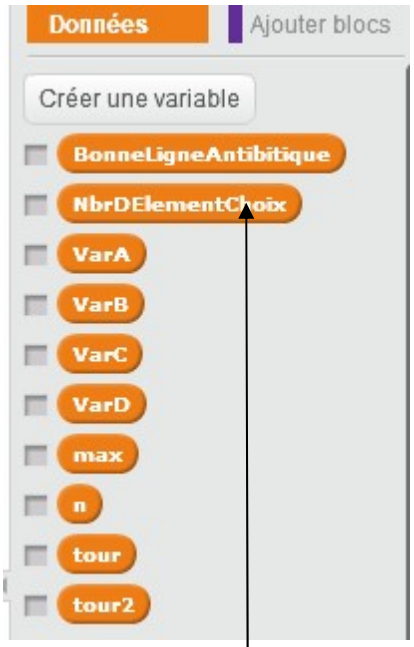
Dans ce projet :



- Pour les lutins 1, 3, 6 et 7 (« colonie sans bactérie »), on veut qu'à chaque « lancement » de l'application, qu'ils aient une taille à 30% de la taille réelle (donc qu'ils soient petits, plus petits que les pastilles des antibiotiques).

- Pour les <u>lutins 2, 4, 5 et 8</u> (« pastille d'antibiotique ») : on veut qu'à chaque « lancement » qu'elles soient au premier plan, surtout devant les « petites colonies sans bactéries »
- Pour le <u>lutin « Wizard2 »,</u> qui joue le rôle du professeur, on veut qu'il commence avec le costume n°1 (vert), puisque plus tard dans l'animation, il passera sur le costume 2 (marron/violet) si la réponse est bonne. On veut également que le professeur souhaite « bonjour » à ses élèves pendant 2 secondes.

- Pour l'arrière plan, c'est plus compliqué parce que c'est la que sont définies et « réinitialisées » les variables et les listes.
  - <u>Variables :</u> ce sont des données qui sont amenées à changer au cours du déroulement et l'exécution du code de l'application. Elles peuvent être sous forme de nombre, de texte, vrai/faux (choix binaire)…
    - Dans ce projet, j'ai créé 10 variables, qui sont des « nombres », je vais les présenter dans l'ordre de « variation (modification)» du programme, pour l'instant vous ne comprendrez peut-être pas totalement leur rôle :
      - **NombreDElementChoix** : cette variable au départ est définie sur « **20** », elle sert à créer ou à supprimer les lignes de la liste nommée « ListeNombre » (voir rubrique juste après) grâce à une boucle « répéter x fois ». Dans un premier temps, on « supprime » toutes les lignes de la liste.



      - **tour** : est une variable nombre, définit sur « 1 » est qui reste sur « 1 » tout le long du code, dans ce cas j'aurais pu ne pas utiliser une variable est écrire directement 1, mais parfois, dans le « programme » un texte (équivalent ici au nombre 1) est plus parlant et permet de mieux comprendre le code.

Donc ici, cela supprime la ligne 1 de notre liste 20 fois de suite, sachant qu'il y avait 20 lignes, il ne reste plus de ligne dans notre liste. Cette ligne de code à pour but de réinitialiser la liste et repartir grâce au code suivante sur une nouvelle liste, qui permettra d'obtenir à la fin de l'animation des résultats différents.

- **tour2** : cette variable débute à 1 au début du code, elle va être utilisée dans le code suivant pour recréer les 20 lignes supprimées juste avant dans la liste « ListeNombre »

Ici on sincère **1** (=tour2) x **1** (tour1) = **1** en position **1** (=tour2) de la liste « liste de nombre », puis à notre variable « tour2 », on ajoute 1, donc « tour2 » devient égal à **2**.
On relance la boucle, on sincère **2** (=tour2) x 1 (tour1) = **2** en position **2** (=tour2) de la liste « liste de nombre », puis à notre variable « tour2 », on ajoute 1, donc « tour2 » devient égal à **3**. Ainsi de suite.
Résultat :

