Question n°1 :

```
> chaine1 = "La poule est un animal"
< 'La poule est un animal'
> chaine1[0] = 5
< 5
> chaine1
< 'La poule est un animal'
```

Le changement de caractère ne s'est pas produit car il n'est pas possible de changer les caractères d'une chaine avec cette expression.

Tout simplement parce qu'en Javascript, les chaines de caractère sont immuables (elles ne peuvent pas être modifiées)

Question n°2 :

```
> 10 == `10`
< true
> 10 === `10`
< false
```

Je remarque que la première formule avec deux fois égale « == » la réponse donnée est "true", alors que la deuxième formule, avec un troisième égale « === », on a comme réponse "false".

Ces deux formules n'ayant pas les mêmes fonctions, c'est pour cela que nous avons deux réponses différentes.

« == » permet de comparer les valeurs des deux éléments : 10 == '10' converti la chaine de caractère '10' en nombre avant de la comparer avec le premier 10, c'est pour cela que la réponse est "true" (les deux valeurs sont égales)

« === » permet de comparer les valeurs et les types d'éléments : 10 === '10' ne converti pas la chaine de caractère '10' avant de comparer 10 et '10', c'ets pour cela que la réponse est "false" (10 étant un « number » et '10' étant un « string », les types d'éléments ne sont pas identiques).

Question n°3 :

```
> let texte = "C'était le meilleur des temps, c'était le pire des temps ; \
  c'était l'âge de la sagesse, c'était l'âge de la folie ; c'était \
  l'époque de la foi, c'était l'époque de l'incrédulité ; c'était la \
  saison de la Lumière ; c'était la saison de l'Obscurité ; c'était \
  le printemps de l'espoir, c'était l'heure du désespoir ; nous \
  avions tout devant nous, nous n'avions rien devant nous ; \
  nous devions tous aller directement au Ciel, nous devions \
  tous prendre l'autre chemin ; bref, l'époque était tellement \
  différente de celle que vous vivons aujourd'hui que quelques- \
  unes des plus tapageuses autorités ne parlaient d'elles, que \
  ce fut en bien ou en mal, qu'au superlatif.";
< undefined
> let longueur = texte.length;
< undefined
> texte.length
< 662
> let mots = texte.match(/\b\w{6,}\b/g);
< undefined
> mots
<   (25) ['meilleur', 'sagesse', 'saison', 'saison', 'Obscurit', 'printemps', 'espoi
  r', 'sespoir', 'avions', 'devant', 'avions', 'devant', 'devions', 'directement',
  'devions', 'prendre', 'chemin', 'tellement', 'vivons', 'aujourd', 'quelques', 'tap
  ageuses', 'autorit', 'parlaient', 'superlatif']
```

Après avoir déclaré le texte, je l'ai initialisé en respectant les sauts de lignes, pour ça j'ai ajouté des antislashs à la fin de chaque lignes (excepté la dernière pour laquelle on a mix un point-virgule).

La longueur de la phrase est de 662 caractères. J'ai d'abord définit une variable « longueur » ayant comme valeurs "texte.length" se qui a donné le résultat de la longueur de la phrase.

Il y a 25 mots de plus de 5 lettres n'étant pas suivis d'un espace. Pour les trouver j'ai eu recours au regex "/\b\w{6,}\b/g" que j'ai initialisé dans une variable « mot » qui avait pour valeur « texte.match » suivi du regex.

```
> texte = texte.replace(/[Cc]'était/g, '');
<- " le meilleur des temps,  le pire des temps ;  l'âge de la sagesse,  l'âge de la folie ;  l'époque de la foi,  l'époque de l'incrédulité ;  la saison de la Lumière ;  la saison de l'Obscurité ;  le printemps de l'espoir,  l'heure du désespoir ; nous avions tout devant nous, nous n'avions rien devant nous ; nous devions tous aller directement au Ciel, nous devions tous prendre l'autre chemin ; bref, l'époque était tellement différente de celle que vous vivons aujourd'hui que quelques- unes des plus tapageuses autorités ne parlaient d'elles, que ce fut en bien ou en mal, qu'au superlatif."
> texte = texte.replace(/^\s*/, '').replace(/^\w/, (match) => match.toUpperCase());
<- "Le meilleur des temps,  le pire des temps ;  l'âge de la sagesse,  l'âge de la folie ;  l'époque de la foi,  l'époque de l'incrédulité ;  la saison de la Lumière ;  la saison de l'Obscurité ;  le printemps de l'espoir,  l'heure du désespoir ; nous avions tout devant nous, nous n'avions rien devant nous ; nous devions tous aller directement au Ciel, nous devions tous prendre l'autre chemin ; bref, l'époque était tellement différente de celle que vous vivons aujourd'hui que quelques- unes des plus tapageuses autorités ne parlaient d'elles, que ce fut en bien ou en mal, qu'au superlatif."
```

Afin de retirer les « [Cc]'était » de la phrase j'ai utilisé la fonction "texte.replace" suivi du regex «

Pour retirer l'espace et mettre la première lettre en majuscule au début de la phrase j'ai utilisé la variable suivante : texte = texte.replace(/^\s*/, '').replace(/^\w/, (match) => match.toUpperCase());

Question n°4 :

```
> let data = "Lepautre, Jean/MALE/Paris/17051999"
<- undefined
> let slash = data.indexOf('/', data.indexOf('/') + 1)
<- undefined
> slash
<- 19
```

Le deuxième slash « / » se trouve en 19ème position dans la chaine de caractère.

Ici, j'ai extrait chaque élément de la date de naissance pour la remplacer par une nouvelle variable.

Je l'ai ensuite concaténé (« dateconca ») pour qu'elle apparaisse sous un nouveau format "AAAA/MM/JJ"