Td Exercice JS B

L'objectif de cet exercice est de continuer la découverte du langage JavaScript en créant une animation sur une page Web. A l'ouverture de la page, on a l'écran de la figure 1.

Après click sur la flèche gauche ou droite, le personnage se déplace dans la fenêtre.

On termine l'activité en mettant en place une animation automatique.

ETAPE 1 : ---------- **Initialisation du travail** ----------

Il s'agit dans un premier temps de télécharger le dossier nommé ***desert.zip***, à télécharger à partir du site www.nsibranly.fr .

Décompresser ce dossier **dans** votre zone de travail, sur votre ordinateur.
Vous y trouverez dossier différents fichiers images et un fichier *html* nommé *:* ***desert.html***

⇨ Ouvrir ce fichier sur *VisualStudioCode* et sur le navigateur *Firefox* .

ETAPE 2 : ---------- **Ecriture du code Html** ----------

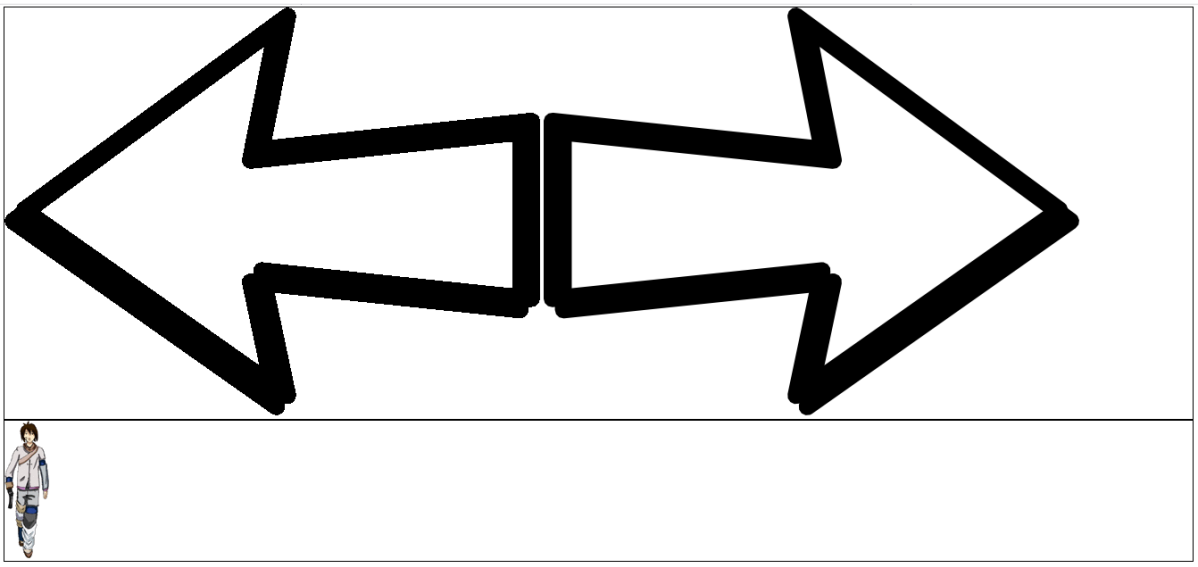
⇨ Créer un bloc div avec l'attribut class= '*conteneur_fleches* ' et un autre bloc div d'attribut class='conteneur_personnage'. Mettre une bordure autour de ces blocs dans la partie Css.

⇨ Placer dans le premier bloc div, les 2 flèches :

```
<img class="fleche_gauche" src="gauche.png">
<img class="fleche_droite" src="droite.png">
```

⇨ Placer dans le second bloc div, l'image 0 du personnage : `<img class="personnage" src="position0.png">`

Cela donne :



⇨ Visualiser toujours régulièrement le résultat sur le navigateur, ici Firefox.