N°	Valeur
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8

9	9
10	10
11	11
12	12
13	13
14	14
15	15
16	16

17	17
18	18
19	19
20	20

- **VarA** : elle définit sur 1 au départ. L'objectif est de lui attribuer une valeur au hasard entre 1 et 20, qui sera utilisée plus tard dans le code pour définir la taille de la partie sans bactérie (lutin1), mais cette valeur ne devra pas être attribuée à Var B, Vac C et Var D, afin qu'on puisse ne trouver qu'un seul antibiotique « le plus efficace » à la fin. Cette contrainte se règle par l'utilisation de la liste. On va donc pendre au hasard une des lignes de la liste qui contient un chiffre entre 1 et 20, et ensuite on le retire de la liste, ce qui fait qu'il ne reste plus que 19 lignes à notre liste.

Voici comment on retire la bonne ligne avec le code suivant

- **VarB** :

On fait la même chose, mais on choisi un chiffre entre 1 et 19, puisqu'il y a une ligne de moins dans notre liste.

- **VarC** :

On fait la même chose, mais on choisi un chiffre entre 1 et 18, puisqu'il y a 2 lignes de moins dans notre liste.

- **Var D** :

On fait la même chose, mais on choisi un chiffre entre 1 et 17, puisqu'il y a 3 lignes de moins dans notre liste.

- Remarque : je viens de découvrir qu'il y avait plus simple, on peut choisir dans élément de la liste, la valeur « hasard ». ce qui signifie qu'on aurait pu éviter les parties vertes dans le code ci-dessus, mais je l'ai laissé pour qu'on comprenne bien la suppression des lignes de la liste

- Je parlerai des trois dernières variables dans la suite de l'explication du code.

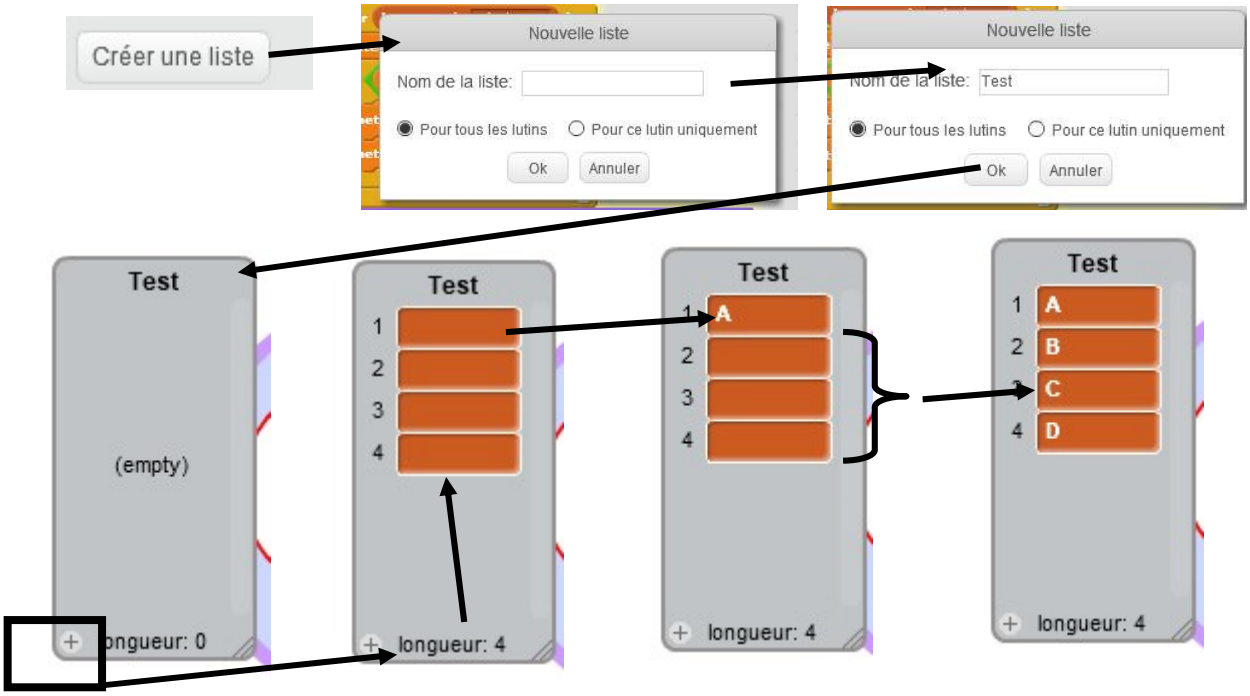
- Les <u>listes :</u> ce sont des données qui sont rangées dans un tableau à une colonne et contenant autant de ligne que nécessaire. L'intérêt des listes, c'est de pouvoir retrouver la donnée nécessaire en demandant le numéro de la ligne de cette liste.
  - Exemple d'une liste :

