TD1 : Dérouler, concevoir des algorithmes

1 Rappels : quelques algorithmes de recherche

(A) un algorithme de recherche dans un tableau non trié

```
public static int recherche(int cherche, int[] t){
  int index=-1;
  for (int i=0; i< t.length;i++){
    if(t[i]== cherche) {
      index=i;
    }
  }
  return index;
}
```

(B) un algorithme de recherche dans un tableau trié

```
public static int rechercheTableauTrie(int cherche, int[] t){
  int index=-1;
  boolean trouve=false;
  int i=0;
  while(!trouve && i<t.length && t[i]<=cherche){
    if(t[i]== cherche) {
      index=i;
      trouve=true;
    }
    i++;
  }
  return index;
}
```

(C) un algorithme de recherche dichotomique dans un tableau trié

```
public static int rechercheDicho(int cherche, int[] t){
  int debut=0;
  int fin=t.length-1;
  boolean trouve=false;
  int milieu= (debut+fin)/2;
  while(!trouve && debut<=fin){
    milieu= (debut+fin)/2;
    if (t[milieu]==cherche)
      trouve=true;
    else if(t[milieu]>cherche)
      fin=milieu-1
    else debut=milieu+1;
  }
  if (trouve)
    return milieu;
  else return -1;
}
```

2 Niveau des exercices

Les notations suivantes indiquent le niveau de difficulté des exercices :

⋆	Exercice de base, à maîtriser, et à savoir réaliser rapidement.
⋆⋆	Exercice de base, à maîtriser. Requiert un peu de reflexion/temps pour le réaliser.
⋆⋆⋆	Exercice d'approfondissement, à maîtriser. Requiert plus de reflexion/temps pour le réaliser que la moyenne des exercices.
⋆⋆⋆⋆	Exercice difficile, à faire en bonus – non traité de manière collective en TD. A ne réaliser que si les autres exercices sont déjà parfaitement maîtrisés.

3 Dérouler un algorithme

1. ⋆ Dérouler l'algorithme de recherche (A) pour les valeurs `cherche= 5` et `t= {4, 5, 12}`.
2. ⋆ Dérouler l'algorithme de recherche (A) pour les valeurs `cherche= 2` et `t= {4, 5, 12}`.
3. ⋆ Dérouler l'algorithme de recherche (B) pour les valeurs `cherche= 5` et `t= {4, 5, 12}`.
4. ⋆ Dérouler l'algorithme de recherche (B) pour les valeurs `cherche= 2` et `t= {4, 5, 12}`.
5. ⋆ Dérouler l'algorithme de recherche (B) pour les valeurs `cherche= 14` et `t= {4, 5, 12}`.
6. ⋆ Dérouler l'algorithme de recherche (C) pour les valeurs `cherche= 10` et `t= {4, 5, 5, 10, 12}`.
7. ⋆ Dérouler l'algorithme de recherche (C) pour les valeurs `cherche= 11` et `t= {4, 5, 5, 10, 12}`.

4 Déterminer des valeurs de paramètres pour une exécution au pire cas

1. ⋆ Donner des valeurs de paramètres pour une exécution au pire cas pour l'algorithme de recherche (A) avec un tableau de 7 éléments. Justifier.
2. ⋆⋆ Donner des valeurs de paramètres pour une exécution au pire cas pour l'algorithme de recherche (B) avec un tableau de 7 éléments. Justifier.
3. ⋆⋆⋆ Donner des valeurs de paramètres pour une exécution au pire cas pour l'algorithme de recherche (C) avec un tableau de 7 éléments. Justifier.

5 Modifier un algorithme

1. ⋆ Modifier l'algorithme de recherche (A) pour améliorer ses performances, en arrêtant la recherche dès que l'élément a été trouvé.
2. ⋆ Modifier l'algorithme de recherche (A) pour déterminer combien d'**occurrences** de la valeur `cherche` sont présentes dans le tableau `t`.
3. ⋆⋆ Modifier l'algorithme de recherche (B) pour déterminer combien d'**occurrences** de la valeur `cherche` sont présentes dans le tableau `t`.
4. ⋆⋆ Modifier l'algorithme de recherche (A) pour déterminer l'élément **minimum** d'un tableau `t`.
5. ⋆⋆⋆ Modifier l'algorithme de recherche (A) pour déterminer si le tableau `t` contient **plusieurs fois** un même élément.