ETAPE 3 : ----------**Ecriture du code Css** ----------

En visualisant toujours régulièrement le résultat sur le navigateur, ici Firefox :

⇨ Dans la partie Css, appliquer sur le body : background-image: url("desert.jpg") ; et background-size: **cover** ;
⇨ Dans la partie Css, appliquer sur l'image avec l'attribut 'fleche_gauche' : width:80px; margin:10px;
⇨ Dans la partie Css, appliquer sur l'image avec l'attribut la 'fleche_droite' : float:right; width:80px; margin:10px;
⇨ Dans la partie Css, appliquer sur le div avec l'attribut 'conteneur_personnage' : height :400px ;
⇨ Dans la partie Css, appliquer sur l'image avec l'attribut 'personnage' : position:relative; height:200px

ETAPE 4 : ---------- **Ecriture du code JavaScript : PARTIE 1** ----------

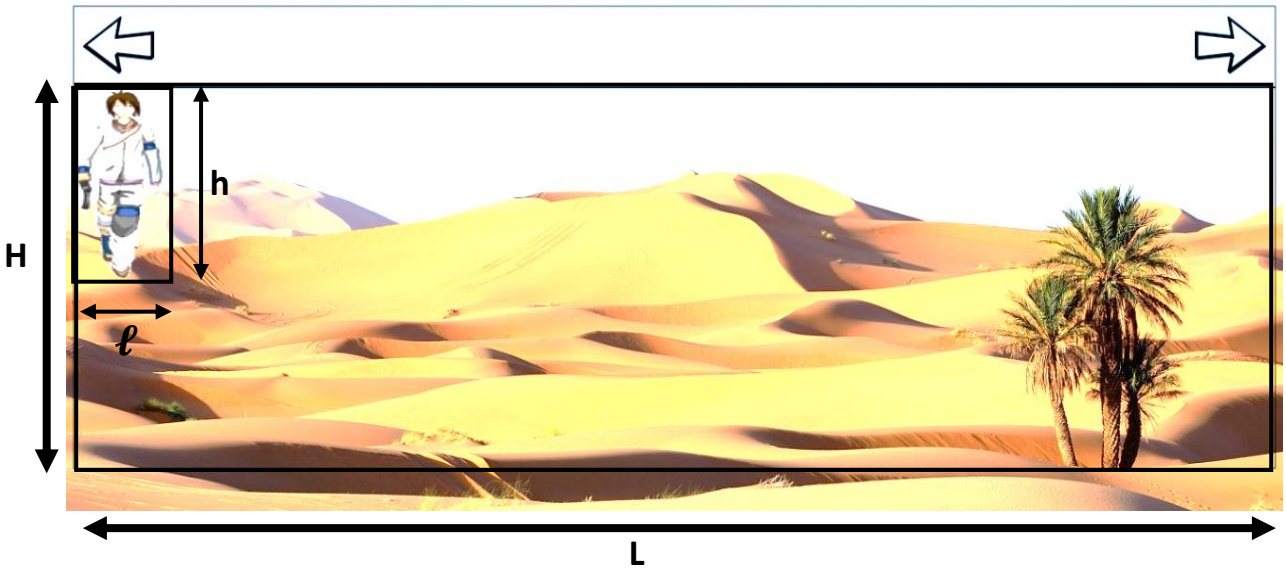
Notre premier objectif est d'écrire un code JavaScript qui permette de centrer l'image du personnage.

⇨ On commence par repérer les différents éléments Html qui seront concernés en les liant à des variables javaScript :

```
/* Repérage des éléments html */
let fleche_gauche = document.querySelector(".fleche_gauche");
let fleche_droite = document.querySelector(".fleche_droite");
let conteneur_personnage = document.querySelector(".conteneur_personnage");
let personnage = document.querySelector(".personnage");
```

⇨ L'image du personnage a une hauteur h = 100 px et une largeur ℓ = 200 px. Le bloc *div* contenant le personnage a une hauteur H et une largeur L qui dépend de l'écran sur lequel on travaille. Afin de gérer la position de l'image dans ce bloc, on récupère les valeurs de H et L en écrivant :

```
/* Positionnement initial de l'image */
  let L = (conteneur_personnage.clientWidth);
  let H = (conteneur_personnage.clientHeight);
```