N°	Valeur
1	12
2	Vrai
3	Faux
4	Toto
5	On
6	Off

- Dans ce projet il y a trois listes :
  - **« ListeNombre »** : elle a été présentée ci-dessus, elle permet de stocker des nombres de 1 à 20, en attribuer un à la VarA et ensuite qu'ils ne soient pas réattribués à une autre variable(VarB, VarC et VarD) en les retirant de la liste.

  - **Résultats** : c'est une liste qui va contenir les 4 valeurs extraites de la liste « ListeNombre ». Cette liste sera expliquée plus tard.

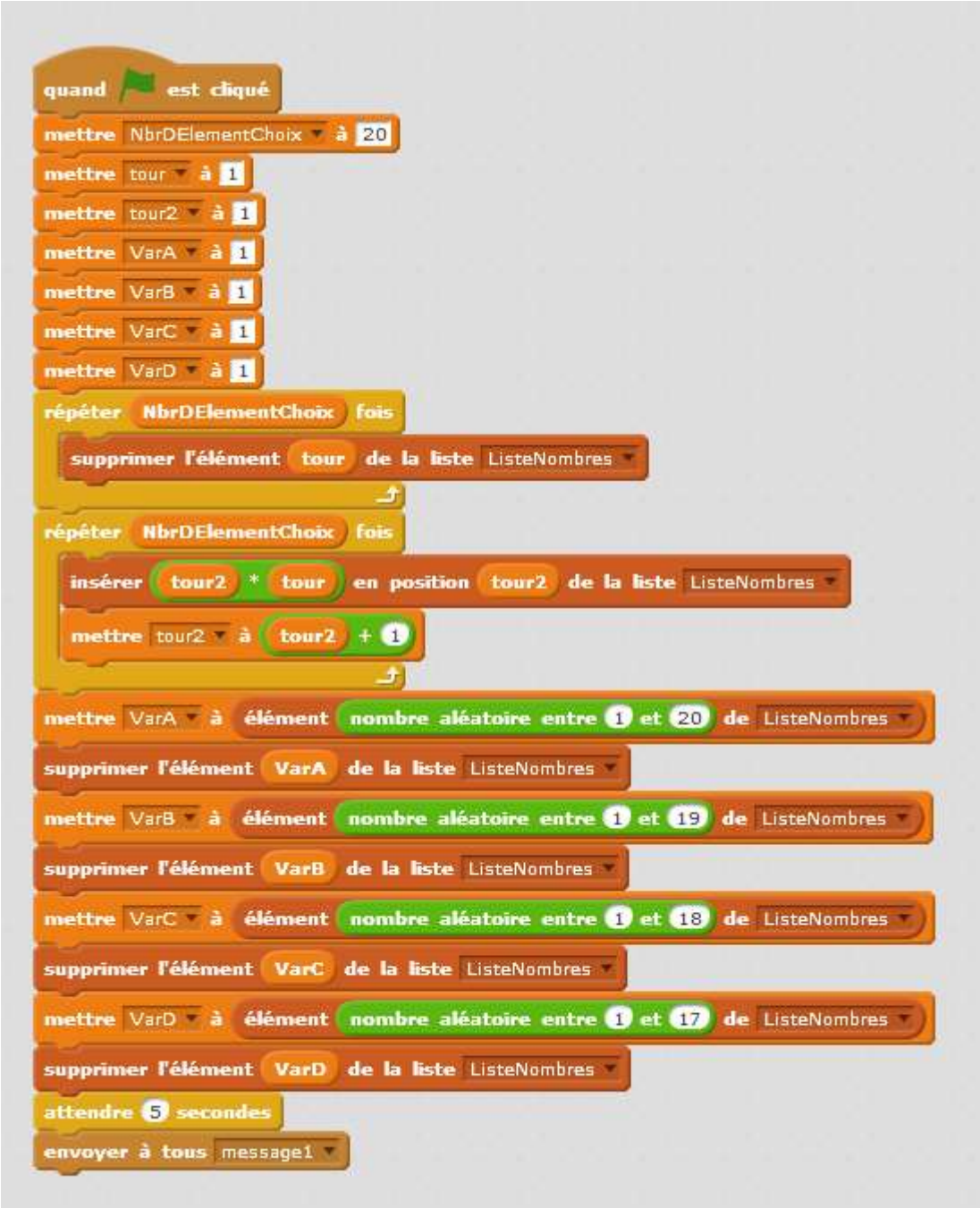
  - **Antibiotique** : c'est une liste qui a été construite à la main, elle contient la lettre A en ligne 1, la lettre B en ligne 2, la lettre C en ligne 3 et la lettre D en ligne 4. On la construit de la manière suivante :



