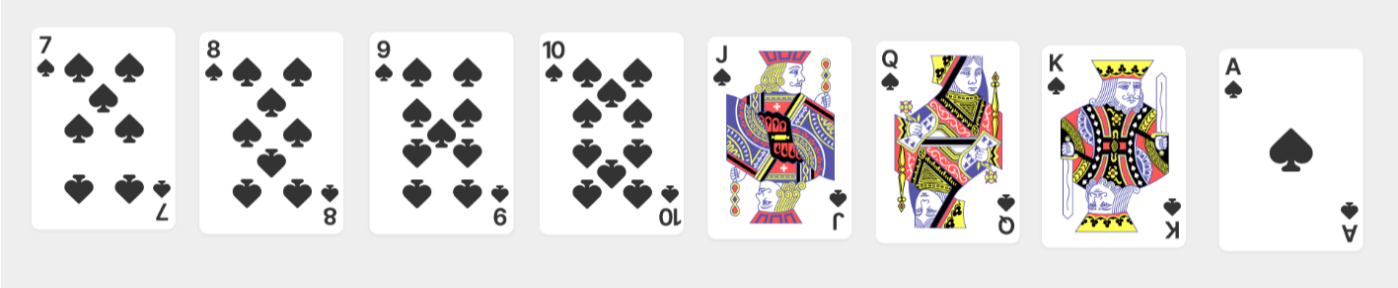
Algorithmique 1 - TD - Recherche dichotomique

L'énoncé de l'algorithme de **recherche dichotomique** peut s'énoncer de la manière suivante :

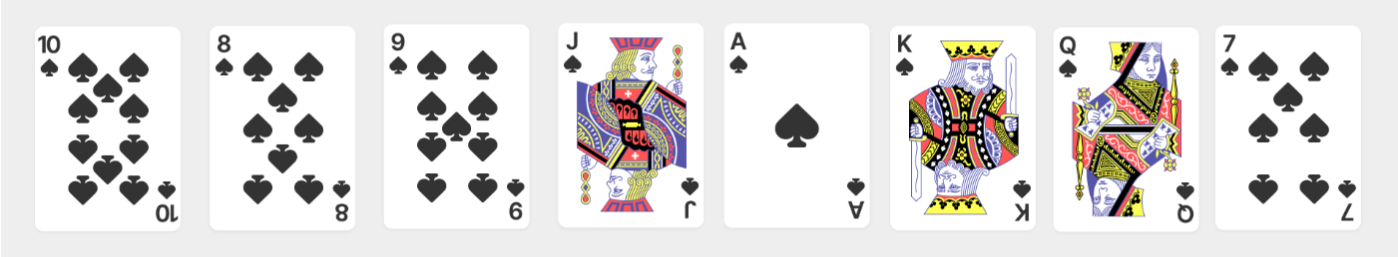
```
On dispose d'un jeu de cartes triées et d'une valeur X (recherchée)
Regarder la carte du paquet milieu (ou juste avant le milieu si le nombre est pair).
Tant Que la carte observée est différente de X, et qu'il reste des cartes dans le
paquet :
      Regarder la carte du milieu du paquet
      Si la carte du milieu est supérieure à milieu :
            Deplacer son regard dans la moitié gauche du paquet
      Sinon, si la carte du milieu est superieure :
            Deplacer son regard dans la moitié droite du paquet
Fin Tant Que
Si la carte observée est la même que X :
      Retourner le rang de la carte
Sinon :
      Retourner -1
```

Question a : Expliquer le fonctionnement de cet algorithme avec la recherche de la carte **`Valet de Pique`**.



Algorithmique 1 - suite - Tri par insertion

Question b : Utiliser maintenant la famille entière d'un jeu de cartes, mais dans le désordre. Par exemple :
(utiliser un VRAI jeu, ou bien la page https://deck.of.cards/)



Trouver une procédure pour remettre les cartes dans l'ordre. C'est ce que l'on appelle, un tri en place (on fait glisser, on créé des intervalles, etc…). Faire plusieurs essais.
Essayer de dégager une procédure que pourrait executer une machine (il faut donner des ordres précis).
Rediger cet algorithme en langage naturel, comme dans l'exemple ci-dessus (recherche dichotomique). (voir au verso)

```
1    ..
2    ..
3    ..
4    ..
5    ..
6    ..
7    ..
8    ..
..   ..
..   ..
```

Programmation

Le langage python possède une fonction de tri, sorted, qui retourne une liste triée à partir de celle en paramètre :
Exemple :

```
>>> liste_1 = ["Hello", "Bonjour", "Salut"]
>>> liste_2 = sorted(liste_1)
>>> print("liste_1 = ", liste_1)
liste_1 =  ['Hello', 'Bonjour', 'Salut']
>>> print("liste_2 = ", liste_2)
liste_2 =  ['Bonjour', 'Hello', 'Salut']
```

Aller sur la page :
https://mcoilhac.forge.apps.education.fr/site-nsi/tris/3_insertion/#ii-algorithme-du-tri-par-insertion-en-python

Faire les activités proposées. Pour compléter la fonction tri_insertion : celle-ci prend comme paramètre un `tableau` et le triant **en place** à l'aide du tri par insertion :

```
1    def tri_insertion(tableau):
2        for i in range( …   ,  …          ):
3            valeur_a_inserer = …
4            j = …
5            while j > … and tableau[…] > …
6                tableau[…] = tableau[…]
7                j = … - 1
8        tableau[j] = …
```

Question c : Compléter le script avec : (certains éléments peuvent être utilisés plusieurs fois)

```
0, 1, tableau[i], len(tableau), i, i-1, j, j-1, valeur_a_inserer
```

Question d : Expliquer ce que fait le tri par insertion lorsque l'on execute :

```
>>> valeurs = [0, 3, 2, 1, 6, 8]
>>> tri_insertion(valeurs)
```