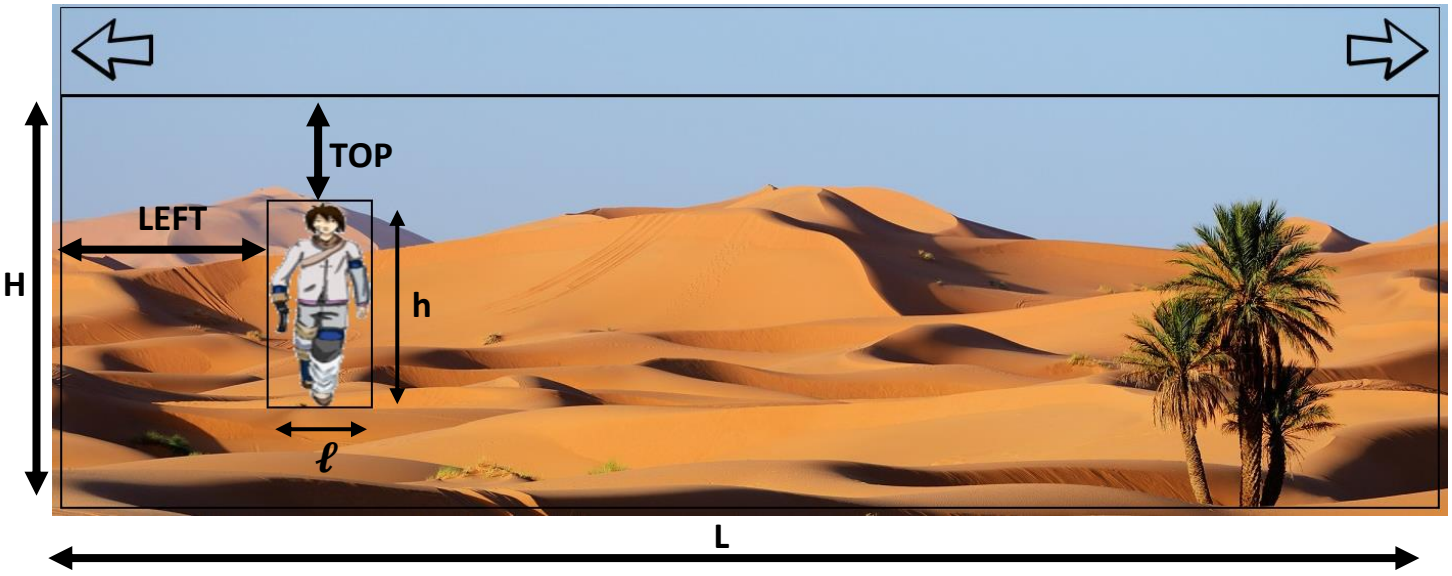


⇨ Ecrire les valeurs des variables L et H dans la console Web de Firefox, en écrivant : `console.log(L,H)`
Ces valeurs sont données en px. **Vérifier leur exactitude en faisant varier la taille de la fenêtre.**

⇨ Dans la partie Css, appliquer en plus sur l'image avec l'attribut 'personnage' : left :195px ; top :96px et une bordure :

```
.personnage{
            position:relative;
            height:200px;
            left:195px;
            top:96px;
            border:2px solid black;
}
```

L'image se déplace à 195 px du bord gauche et à 96 px du bord haut du bloc div d'attribut 'conteneur_personnage' :



⇨ Montrer que pour positionner l'image au centre du bloc div, il faut que les variables LEFT et TOP soient égales à :

$$LEFT = \frac{L-\ell}{2} \quad \text{et} \quad TOP = \frac{H-h}{2}$$

⇨ On ajoute les commandes ci-contre dans le code JavaScript afin de modifier les commandes Css « *left* et *top* » par l'intermédiaire du JavaScript :

```
/* Positionnement initial de l'image */
let L = (conteneur_personnage.clientWidth);  /*largeur écran*/
let H = (conteneur_personnage.clientHeight);   /*hauteur bloc*/
let l = 100; /*largeur image*/
let h = 200;  /*hauteur image*/
let LEFT = (L-l)/2;
let TOP = (H-h)/2;
console.log(LEFT , TOP);
personnage.style.left=(LEFT+"px");   /*positionnement horizontal*/
personnage.style.top = (TOP+"px");   /*positionnement vertical*/
```

⇨ Vérifier que l'image soit à présent centrée.