On construit une liste en cliquant sur le bouton « créer une liste », ensuite on peut créer les lignes de cette liste à la main ou via le code.

- <u>Explication complète du code lancée avec le drapeau</u> : dans un premier temps, on définit la valeur des variables, puis on supprime toutes les valeurs de la liste « ListeNombre », puisque si l'application a déjà été lancée une fois il ne reste que 16 choix. Ensuite on reforme la liste complète « ListeNombre » pour qu'elle contienne les 20 valeurs. On choisit au hasard une parmi les 20 valeurs qu'on attribue à VarA, puis on choisit une des 19 valeurs restantes qu'on attribue à VarB, puis on choisit une des 18 valeurs restantes que l'on attribue à VaC, puis on choisit une des 17 valeurs restantes que l'on attribue à VarD. On attend 5 secondes et on envoie le « message1 » à tous les lutins.

Voici le code complet de l'arrière plan se lançant avec le drapeau vert :



Deuxième partie du code :

Dans mon programme, les lutins 1, 3, 6, 7 et wizard2 sont « sensibles » à l'envoi de ce message.

- **Wizard2 :**

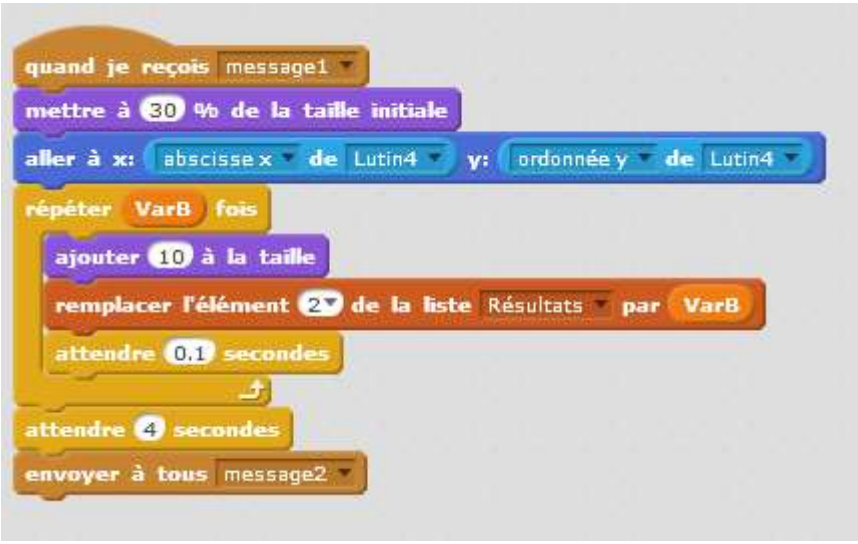
Le magicien pense pendant 4 secondes « Le temps passe... »

