Projet Cryptographie de César

On appelle cryptographie les méthodes pour coder les messages.

Le **chiffre de César** (ou *chiffrement par décalage*) est un algorithme de chiffrement très simple que Jules César utilisait pour chiffrer certains messages qu'il envoyait.

Un message crypté par cette méthode débutait par un nombre puis le message dont les lettres avaient été décalées de ce nombre de rang dans l'ordre alphabétique.

Chiffrement

Par exemple, le message : « 4 N'EHSVI PIW QEXLW» signifie

« J'ADORE LES MATHS »

Le 4 signifie que l'on a décalé les lettres de quatre rangs et donc un A est transformé en D, un B en E, etc.

À chaque lettre de l'alphabet est donc associé un nombre qui correspond à son rang dans l'alphabet.

A → 0, B → 1, …Z→ 25

Pour crypter notre message, nous devons ajouter au rang de la lettre que l'on crypte, la valeur de la clé de cryptage (le décalage à effectuer).

Par exemple, pour coder la première lettre de l'expression « J'adore les maths ! », avec une clé de 4, on fera : rang J = 9

Cryptage : (9 + 4) = 13 → lettre N

Le J sera donc codé par un N.

Pour le A → (0 + 4) = 4 → lettre E

Et si on a un Z à crypter, que fait-on ?

Rang de Z = 25

Cryptage : (25 + 4) = 29

Nous dépassons le nombre de lettre dans l'alphabet.

De Z, il faut reboucler au début de l'alphabet.

Z est alors crypté par D qui correspond à la lettre située à 4 rangs de Z.

Pour assurer ce bouclage de la fin de l'alphabet à son début, il suffit de faire : (**rang de la lettre à crypter** + **clé de cryptage**) modulo 26

Ainsi le Z, sera crypté avec notre clé de 4, par :

(25 + 4) modulo 26 = 29 modulo 26 = 3 → D

Et si le décalage, m'amène sur la lettre Z, cela fonctionne-t-il ?

Avec un décalage de 4, la lettre V doit être cryptée par un Z.

Lettre V → (21 + 4) modulo 26 = 25 modulo 26 = 25 → Z

Le modulo revient à **soustraire** 26 à l'addition, lorsque le résultat de cette addition est supérieur à 26.

Déchiffrement

Le déchiffrement fonctionne sur le même principe que le chiffrement.

On regarde quel est le rang dans l'alphabet de la lettre cryptée, et on la décrypte en prenant dans l'alphabet la lettre située « clé » rang **avant**.

Par exemple, avec le message : « 4 N'EHSVI PIW QEXLW»

Rang de N = 13 → (13 – 4) = 9 → J

Comment décoder un A ?

Rang de A = 0 → (0 – 4) ?

Un A doit être décodé par un W, quatrième lettre à la fin de l'alphabet.

Là aussi, il faut boucler le début de l'alphabet avec la fin de l'alphabet.

Pour assurer ce bouclage il suffit de faire :

(**rang de la lettre à crypter** - **clé de cryptage**) modulo 26

A → (0 – 4) modulo 26 = 22 → W

B → (1 – 4) modulo 26 = 23 → X

C → (2 – 4) modulo 26 = 24 → Y

D → (3 – 4) modulo 26 = 25 → Z

E → (4 – 4) modulo 26 = 0 → A

Le modulo revient à **ajouter** 26 à la soustraction, lorsque le résultat de cette soustraction est négatif.

Programmation : Version simple

- On utilisera un seul lutin.
- Une liste « **abc** » contenant les lettres de l'alphabet.
- Une variable **globale** « **clef** » qui contiendra la clef de codage.
- Une variable « **phraseCodée** » **globale** qui contiendra la phrase une fois codée. Vous afficherez cette variable sur la scène, en mode « grande lecture »
- Une variable « **phrase** » **locale** au lutin qui contiendra la phrase à coder.
- Une variable « **phraseDécodée** » pour l'instant globale**.**
- Les autres variables utiles peuvent être globales.
- Une procédure « **TrouverRang** » qui, reçoit en paramètre une lettre et rend le numéro de cette lettre dans l'alphabet.