ATTENTION : c'est $\frac{L-\ell}{2}$ ne pas confondre le ℓ avec 1

On maîtrise à présent la position du personnage en jouant sur les variables LEFT et TOP.

ETAPE 5 : ---------- **Ecriture du code JavaScript : PARTIE 2** ----------

Notre second objectif est d'écrire un code JavaScript qui permette au click sur les flèches droite ou gauche, de modifier l'image du personnage.

⇨ On crée un évènement lié au click sur la flèche droite :

```
/* Creation d'évènements liés aux clicks sur les flèches*/
fleche_droite.addEventListener('click',suivante);
```

⇨ On définit la fonction « *suivante* » qui permet de changer d'image :

```
/* Creation d'évennements liés aux clicks sur les flèches*/
fleche_droite.addEventListener('click',suivante);
let n = 0;
function suivante(){
     if(n == 47){n = 0}
     else {n = n + 1}
     let nom_fichier = "position"+n+".png";
     personnage.setAttribute('src',nom_fichier);
}
```

La variable n repère le numéro de l'image.

On change le nom du fichier de l'image du personnage

⇨ Vérifier le bon fonctionnement du code en cliquant sur la flèche droite.

⇨ Continuer le code en créant un évènement lié au click sur la flèche gauche afin de modifier les images en prenant ici l'image précédente. Créer ainsi une fonction que vous appellerez « *précédente* ».

⇨ Vérifier le bon fonctionnement du code en cliquant sur la flèche gauche. **ATTENTION** à ne pas initialiser à nouveau la variable n et surtout à ne pas la redéfinir une seconde fois avec ***let***.

ETAPE 6 : ---------- **Ecriture du code JavaScript : PARTIE 3** ----------

Notre troisième objectif sera de faire en sorte que l'image du personnage se déplace en avant, arrière, droite ou gauche, suivant sa position.

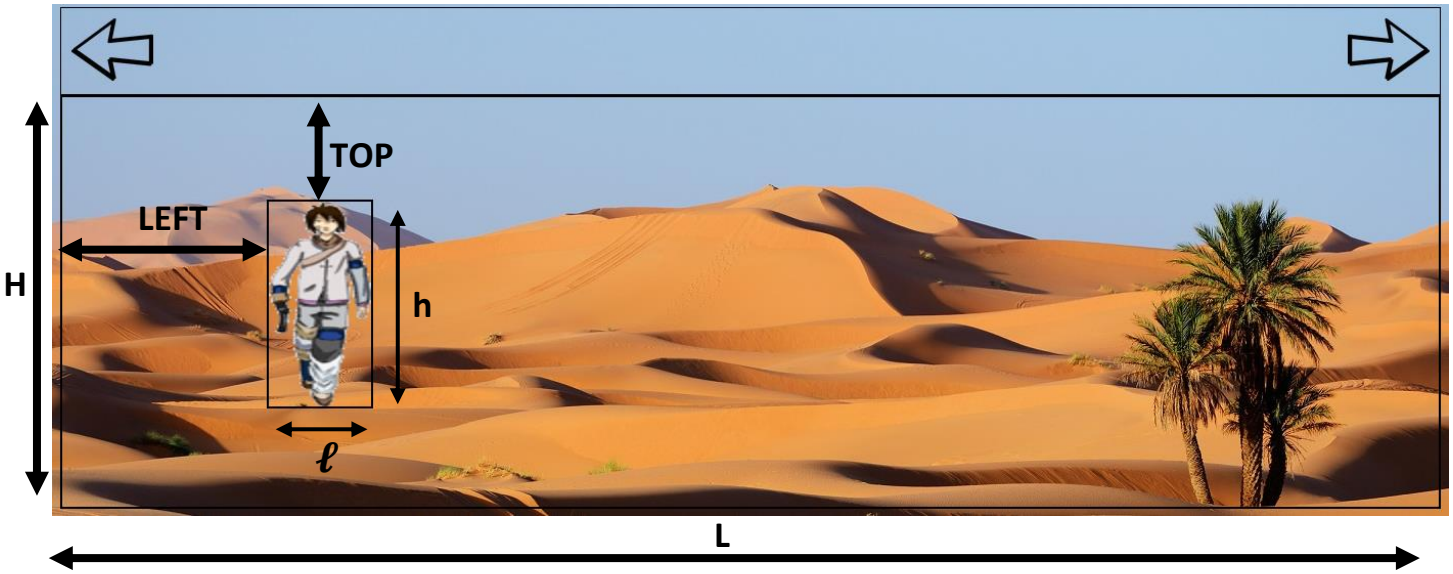
Vérifier que dans le répertoire, les images du personnage:

