Université de M'hamad Bougara de Boumerdès
Faculté des Sciences
Deuxième Année Licence
Recherche Opérationnelle

Département de Mathématiques
Responsable du Module:
Mr. M. BEZOUI

Durée 1h30'

Semestre 03

Correction de l'Examen Final de Structures de Données

Exercice 1 (6point=1+1+1+1+2).
Soient les deux fonctions *mystere*1 et *mystere*2:

1. Quels sont les résultats de:
   *mystere*1(2), *mystere*1(3), *mystere*1(4)?
   mystere1(2)=3;mystere1(3)=∞;mystere1(4)=3.5;

2. Pour quelles valeurs de n, la fonction
   *mystere*1 ne s'arrête pas.
   La fonction mystere1 ne s'arrête
   pas pour n impair ou négatif.

3. Quel sont les résultats de:
   *mystere*2(4, 6), *mystere*2(6, 9)?
   *mystere*2(4, 6) = 2, *mystere*2(6, 9) = 3.

4. Déduire le rôle de la fonction *mystere*2.
   mystere2 est une fonction qui calcule le pgcd de a et b.

5. Réaliser une fonction itérative qui
   a le même rôle que *mystere*2.

```
fonction mystere1(n:entier):réel
début
si n=0 alors
  renvoyer(2);
sinon
renvoyer(1/2*mystere1(n-2)+2);
fin si
fin;

fonction mystere2(a,b:entier):entier;
Var
résultat: entier;
debut
Si(a = b) Alors
  resultat ← a;
Sinon Si(a > b) Alors
  resultat← mystere2(a - b, b);
  Sinon
    resultat← mystere2(a, b - a);
FinSi FinSi
  Renvoyer resultat;
Fin;
```

Algorithme 1 exo1R5

```
fonction mystere2(a,b:entier):entier
debut
tantque a * b <> 0 faire
  si a>b alors
    a ← a − b
  sinon
    b ← b − a
  finsi
fin tantque
si a=0 alors
  renvoyer(b);
sinon
  renvoyer(a);
finsi
fin;
```

Exercice 2 (8points=1.5+1.5+1.5+1.5+1.5+2).
Algorithme *exo2*
type *Liste=enregistrement*
info:entier;

suivant: ↑ *Liste;*
fin_ enregistrement
var *L:*↑*Liste;*

1. **Réalisez une fonction qui renvoie le produit des éléments de la liste L.**

Algorithme 2 Exo1R1

```
fonction Prod(L:↑ liste): entier;
var P:↑ liste; produit: entier;
debut
P ← L;
produit← 1;
tantque P <> NIL faire
  produit← produit*P↑ .info;
  P ← P ↑ .suivant;
fin tantque
renvoyer(produit);
fin;
```

2. **Réalisez une fonction qui renvoie la moyenne des éléments de la liste L.**

Algorithme 3 Exo2R2

```
fonction produit(L:↑ liste): réel;
var P:↑liste; somme, nbr:entier;
debut
P ← L;
somme← 0; nbr←0;
tantque P <> NIL faire
  somme← somme+P↑ .info;
  nbr←nbr+1;
  P ← P ↑ .suivant;
fin tantque
renvoyer(somme/nbr);
fin;
```

3. **Réalisez une procédure qui affiche les éléments pairs de la liste L.**

Algorithme 4 Exo2R3

```
procédure pair(L:↑ liste)
var P:↑liste;
debut
P ← L;
tantque P <> NIL faire
  si mod(P↑.info,2)=0 alors
    écrire(P↑.info);
  finsi
  P ← P ↑ .suivant;
fin tantque
fin;
```

4. *Vérifiez si les éléments de la liste L sont tous* ***distincts*** *(différents deux à deux).*

Algorithme 5 Exo2R4

```
fonction test(L:↑ liste):booléen
var P1,P2:↑liste; t:booléen;
debut
si L = NIL alors
  renvoyer(vrai);
sinon
  P1 ← L;P2 ← P ↑ .suivant;t← vrai;
  tantque P1 ↑ .suivant <> NIL faire
    tantque P2 <> NIL faire
      si P1 ↑ .info == P2 ↑ .info alors
        renvoyer(vrai);
      finsi
      P1 ← P1 ↑ .suivant;P2 ← P2 ↑ .suivant;
    fin tantque
  fin tantque
finsi
renvoyer(t);
fin;
```