- Une procédure « **Codage** » qui aura deux paramètres : la phrase à coder et la clé. C'est cette procédure qui remplit la variable « **PhraseCodée** »
- Une procédure « **Décodage** » qui aura deux paramètres : la phrase à décoder et la clé. Cette procédure rempli la variable « **PhraseDécodée** ». Cette procédure étant quasi identique à la procédure « Codage » vous avez intérêt à dupliquer les instructions situées sous le chapeau « Codage », à créer un nouveau bloc « Décodage », à y accrocher les instructions dupliquées et à modifier les instructions, quand cela est nécessaire.

Attention

Dans Scratch les caractères dans un mot sont numérotés à partir de 1. Il faudra donc diminuer le rang de 1 lorsqu'on a trouvé la lettre, dans la procédure « TrouverRang ».

Par exemple, si l'on doit coder un A, dans la liste « abc » A est au rang 1, mais pour que notre algorithme fonctionne A doit avoir le rang 0.

Lorsqu'on recherchera la lettre de remplacement, dans la liste « abc », il faudra ajouter 1 au résultat de (rang + clef) modulo 26, pour tomber sur la bonne lettre.

Par exemple, si le résultat de (rang + clef) modulo 26 vaut 0, cela veut dire qu'il faut coder par la lettre A. Or cette lettre est au rang 1, dans la liste « abc ».

Scénario

- Lorsqu'on appuie sur la lettre « c » le lutin demande la clé de cryptage, puis la phrase à coder

- Il code la phrase et il la dit. Nous la voyons aussi affichée dans la variable « phraseCodée », ce qui va nous faciliter le test de la partie décodage.

- Lorsqu'on appuie sur la lettre « d », le lutin demande la clé de cryptage et la phrase à décoder.

- Il décode la phrase et l'énonce

Programmation : Version plus ludique

Repartir de la version simple.

On utilisera :

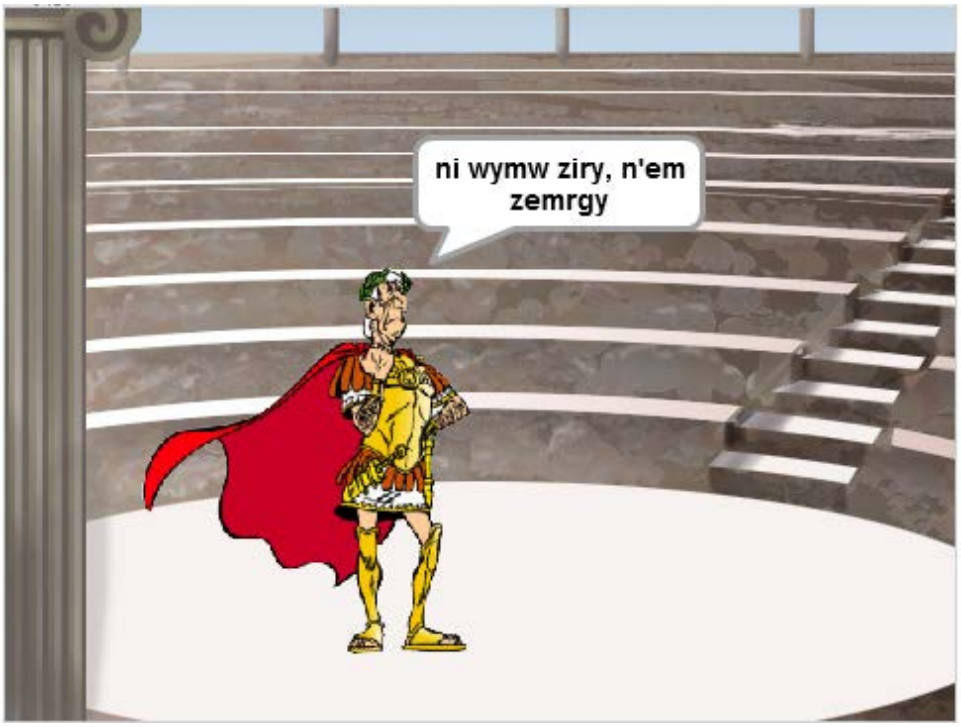
- Deux lutins, « César » et un « centurion ». Les costumes sont fournis. César vient remplacer le lutin de la version simple.
- Deux arrière-plans : un pour César et un pour le centurion. Ils sont fournis.
- Nous avons les mêmes variables.
    - La variable globale « **clef** » est visible par César et par le centurion, ce qui évitera de la redemander.
    - La variable globale « **phraseCodée** » est visible par César et par le centurion, ce qui évitera aussi de la redemander.
    - La variable locale « **phrase** » est locale au lutin « César ». Le lutin « Centurion » qui devra décoder la phrase ne doit pas avoir accès à cette phrase.
    - Supprimer la variable « **phraseDécodée** » dans le lutin « César » et la recréer en **local** dans le lutin « centurion »
- Dupliquer dans le script du centurion tout le code nécessaire au décodage de la phrase cryptée. Pour cela il suffit depuis la zone de script du lutin « César » de faire glisser les blocs à dupliquer, sur le lutin « centurion ». Une fois le code dupliqué, supprimer ce code du