- nommées de position**0**.png à position**11**.png représentent le personnage marchant vers l'avant,
- nommées de position**12**.png à position**23**.png représentent le personnage marchant vers la gauche,
- nommées de position**24**.png à position**35**.png représentent le personnage marchant vers la droite,
- nommées de position**36**.png à position**47**.png représentent le personnage marchant vers l'arrière,

Le scénario que l'on va coder est le suivant :

- Au chargement de la page, le personnage est dans la position **0** (position**0**.png) et se trouve à droite de l'écran, en hauteur réduite : 100px au lieu des 200px définis dans le CSS avant.
- En cliquant sur la flèche  , on incrémente la position et :
    - la taille du personnage augmente jusqu'à la position **11** en jouant sur la hauteur de l'image,
    - le personnage se déplace vers la gauche jusqu'à la position **23** en jouant sur la variable LEFT,
    - le personnage se déplace vers la droite jusqu'à la position **35** en jouant sur la variable LEFT,
    - la taille du personnage diminue jusqu'à la position **47** en jouant sur la hauteur de l'image.

⇨ On commence par modifier la variable LEFT pour placer le personnage à droite et la hauteur de l'image 100px au lieu de 200px :



- pour que le personnage soit à droite de la fenêtre, on doit avoir : $LEFT = L - \ell$
- pour que l'image du personnage ait une hauteur de 100px, on doit mettre height :100px dans le Css

Cela donne dans le code javaScript:

```
/* Repositionnement initial de l'image à droite avec une taille réduite */
LEFT = (L-l);
let hauteur = 100
personnage.style.left=(LEFT+"px");
personnage.style.height=(hauteur+"px");
```

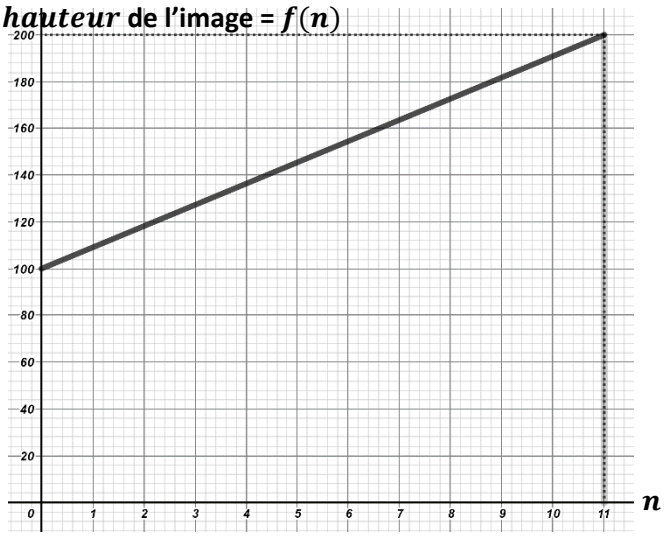
⇨ Vérifier le bon fonctionnement …..

