TD 12
Recherche dichotomique

Algorithmique
Semestre 1

1 Spécification

1.1

```
1  --Rechercher dans le tableau tab de n elements le rang s'il existe d'un
       element x
2  --si l'element existe trouve=VRAI et le rang est celui de l'une de ses
       occurences
3  --sinon trouve=FAUX
4  procedure rechercherParDihotomie(entree tab <TabEntier>,
5                   entree n <Entier>,
6                   entree x <Entier>,
7                   sortie trouve <Booleen>,
8                   sortie rang <Entier>);
```

Listing 1 – Entête de rechercherParDichotomie

1.2

```
1  -- Rechercher dans le tableau tab de n elements le rang s'il existe d'un
       element x
2  -- si l'element existe trouve=VRAI et le rang est celui de l'une de ses
       occurences
3  -- sinon trouve=FAUX
4
5  -- Necessite
6  --1 <= n  N declenche trancheInvalide
7  -- \forall i \in [1..n-1], tab[i] <= tab[i+1]
8
9  -- Entraine
10 -- \forall i \in [1..n], tab[i] /= x => trouve = FAUX
11 -- \exists i \in [1..n], tab[i] = x => trouve = VRAI
12
13
14 procedure rechercherParDihotomie(entree tab <TabEntier>,
15                   entree n <Entier>,
16                   entree x <Entier>,
17                   sortie trouve <Booleen>,
18                   sortie rang <Entier>);
```

Listing 2 – Entête de rechercherParDichotomie avec précondition et postcondition

2 Algorithme

2.1

On divise le tableau en deux sous ensemble si on n'a pas trouvé l'élément on répéte(=>boucle) le processus dans la tranche où il peut se trouver.

2.2

3 Programmation

3.1

Tranche non valide : $iDeb \leq iFin$
Élément non trouvé : $tab[iMil] \neq x$

3.2

Réduire à la partie gauche : iFin <- iMil - 1 ;
Réduire à la partie droite : iDeb <- iMil + 1 ;

3.3

```
1  -- Rechercher dans le tableau tab de n elements le rang s'il existe d'un
       element x
2  -- si l'element existe trouve=VRAI et le rang est celui de l'une de ses
       occurences
3  -- sinon trouve=FAUX
4
5  -- Necessite
6  --1 <= n  N declenche trancheInvalide
7  -- \forall i \in [1..n-1], tab[i] <= tab[i+1]
8
9  -- Entraine
10 -- \forall i \in [1..n], tab[i] /= x => trouve = FAUX
11 -- \exists i \in [1..n], tab[i] = x => trouve = VRAI
12
13
14 procedure rechercherParDihotomie(entree tab <TabEntier>,
15                 entree n <Entier>,
16                 entree x <Entier>,
17                 sortie trouve <Booleen>,
18                 sortie rang <Entier>)
19
20 glossaire
21   iDeb <Entier>; --indice debut tranche
22   iFin <Entier>; --indice fin tranche
23   iMil <Entier>; --indice de l'element du milieu
24
25 debut
26   si (n < 1) ou (n > N) alors
27     declancher trancheInvalide;
28   fin si;
29
30   iDeb <- 1;
31   iFin <- n;
32   iMil <- (iDeb + iFin) div 2; -- POINT D'ARRET 1
33
```

```
34    tantque iDeb <= iFin et tab[iMil] /= x faire
35      si tab[iMil] > x alors
36        iFin <- iMil - 1; -- POINT D'ARRET 2
37      sinon
38        iDeb <- iMil  +1; -- POINT D'ARRET 3
39      fin si;
40      iMil <- (iDeb + iFin) div 2; -- POINT D'ARRET 4
41    fin tantque;
42    si iDeb > iFin alors
43      trouve <- FAUX; --POINT D'ARRET 5
44    sinon -- POINT D'ARRET 6
45      trouve <- VRAI;
46      rang <- iMil;
47    fin si;
48  fin
```

Listing 3 – Corps de rechercherParDichotomie

3.4

3.4.1 Pour x = 5

Situation	iDeb	iFin	iMi	tab[iMil]	trouvé	rang
1	1	11	6	35		
2	1	5	6	35		
4	1	5	3	20		
2	1	2	3	20		
4	1	2	1	10		
2	1	0	1	10		
4	1	0	0			
5	1	0	0		FAUX	