- **Lutin 1** :

Mettre à 30% pourrait être enlevé, puisque on appuie sur le drapeau vert qui met lui aussi à 30% la taille. Cette ligne est en prévision d'une amélioration du programme avec l'ajout d'un bouton pour réinitialiser l'application sans utiliser le drapeau vert.
La deuxième ligne permet d'aligner la pastille d'antibiotique A (Lutin2) avec la disparition des colonies qui est plus petite (30%) et caché dessous puisque le lutin 2 est au premier plan.
La troisième ligne correspond à une boucle qui sera répétée autant que le nombre attribué à VarA qui a été tiré au sort dans le tableau.

- Si VarA=1 alors la boucle ne se fait qu'une fois et la taille de la colonie n'augmente que de 10%, elle passe à 40% de sa taille réelle, ce qui fait qu'elle est plus petite que la pastille antibiotique (lutin2), c'est comme si l'antibiotique n'avait pas agit.
- Si VarA = 20 alors la boucle se fait 20 fois et la taille de la colonie augmente de 200%, soit 230% de la taille réelle ; elle est donc beaucoup plus grosse que la pastille « antibiotiqueA » (lutin2).
- La ligne « remplace l'élément » de la ligne n°1 de la liste « Résultats » par la valeur de « VarA ». Cette ligne de code doit pouvoir se mettre à l'extérieur de la boucle, tout à la fin de ce code, en effet il semble inutile de remplacer parfois 20 fois la valeur par VarA qui n'est pas modifiée entre temps ➔ a vous de tester si ca marche aussi !

- **Lutin 3** :

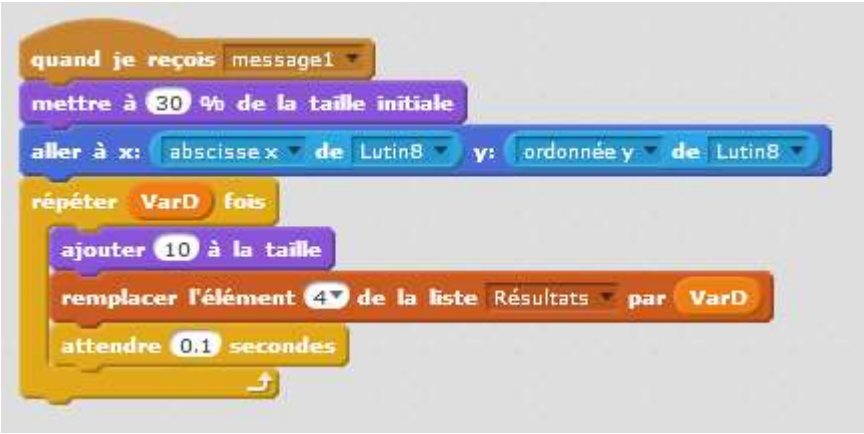


C'est le même code, mais avec VarC, et on envoie en plus le message 2 pour lancer la troisième phase du programme. Cette dernière ligne de code aurait pu être mise sur les lutins 1, 6 ou 7.

