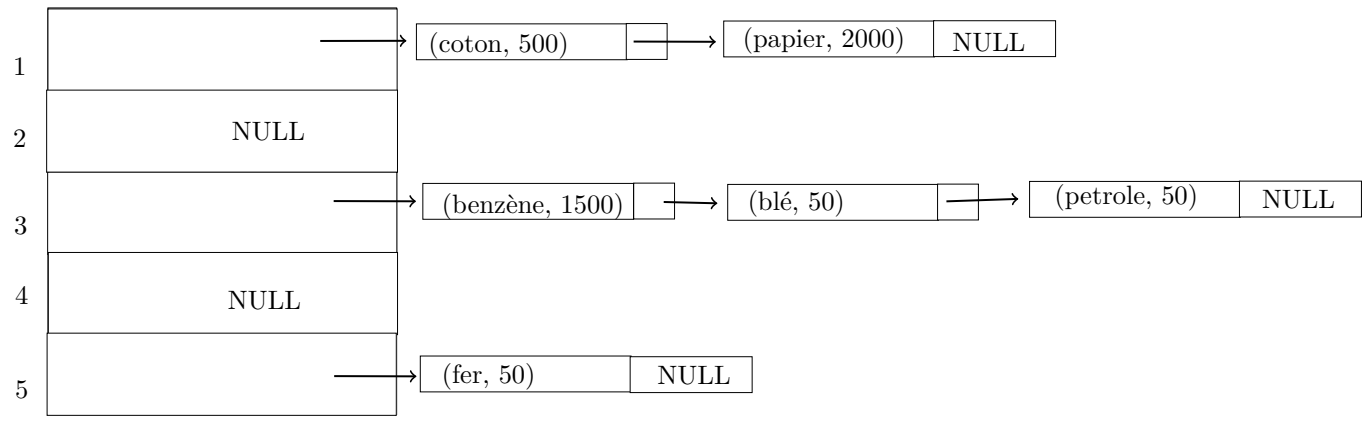
TD 11
Représentation physique d'une file avec priorité

TAD
Semestre 2

1 Représentation par un tableau de liste

1.1



1.2

```
1  constante l<Entier> = 5;
2  type FileAvecPriorite[T]: tableau [1 a P] de <Liste[T]>;
3  type Liste[T]: pointeur sur <Cellule[T]>;
4  type Cellule[T]: enregistrement
5    element <T>,
6    suivant <Liste[T]>;
```

Listing 1 – Représentation physique

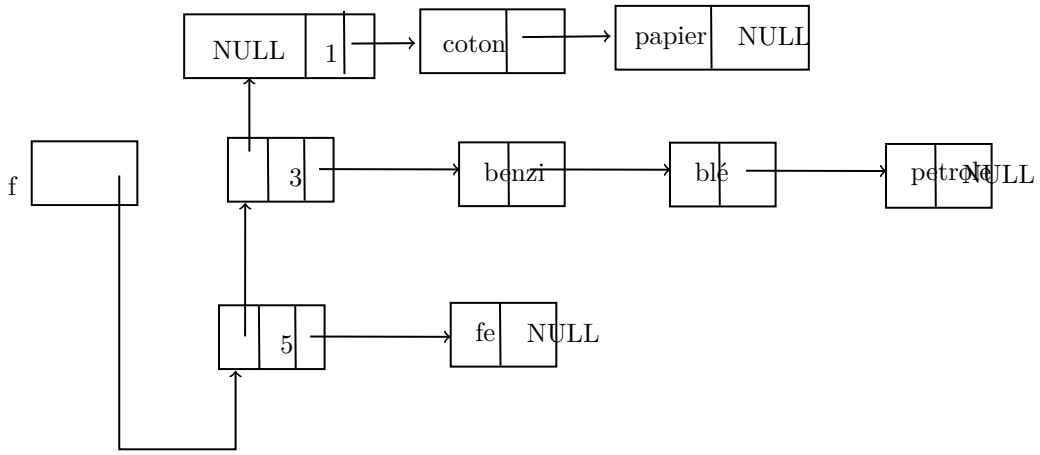
1.3

```
1  procedure creerFile(sortie f <FileAvecPriorite[T]>)
2  glossaire
3    i <Entier>;
4  debut
5    i <- 1;
6    tantque i <= p faire
7      f[i] <- NULL;
8      i <- i+1;
9    fin tantque;
10 fin
11
```

```
12 procedure defiler(maj file <FileAvecPriorite[T]>, sortie element <T>)
13   declenche fileVide;
14 glossaire
15   i <Entier>;
16   trouvé <Booleen>;
17   courant <Liste[T]>;
18 debut
19   trouvé <- faux;
20   i <- p;
21   tantque (i > 0) et (non trouvé) faire
22     si l[i] /= NULL alors
23       trouvé <- vrai;
24       courant <- file[i];
25       file[i] <- file[i]↑.suivant;
26       element <- file[i]↑.element;
27       liberer(courant);
28     fin si;
29     i <- i - 1;
30   fin tantque;
31   si non trouvé alors
32     declencher fileVide;
33   fin si;
34 fin
35
36 procedure enfiler(maj file <FileAvecPriorite[T]>, entree ele <T>, entree
       propriete <Entier>)
37   declenche filePleine
38 glossaire
39   nouveau <Liste[T]>;
40   courant <Liste[T];
41 debut
42   allouer(nouveau);
43   si nouveau = NULL alors
44     declencher filePleine;
45   fin si;
46   nouveau↑.element <- ele;
47   nouveau↑.suivant <- NULL;
48   si file[propriete] = NULL alors
49     file[propriete] <- nouveau;
50   sinon
51     courant <- file[propriete];
52     tantque courant↑.suivant /= NULL faire
53       courant <- courant↑.suivant;
54     fin tantque;
55     courant↑.suivant <- nouveau;
56   fin si;
57 fin
```

