Document 1 - Recherche linéaire

```
def recherche_lin(T, x):
  """
  :param T: list of elements
  :param x: element
  :return: int, index of x in the list
           else -1
  """
  i = 0
  while i < len(T) and T[i] != x:
    i = i + 1
  if i >= len(T):
    return -1
  else:
    return i
```

Questions

1. Que retourne l'instruction suivante:

   `recherche_lin(['a','b','c'], 'b')`

   Justifier sommairement.

2. Que retourne l'instruction suivante:

`recherche_lin(['a','b','c'], 'd')`

Justifier sommairement.

Document 2 - jeu du nombre mystère

Le jeu auquel vous allez jouer est celui d'une recherche dichotomique. Vous allez chercher un élément (une valeur) dans un ensemble trié (l'ensemble des entiers [0 ..100])

```
fonction devine_un_nombre(borne_inf: entier, borne_sup: entier) -> c: entier
  var N: entier[borne_inf .. borne_sup] <- tirage_aleatoire(borne_inf, borne_sup)
  var c: entier <- 0 # compteur du nombre d'essais
  var p: entier[borne_inf .. borne_sup] <- -1 # valeur proposée par l'adversaire
  tant que p != N faire:
    p <- saisie("quel nombre proposes tu?")
    c <- c + 1
    si p > N alors:
      afficher("c'est moins")
    sinon si p < N alors:
      afficher("c'est plus")
    sinon:
      afficher("bravo c'est gagné")
  fintantque
  retourner c
```

Questions: Joue à ce jeu contre l'ordinateur.

1. Faire plusieurs parties avec pour bornes: [0 .. 100]. Quelles sont les valeurs possibles pour c dans chaque cas? Donner un encadrement, en précisant bien la stratégie utilisée.

2. Quelles sont les puissances m et n de 2 qui encadrent la valeur $2^m < 100 < 2^n$?

3. Jouer à ce même jeu, mais avec les bornes: [0 .. 1000] Quelles sont les valeurs possibles pour c dans chaque cas? Donner un encadrement, en précisant bien la stratégie utilisée.
4. Comparer c avec n et m tels que $2^m < 1000 < 2^n$

Document 3 - recherche dichotomique

```
def recherche_dicho(T,x):
    """
    :param T: sorted list of elements
    :param x: element
    :return: int, index of x in the list
           else -1
    """
    g = 0
    d = len(T) - 1
    while (d >= g):
       m = (..
       if T[m] == x:
            return ..
        elif T[m] < x:
            g = ..
        else:
            d = ..
    return -1
```

Questions

1. Compléter la fonction de recherche dichotomique.
2. Peut-on utiliser cet algorithme pour effectuer une recherche dichotomique dans la liste `[0, 2, 5]`?
3. Même question pour la liste `[10, 2, 5]`?
4. Combien d'étapes sont nécessaires pour trouver la valeur 5 dans la liste `[0, 2, 5]` à partir de cet algorithme? Faire le suivi des variables **`g, d et m`** au cours de l'avancée du programme avec le tableau ci-dessous.
5. Un colloque scientifique réunit les noms de la liste ci-contre. Rechercher **`Morin Arthur`**. Comparer le nombre d'itérations avec l'algorithme de recherche séquentielle et celui de recherche dichotomique.

Participants:
Abraham, Max
Abt, Antal
Barlow, Peter
Bartoniek, Géza
Barus, Carl
Bauer, Edmond
Beetz, Johan
Belar, Albin
Blondel, André
Brewster, David
Brillouin, Léon
Dalén, Gustaf
Dolbear, Amos
Duhem, Pierre
Eötvös, Loránd
Fröhlich, Pál
Graetz, Leo
Hall, Edwin
Holten, Carl
Khvolson, Orest
Knudsen, Martin
Küch, Richard
Lamb, Horace
Lebedev, Peter
Lehmann, Otto
Lemoine, Jules
Marsden, Ernest
Morin, Arthur
Perrin, Jean
Poni, Petru
Soret, Charles
Weiss, Pierre
Zeeman, Pieter

Tableau pour le suivi des variables (question 3):

	d >= g ?	m	T[m]	Moitié à conserver (droite ou gauche ou fin ?)	g	d
Avant l'itération 1						
Après l'itération 1						
Après l'itération 2						
...						