lutin « César ». Celui-ci ne s'occupera plus que du codage, et le lutin « centurion » du décodage.

Scénario

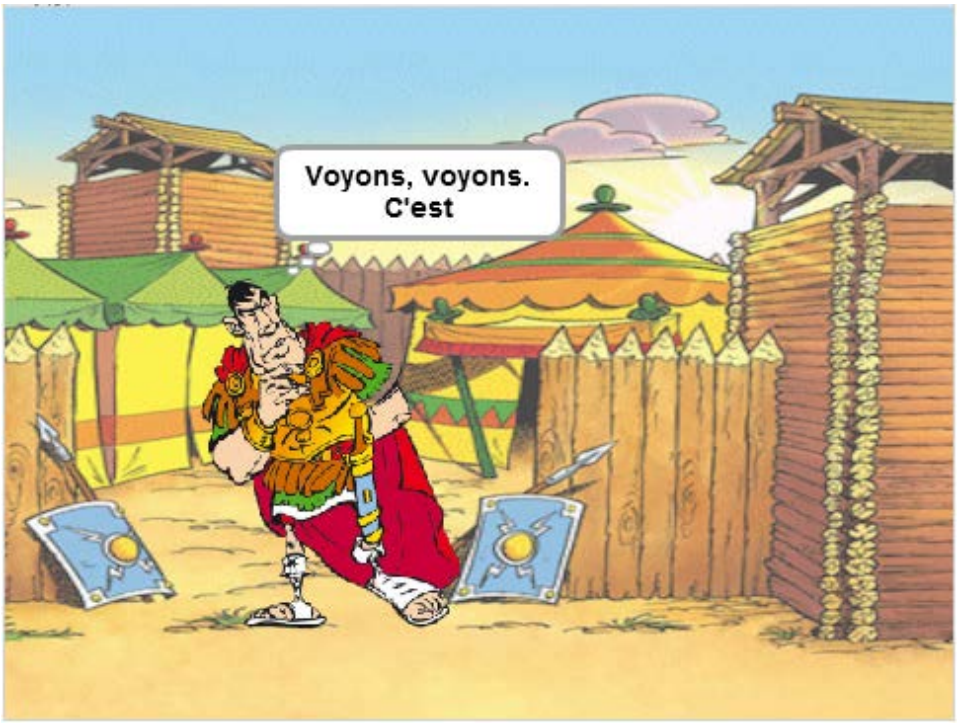
- Au démarrage, César est debout dans son théâtre.
- Il demande la clé de cryptage, la mémorise dans la variable globale « **clef** », puis il demande la phrase à crypter.

- Il exécute le cryptage, la phrase cryptée est mémorisée dans la variable globale « **PhraseCodée** » et il dit la phrase cryptée.



- Il envoie à tous le message « **décoder** ».
- L'arrière-plan bascule sur le camp romain, et le centurion apparait.
- Celui-ci a accès à la clé de codage et à la phrase codée, puisque les deux variables sont globales. Il énonce ces deux données.


le message est "ni wymw ziry, n'em zemrgy"


Voyons, voyons. C'est

- Il décode la phrase et dit la phrase décodée.