2 Représentation par une liste ordonnée de listes

2.1



2.2

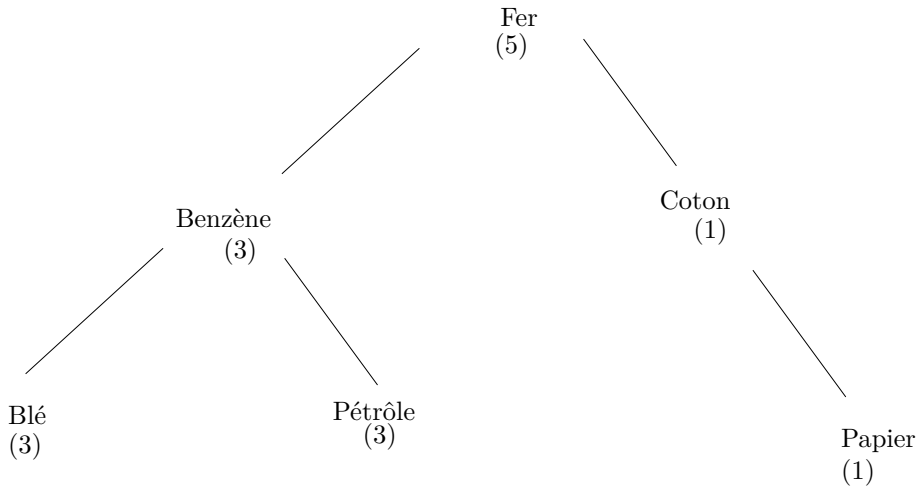
```
type FileAvecPriorite[T]: pointeur sur <Priorité>;
type Priorité[T] : enregistrement
  suivant <FileAvecPriorite[T]>,
  niveau <Entier>,
  elements <Liste[T]>;

type Element[T] : enregistrement
  valeur<T>,
  suivant <Liste[T]>;

type Liste[T] : pointeur sur <Element[T]>;
```

2.3

```
procedure creerFile(sortie file <FileAvecPriorite[T]>)
debut
  file <- NULL;
fin

procedure defiler(maj file <FileAvecPriorite[T]>, sortie ele <T>)
  declencher fileVide
debut
  si file = NULL alors
    declencher fileVide;
  fin si;

  courant <- file;
  ele <- courant↑.elements;
  courant↑.elements <- element↑.suivant;
  liberer(element);

  si courant↑.elements = NULL alors
    file <- courant↑.suivant;
    liberer(courant);
  fin si;
fin
```

3 Representation par un arbre binaire partiellement ordonnée

3.1



3.2

Pour la file f du sujet :

	Valeur	Priorité
1	fer	5
2	benzène	3
3	coton	1
4	blé	3
5	pétrôle	3
6	papier	1
7	///	///
N	///	///

f

élément

6

nbElements

```
constante m <Entier> = 100;
type Cellule[T] : enregistrement
  valeur <T>,
  priorite <Entier>;

type TabNoeuds[T] : tableau de [1..M] de <Cellule[T]>;

type FileAvecPriorite[T]: enregistrement
  valeurs <TabNoeuds[T]>,
  nbElement <Entier>;
```

Listing 2 – Définition