table des matières

Université Bordeaux 1

Licence Semestre 3 - Algorithmes et structures de données 1

Dernière mise à jour effectuée le 1 Septembre 2013

Piles et Files

1. Définitions

Les piles et les files sont des containeurs dans lesquels l'accès ne peut se faire qu'à un objet particulier.

> **Définition 3.1.** Dans une pile, l'objet supprimé est le dernier inséré (LIFO, Last-In, First-Out).
>
> **Définition 3.2.** Dans une file, l'élément supprimé est le plus ancien dans la file (FIFO, First-In, First-Out).

On écrira pour déclarer des variables :

```
type_pile=pile de objet;
type_file=file de objet;
```

2. Primitives de pile

Une pile est défini par les opérations suivantes :

- accès

```
fonction valeur(val P:pile de objet):objet;
fonction pileVide(val P:pile de objet):booléen;
```

- modification

```
fonction créerPile(ref P:pile de objet):vide;
fonction empiler(ref P:pile de objet;
                 val x:objet):vide;
fonction dépiler (ref P:pile de objet):vide;
fonction detruirePile(ref P:pile de objet):vide;
```

Construire une listeSC inverse à partir d'une listeSC

```
fonction listeInverse(val L:listeSC de objet):listeSC de objet;
    var P:pile de objet;
    var LR:liste de objet;
```

```
    début
      creerListe(LR);
      creerPile(P);
      debutListe(L);
      tant que !finListe(L) faire
        empiler(P,valeur(L));
        suivant(L);
      fintantque
      insererEnTete(LR,valeur(P));
      depiler(P);
      tant que non(pileVide(P)) faire
        insererApres(LR,valeur(P));
        suivant(LR);
        depiler(P);
      fintantque;
      detruirePile(P)
      retourner(LR);
    fin
finfonction
```

On peut peut-être se dispenser de pile ! A vous de trouver pouqruoi et comment.

Compter le nombre d'élément d'une pile d'entiers

```
fonction comptePile(val P:pile de objet): entier;
    var compt:entier;
    var PS:pile de objet;
    var v:objet;
    début
      compt=0;
      creerPile(PS);
      tant que non(pileVide(P)) faire
        compt=compt+1;
        v=valeur(P);
        depiler(P);
        empiler(PS,v);
      fintantque;
      tant que non(pileVide(PS)) faire
        v=valeur(PS);
        depiler(PS);
        empiler(P,v);
      fintantque;
      detruirePile(PS);
      retourner(compt);
    fin
finfonction
```

---

3. Primitives de File

Une file est défini par les opérations suivantes :

- accès

```
fonction valeur(val F:file de objet):objet;
fonction fileVide(val F:file de objet):booléen;
```

- modification

```
fonction créerFile(ref F:file de objet):vide;
fonction enfiler(ref F:file de objet;
                 val v:objet):vide;
fonction défiler(ref F:file de objet):vide;
fonction detruireFile(ref F:file de objet):vide;
```

Compter le nombre d'élément d'une file d'entiers non nuls

```
fonction compteFile(ref F:file de entier): entier;
  var v,compt:entier;
  début
    compt=0;
    enfiler(F,0);
    tant que valeur(F)!=0 faire
      compt=compt+1;
      v=valeur(F);
      defiler(F);
      enfiler(F,v);
    fintantque;
    defiler(F);
    retourner(compt);
  fin
finfonction
```

inverser une file d'entiers non nuls

```
fonction inverserFile(ref F:file de entier): file d'entier;
  var P:pile de entier;
  var FS:file d'entier;
  début
    creerPile(P);
    creerFile(FS);
    enfiler(F,0);
    tant que valeur(F)!=0 faire
      v=valeur(F);
      defiler(F);
      enfiler(F,v);
      empiler(P,v);
    fintantque
    defiler(F);
    tant que non(pileVide(P)) faire
      v=valeur(P);
      enfiler(FS,v);
      depiler(P);
    fintantque;
    detruirePile(P);
    retourner(FS);
  fin
finfonction
```