ATTENTION : c'est $(L - \ell)$, ne pas confondre le ℓ avec 1

⇨ On continue en insérant dans le code de la fonction suivante déjà écrit, un bloc if qui modifie la hauteur de l'image en la faisant passer de 100px pour l'image *position**0**.png* à 200px pour l'image *position**11**.png*

```
function suivante(){
     if(n == 47){n = 0}
     else {n = n + 1}
     let nom_fichier = "position"+n+".png";
     personnage.setAttribute('src',nom_fichier);
     /* Gestion des images des positions 0 à 11 */
      if(n >= 0 & n <= 11){
           let hauteur = 100/11*n+100;
           personnage.style.height=(hauteur+"px");
      }
}
```

Comment calculer la hauteur de l'image en fonction de n *?* …….. On utilise les fonctions affines ….



On a : $f(n) = a\,n + b$

avec $f(0) = 100$ et $f(11) = 200$

On en déduit que : $\begin{cases} a \times 0 + b = 100 \\ a \times 11 + b = 200 \end{cases}$

Soit le système :

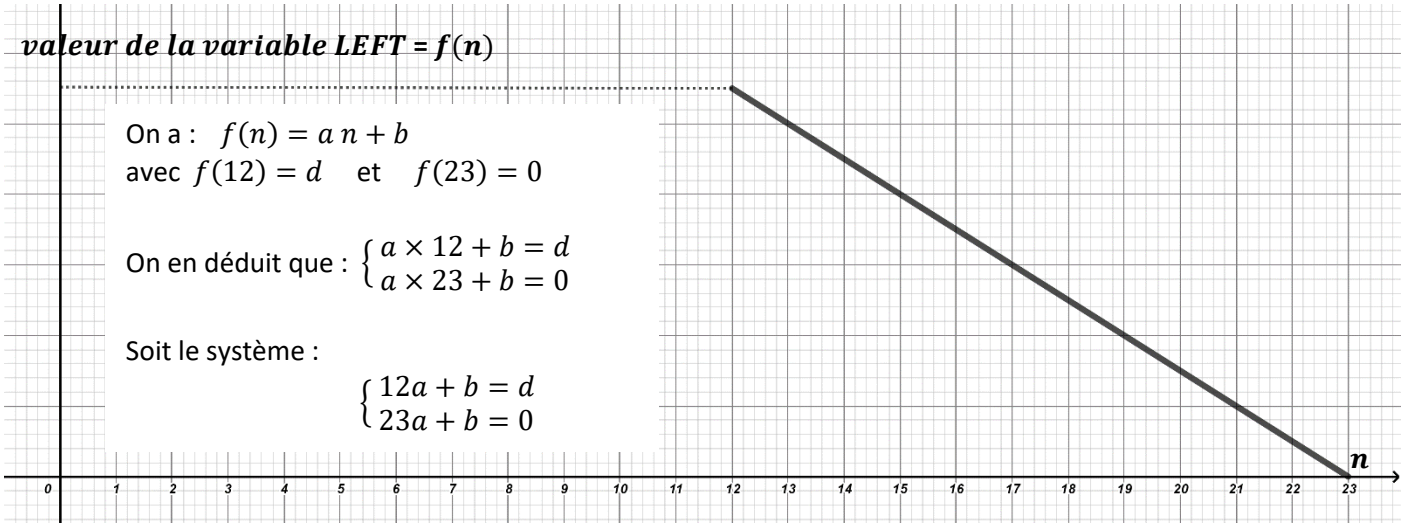
$$\begin{cases} b = 100 \\ 11a + b = 200 \end{cases}$$

D'où : $a = \frac{100}{11}$ et $b = 100$

Et donc : $\boldsymbol{f(n) = \frac{100}{11}n + 100}$

⇨ On continue en gérant le déplacement vers la gauche en modifiant la variable LEFT qui passe d'une valeur égale à $(L - \ell)$ pour la position **12** (*position**12**.png*) à 0 pour la position **23** (*position**23**.png*).