3.4.2 Pour x = 10

Situation	iDeb	iFin	iMi	tab[iMil]	trouvé	rang
1	1	11	6	35		
2	1	5	6	35		
4	1	5	3	20		
2	1	2	3	20		
4	1	2	1	10		
6	1	2	1	10	VRAI	1

3.4.3 Pour x = 20

Situation	iDeb	iFin	iMi	tab[iMil]	trouvé	rang
1	1	11	6	35		
2	1	5	6	35		
4	1	5	3	20		
6	1	5	3	20	VRAI	3

3.4.4 Pour x = 5

Situation	iDeb	iFin	iMi	tab[iMil]	trouvé	rang
1	1	11	6	35		
3	7	11	6	35		
4	7	11	9	65		
3	10	11	9	65		
4	10	11	10	70		
3	11	11	10	70		
4	11	11	11	75		
3	12	11	11	75		
5	12	11	11	75	FAUX	

3.4.5 Pour x = 5

Situation	iDeb	iFin	iMi	tab[iMil]	trouvé	rang
1	1	11	6	35		
2	1	5	6	35		
4	1	5	3	20		
2	1	2	3	20		
4	1	2	1	10		
2	1	0	1	10		
4	1	0	0			
5	1	0	0		FAUX	

3.4.6 Pour x = 80

Situation	iDeb	iFin	iMi	tab[iMil]	trouvé	rang
1	1	11	6	35		
3	7	1	6	35		
4	1	5	3	20		
3	1	2	3	20		
4	1	2	1	10		
3	1	0	1	10		
4	1	0	0			
3	1	0	0		FAUX	

3.5

$$2^{k-1} < n < 2^k$$

$$k - 1 < \log(n) \leq k$$

$$O(\log(n))$$

n	Recherche séquentielle	Recherche dichotomique
10	10	$4(2^4)$
100	100	7
⋮	⋮	⋮

3.6

```
1  fonction estTrie (entree <TabEntier>,
2           entree n <Entier>)
3           retourne <Booleen>
4
5  glossaire
6    i <Entier>;
7
```

```
8  debut
9    i <- 0;
10
11   tantque i < n faire
12     si tab[i] > tab[i+1] alors
13       retourner FAUX;
14     fin si;
15     i <- i+1;
16   fin tantque;
17   retourner(VRAI);
18 fin
```

Listing 4 – Fonction estTrié

4 Ecriture récursive

```
1  -- Rechercher dans le tableau tab de n elements le rang s'il existe d'un
       element x
2  -- si l'element existe trouve=VRAI et le rang est celui de l'une de ses
       occurences
3  -- sinon trouve=FAUX
4
5  -- Necessite
6  --1 <= n  N declenche trancheInvalide
7  -- \forall i \in [1..n-1], tab[i] <= tab[i+1]
8
9  -- Entraine
10 -- \forall i \in [1..n], tab[i] /= x => trouve = FAUX
11 -- \exists i \in [1..n], tab[i] = x => trouve = VRAI
12
13
14 procedure rechercherParDihotomieR(entree tab <TabEntier>,
15                 entree x <Entier>,
16                 entree iDeb <Entier>,
17                 entree iFin <Entier>,
18                 sortie trouve <Booleen>,
19                 sortie rang <Entier>)
20
21 glossaire
22   iMil <Entier>;
23
24 debut
25   si iDeb <= iFin alors
26     iMil <- (iDeb + iFin) div 2;
27     si tab[iMil] = x alors
28       trouve <- VRAI;
29       rang <- iMil;
30     sinon
31       si x < tab[iMil] alors
32         rechercherParDihotomieR(tab, x, iDeb, iMil, trouve, rang);
33       sinon
34         rechercherParDihotomieR(tab, x, iMil+1, iFin, trouve, rang);
35       fin si;
36     fin si;
37   sinon
38     trouve <- FAUX;
39   fin si;
```

```
40  fin
```

Listing 5 – procédure rechercherParDichotomie en récurisf