---

4. Implémentations de pile

Implémentation dans un tableau

Chaque objet de la pile est un élémént du tableau. On doit de plus avoir un champs qui permet d'accéder au sommet de pile. On a donc

```
pile d'objet=structure
                taille:entier;
                sommet:entier;
                pile:tableau[1..taille] d'objets;
              finstructure;
```

- accès

```
fonction valeur(ref P:pile de objet):objet;
  début
    retourner(P.pile[P.sommet]);
  fin
finfonction

fonction pileVide(ref P:pile de objet):booléen;
```

```
  début
    retourner(P.sommet==0);
  fin
finfonction
```

- modification

```
fonction empiler(ref P:pile de objet; x:objet):booleen;
  /* l'espace de stockage peut être saturé */
  début
    si P.sommet==P.taille alors
      retourner(FAUX)
    sinon
      P.sommet=P.sommet+1;
      P.pile[P.sommet]=x;
      retourner(VRAI)
    finsi
  fin
finfonction

fonction dépiler(ref P:pile de objet):vide;
  début
    P.sommet=P.sommet-1;
  fin
finfonction

fonction créerPile(ref P:pile de objet):pile de objet;
  début
    P.sommet=0;
  fin
finfonction
```

Implémentation par une listeSC

Chaque objet de la pile est un objet de la listeSC.

```
pile d'objet=listeSC de objet;
```

- accès

```
fonction valeur(ref P:pile de objet):objet;
  début
    débutListe(P);
    retourner(valeur(P));
  fin
finfonction

fonction pileVide(ref P:pile de objet):booléen;
  début
    retourner(listeVide(P));
  fin
finfonction
```

- modification

```
fonction empiler(ref P:pile de objet; x:objet):vide;
  début
    insérerEnTete(P,x)
  fin
finfonction

fonction dépiler(ref P:pile de objet):vide;
  début
    supprimerEnTete(P);
  fin
```

```
finfonction

fonction créerPile(ref P:pile de objet):vide;
  début
    créerListe(P);
  fin
finfonction
```

---

5. Implémentations de file

Implémentation dans un tableau

Chaque objet de la file est un élément du tableau. On utilise le tableau de manière circulaire avec un pointeur donnant le premier et un autre donnant le dernier.

```
file d'objet=structure
                taille:entier;
                premier:entier;
                dernier : entier;
                plein:booléen;
                file:tableau[0..taille-1] d'objets;
              finstructure;
```

- accès

```
fonction valeur(ref F:file de objet):objet;
  début
    retourner(F.file[F.premier]);
  fin
finfonction

fonction fileVide(ref F:file de objet):booléen;
  début
    retourner(F.premier==F.dernier & non(F.plein));
  fin
finfonction
```

- modification

```
fonction enfiler(ref F:file de objet; x:objet):booleen;
  début
    si F.plein alors
      retourner(FAUX)
    sinon
      F.file[F.dernier]=x;
      F.dernier=(F.dernier+1) mod F.taille;
      F.plein=F.dernier==F.premier;
      retourner(VRAI)
    finsi
  fin
finfonction

fonction défiler(ref F:file de objet):vide;
  début
    F.premier=(F.premier+1) mod F.taille;
    F.plein=Faux
  fin
finfonction

fonction créerFile(ref F:file de objet):file de objet;
  début
    F.premier= 0;
    F.dernier= 0;
    F.plein=FAUX;
```

```
  fin
finfonction
```

Implémentation par une listeDC

Chaque objet de la file est un objet de la listeDC car Il faut un accès au dernier.

```
file d'objet=listeDC de objet;
```

- accès

```
fonction valeur(ref F:file de objet):objet;
  début
    débutListe(F);
    retourner(valeur(F));
  fin
finfonction

fonction fileVide(ref F:file de objet):booléen;
  début
    retourner(listeVide(F));
  fin
finfonction
```

- modification

```
fonction enfiler(ref F:file de objet; x:objet):vide;
  début
    dernier(F);
    insérerAprès(F,x);
  fin
finfonction

fonction défiler(ref F:file de objet):vide;
  début
    supprimerEnTete(F);
  fin
finfonction

fonction créerFile(ref F:file de objet):vide;
  début
    créerListe(F);
  fin
finfonction
```