- **Lutin 6** :

Même code avec VarC

- **Lutin 7** :



Même code avec VarD

Troisième partie du code :

Il n'y a que « Wizard2 » qui est concerné par la réception de ce message2

Dans ce programme, on va attribuer des valeurs aux trois variables dont on n'a pas encore parlé :

- **max**
- **n**
- **BonneLigneAntibiotique**

On attribut à max la valeur de ligne 1 de notre liste résultat qui a été construite avec les 4 valeurs de VarA, VarB, VarC et VarD, cela correspond donc à la même valeur que VarA puisque c'est cette valeur qui est dans la première ligne de la liste.
On met n à 0
On met BonneLigneAntibiotique à 1 : cette variable, par l'exécution du programme suivant va avoir une valeur qui correspond soit à 1, 2, 3 ou 4, ce qui correspondra à la valeur la plus élevée de la liste « Résultat ».

Dans cette partie du code on réalise une boucle de la taille de la liste « Résultats », soit = 4, donc elle sera répétée 4 fois.
n passe de 0 à 1 puisqu'on luis ajoute 1.

Pour comprendre la **condition si** de ce code on va prendre un exemple :

Résultat	
1	**15**
2	**6**
3	**9**
4	**18**

Max = ligne 1 du tableau, donc 15
Si 15 strictement inférieur à « ligne 1 (c'est la valeur de n) de liste « Résultat » » **alors** on fait la suite du code. Dans ce cas, 15 (=max) n'est pas strictement inférieur à 15, on ne remplit donc pas la condition.
La boucle s'exécute pour la 2ème fois, n passe à 2. **Si** 15 (=max) strictement inférieur à ligne 2 du tableau, ici 6 **alors**, mais ici ce n'est pas le cas donc on n'applique pas la condition.
La boucle s'exécute pour la 3ème fois, n passe à 3. **Si** 15 (=max) strictement inférieur à ligne 3 du tableau, ici 9 **alors**, mais ici ce n'est pas le cas donc on n'applique pas la condition.
La boucle s'exécute pour la 4ème fois, n passe à 4. **Si** 15 (=max) strictement inférieur à ligne 4 du tableau, ici 18 **alors**, ici c'est le cas, donc on applique la condition.

Vu que la condition s'applique, on attribue à max, la ligne 4 de la liste « Résultats », max a donc la valeur 18 qui lui est attribuée à partir de maintenant.
Ensuite on attribue à la variable « BonneLigneAntibiotique » la valeur n de cette 4ème boucle soit 4. La variable « BonneLigneAntibiotique » = 4

Ensuite on fait parler le magicien qui joue le rôle du professeur :

Il pose la question quel est l'antibiotique le plus efficace pendant 3 seconde.
« demander « Le A, le B, le C ou le D ? et attendre » ». Une boite de dialogue s'ouvre en bas

Elle permet à l'utilisateur d'entrer la bonne réponse par rapport à ce qu'il voit sur l'antibiogramme. Ici D.
Le magicien dit « pour toi l'antibiotique D (➔code « réponse » bleu) est le plus efficace, pendant 4 secondes.

Enfin, il faut vérifier si la réponse est bonne

On avait créé une liste à la main nommée « Antibiotique »

Antibiotiques	
1	A
2	B
3	C
4	D

Si la ligne (« élément ») correspondant à la variable « BonneLigneAntibiotique » déterminée juste avant, correspond (dans notre exemple à la ligne 4 de la liste antibiotique) = réponse à la question que l'on a entrée dans la boite de dialogue, ici D donc D = D alors on dit bravo et le magicien passe sur le costume marron avec le chapeau violet.
Sinon (D = C, ou D = A, ou D = B) on dit à l'élève qu'il n'a pas compris.

Voila la description du fonctionnement de ce programme scratch est terminé.
Il permet de voir de nombreuses facettes de la programmation :

- **variables,**
- **listes,**
- **questions réponses,**
- **message,**
- **costumes,**
- **dialogue d'un personnage,**
- **boucles,**
- **Si alors,**
- **Si alors Sinon**
- **Coordonnée X et Y**
- **…**