Algorithme 6 Exo2R5

```
fonction max(L:↑ liste,max:entier):entier
{Il faut vérifier que la liste n'est pas initialement vide dans l'algo principal}
{max est initialisé à L ↑ .info, l'information de la tête}
var P:↑liste;
debut
si L ↑ .suivant=NIL alors
  renvoyer(max);
sinon
  si L ↑ .suivant ↑ .info > max alors
    max ← L ↑ .suivant ↑ .info
  finsi
  renvoyer(L ↑ .suivant,max);
finsi
fin;
```

5. **Réalisez une fonction récursive qui renvoi le maximum de la liste L.**

Les exercice 3 et 4 sont au choix

Exercice 3 (6points=2+2+2)**.** Soient P une pile d'entiers, trié selon un ordre croissant et F est une file d'entier vide. Réalisez une fonction qui:

1. **Transfert les éléments de la Pile P vers la File F, en gardant le même ordre (croissant).**

2. **Insert un entier X dans la file F, tout en gardant l'ordre des éléments de F.**

3. **Supprime tous les éléments impaires de P.**

Exercice 4 (6points=3+3)**.** On considère deux liste chaînées L1 et L2 (définies comme dans l'exercice 2) contenant des entiers et triées par ordre croissant.

1. Créez une troisième liste L3, sans détruire les précédentes, et vérifiant la propriété suivante: L3 contient, dans l'ordre croissant, les entiers communs à L1 et L2.

2. Créez une quatrième liste L4, qui contient les entiers de L1 et L2 sans répétition.

Pour la fonction il suffit d'éliminer les doublants de la liste L3.

Algorithme 7 Exo3R1

```
fonction transfert(P: Pile): File
var P1: Pile; F: File; val: entier;
debut
créer_pile(P1); créer_file(F);
tantque non_pile_vide(P) faire
  depiler(P,val);
  empiler(P1,val);
fin tantque
tantque non_pile_vide(P1) faire
  depiler(P1,val);
  empiler(P,val);
  enfiler(F,val);
fin tantque
renvoyer(F);
fin;
```

Algorithme 8 Exo3R2

```
fonction insert(F: File, X:entier): File
var F1,F: File; val: entier; trouve: booléen;
debut
créer_file(F1);
tantque non_file_vide(F) faire
  defiler(F,val);
  emfiler(F1,val);
fin tantque
trouve← faux;
tantque non_file_vide(F1) faire
  defiler(F1,val);
  si X<val et trouve=faux alors
    enfiler(F,X);
    trouve← vrai;
  finsi
  empiler(F,val);
fin tantque
renvoyer(F);
fin;
```

Algorithme 9 Exo3R3

```
fonction insert(P: Pile): Pile
var P1,P: Pile; val: entier;
debut
créer_pile(P1);
tantque non_pile_vide(P) faire
  depiler(P,val);
  empiler(P1,val);
fin tantque
tantque non_file_vide(F1) faire
  defiler(F1,val);
  si mod(val,2)=0 alors
    enpiler(P,val);
  finsi
fin tantque
renvoyer(P);
fin;
```

Algorithme 10 Exo4R1

Exercice 5. fonction fusion1(L1,L2:↑Liste):↑Liste

```
var P1,P1,P33,L3:↑ liste;
debut
P1 ← L1; P2 ← L2;
nouveau(L3);P3 ← L3;
tantque P1 <> NIL ou P2 <> NIL faire
  si P1↑.info <= P2↑.info alors
    L3↑.info ← P1↑.info;
    P1 ← P1↑.suivant;
  sinon
    L3↑.info ← P2↑.info;
    P1 ← P1↑.suivant;
  finsi
  nouveau(P3↑.suivant);
  P33 ← P3;
  P3 ← P3↑.suivant;
fin tantque
si P1=NIL alors
  P3↑.suivant ← P2;
sinon
  P3↑.suivant ← P1;
finsi
renvoyer(L3);
fin;
```