Pour simplifier on pose : $d = (L - \ell)$



On a : $f(n) = a\,n + b$
avec $f(12) = d$ et $f(23) = 0$

On en déduit que : $\begin{cases} a \times 12 + b = d \\ a \times 23 + b = 0 \end{cases}$

Soit le système :

$$\begin{cases} 12a + b = d \\ 23a + b = 0 \end{cases}$$

En soustrayant la 1ère ligne et la 2nde , on obtient : $-11a = d$ d'où : $a = \frac{-d}{11}$

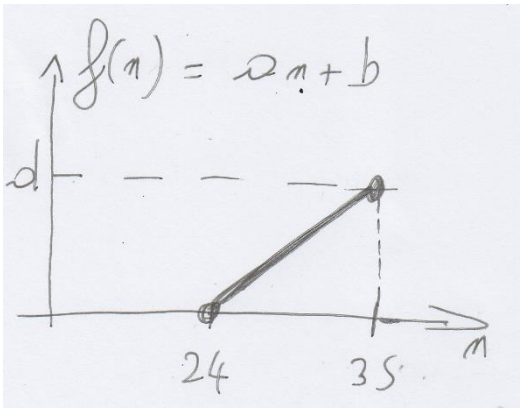
En utilisant la 2nde ligne, on obtient : $b = -23a = \frac{23\,d}{11}$

Et donc : $\boldsymbol{f(n) = \frac{-d}{11}n + \frac{23d}{11}}$

Partie code python, cela donne ceci à insérer dans la fonction ***suivante*** :

```
/* Gestion des images des positions 12 à 23 */
if(n >= 12 & n <= 23){
        let d = (L-l);
        LEFT = -d/11*n + 23*d/11
        personnage.style.left=(LEFT+"px");
}
```

⇨ Vérifier le bon fonctionnement …..

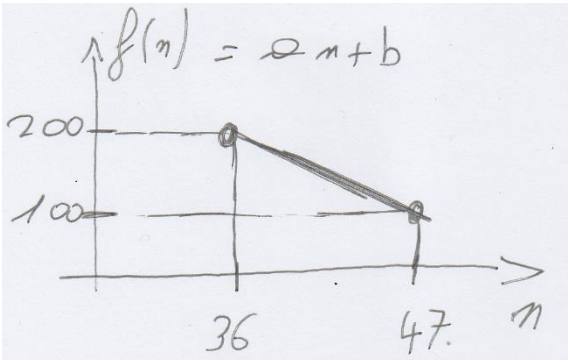


⇨ On continue en gérant le déplacement vers la droite en modifiant la variable LEFT qui passe d'une valeur égale à 0 pour la position **24** (*position**24**.png)* à $(L - \ell)$ pour la position **35** (*position**35**.png).*

- trouver l'expression $f(n)$ de la fonction affine f
- insérer dans la fonction ***suivante*** le bout de code correspondant.

⇨ Vérifier le bon fonctionnement …..

⇨ On termine en gérant le déplacement vers l'arrière en modifiant la hauteur de l'image du personnage qui passe d'une valeur égale à 200 px pour la position **36** (*position**36**.png)* à 100 px pour la position **47** (*position**47**.png).*



-trouver l'expression $f(n)$ de la fonction affine f
-insérer dans la fonction ***suivante*** le bout de code correspondant.

⇨ Vérifier le bon fonctionnement …..

⇨ Penser à enlever dans le CSS les bordures ajoutées au début de l'exercice.

ETAPE 8 : ---------- **Amélioration pour un affichage sur smartphone** ----------

Sur smartphone, ce n'est pas toujours facile de cliquer sur les flèches. On peut écrire dans le code une animation automatique.

Pour cela on écrit en fin de code JavaScript :

```
for (let i = 0 ; i<200; i++){setTimeout(suivante,i*100);}
```

On exécute ainsi la fonction ***suivante*** toutes les 100 millisecondes durant 200 x 100 = 20 000 millisecondes, soit sur une durée de 20 secondes.

⇨ Vérifier le bon fonctionnement …..

⇨ Pour réserver cette animation automatique aux smartphones qui ont une largeur d'écran inférieure à 600px par exemple, on modifie insérant un bloc ***if*** :

```
if (L < 600){
        for (let i = 0 ; i<500; i++){setTimeout(suivante,i*100);}
  }
```

⇨ Vérifier le fonctionnement