**Université de Bejaia**
**Faculté des sciences exactes**
**Département M/MI**
**1ère année MI**
**Module PSSD**

**Nom :**
**Prénom :**
**Groupe :**
**Emargement:**
**11/06/2015, Durée : 2 heures**

**Exercice 01 :**

```
algorithme exo_01 ;
    Type
        etud = enregistrement
                ide : entier ;
                nom, pre : chaine[25];
                moy : reel;
               Fin ;
        tabe=tableau[1..200] de etud ;
        date = enregistrement
                jour : 1..31 ;
                mois : 1..12;
                ann : entier;
               Fin ;

        stag = enregistrement
               ides : entier ;
               code : chaine[10] ;
               lieu :   chaine[30]
               datst : date;
               nbrp: entier;
               Fin ;
        tabs=tableau[1..200] de stag ;

Var     tae :tabe ; tas : tabs;ne1, id1 :entier ; ent1 : chaine[50]
    Procedure     saisir (var Te: tabe ; var Ts: tabs ; ne: entier) ;
        Var             i: entier ;
        Debut
        |    Pour  i  allant de   1  jusqu'à   nefaire
        |    | avec Te[i]faire
        |    | |lire ( nom, pre,moy );
        |    | |ide ←i ;
        |    | finavec ;
        |    | avec Ts[i] faire
        |    | |lire ( code, lieu,nbrp, datst.jour ,datst.mois , datst.ann );
        |    | |ide ←i ;
        |    | finavec ;
        |    |finpour ;
        |fin ;
    Procedure     existe1(Te:tabe ; ne :entier ;id :entier);
        Var  j :entier ; trouv:booleen ;
        Debut
        |    j← 1;
        |    trouv ← faux  ;
        |    tantque   ((j≤ne) et non trouv )  faire
        |    |        avec Te[j] faire
        |    |        | Si   (id=ide)  alors
        |    |        | | trouv← vrai ;ecrire(nom, pre) ;
        |    |        | |sinon
```

0,25 0,25 0,25 0,25 0,25 0,25 0,25 0,25 0,25 0,25 0,25 0,25 0,25 0.25

1 pts 1 pts 1 pts

```
|     |       |       j ← j+1 ;
|     |       |finsi ;
|     |fintantque ;
|fin ;
```

```
Procedure      afficher1 ( Ts: tabs ;ne: entier ; Te :tabe ;ent:chaine[10]) ;
    Var              i: entier ;
    Debut
    |    Pour  i  allant de    1   jusqu'à    ne faire
    |    |  avec Ts[i] faire
    |    | |    si(ent=lieu)alors
    |    | |    |existe1(Te, ne, i);
    |    | |    |finsi ;
    |    | |finavec ;
    |    |finpour ;
    |fin ;
```

0,5 pts

0,25

0,25

```
Procedure  existe2(Ts:tabs ; ne :entier ; id :entier);
    Var  j :entier ; trouve:booleen ;
    Debut
    |    trouv ← faux  ;   j← 1;
    |    tantque   ((j≤ne) et non trouv )  faire
    |    |        avec Ts[j] faire
    |    |        |  Si   (id=ides)  alors
    |    |        |  |trouv← vrai ; ecrire(datst.jour, datst.mois, datst.ann) ;
    |    |        |sinon
    |    |        |       j← j+1 ;
    |    |        |finsi ;
    |    |fintantque ;
    |fin ;
```

On a fait le barème de existe1,

```
Procedure      afficher2 ( Ts: tabs ;ne: entier ; Te :tabe ;id :entier) ;
    Var              i: entier ;
    Debut
    |    i← 1;
    |    trouv ← faux  ;
    |    tantque   ((i≤ne) et non trouv )  faire
    |    |        avec Te[i] faire
    |    |        |  Si   (id=ide)  alors
    |    |        |  |    trouv← vrai ;
    |    |        |  |sinon
    |    |        |  |    i ← i+1 ;
    |    |        |  |finsi ; finavec ;
    |    |fintantque ;
    |    si(trouve=vrai)alors
    |    |    existe2(Ts, ne, i);
    |    |sinon
    |    |  ecrire('ident n''existe pas') ;
    |    |finsi;
    |fin ;
```

0,25

0,25

0,25

0,25

0,25

1,5 pts

```
Procedure      ajouter (var Te: tabe ; ne: entier ; var Ts : tabs) ;
    Var
    Debut
    |    avec Te[ne+1] faire
    |    | lire (moy, nom,pre);
    |    | ide ←ne+1 ;
    |    |finavec ;
    |    avec Ts[ne+1] faire
    |    | lire (code, lieu,datst.jour, datst.mois, datst.ann,nbrp);
    |    | ides ←ne+1 ;
    |    |finavec ;
    |fin ;
```

0,25 0,25 0.25 0,25 1 pts

```
Debut
|       repeter
|       |    lire(ne1) ;
|       |jusqu'à (ne1≥1) et (ne1<200) ;
|      lire( ent1, id1);
|     saisir (tae,tas, ne1) ;
|     afficher1 ( tas, ne1,tae,ent1) ;
|    afficher2 ( tas, ne1,tae,id1) ;
|    ajouter (tae, ne1 ,tas) ;
| fin .
```

0.25 0.25 0,25 0,25 1 pts

**Exercice 2:**

```
algorithme exo_02;
Type    pointeur = ↑med  ;
        med = enregistrement
                lib: chaine[25] ;
                nbb:entier
                pri: reel ;
                  Suivant : pointeur ;
              Fin ;
    Var     L :pointeur ;
        Procedure      creation (var liste : pointeur) ;
            Var              P   : pointeur ; rep :chaine[3] ;
            Debut
            |     liste← Nil ;
            |     repeter
            |     |         allouer (P)
            |     |   avec P↑faire
            |     |   |lire (lib,nbb,pri)  ;
            |     |   |suivant ←liste ;
            |     |   finavec ;
            |     |         liste←  P ;
            |     |    ecrire('voulez vous rajouter un médicament ?oui/non') ;
            |     |    lire (rep) ;
            |     |jusqu'à (rep='non');
            |fin ;

        Procedure     vendre (var liste : pointeur, lib1: chaine[25],Nbboite :entier) ;
            Var              P,pp   : pointeur ;
            Debut
            |     P← liste;
            |     tantque    (P<> Nil) et (lib1≠P↑.lib)faire
            |     |         pp ←P ;   P ← P↑.suivant ;
            |     |fintantque ;
            |     Si    (P= Nil )  alors
            |     | ecrire ('le medicament n''existe pas')   ;
            |     |sinon
            |     |       si   (Nbboite<P↑.nbb) alors   ecrire ('quantité insuffisante')
            |     |       |sinon
            |     |       |     P↑.nbb ← P↑.nbb - Nbboite ;
            |     |       |     si (P↑.nbb = 0) alors
            |     |       |     |  si (P = liste) alors
            |     |       |     | |     liste ← liste↑.suivant ;   liberer(P) ;
            |     |       |     | |sinon
            |     |       |     | |     pp↑.suivant ← P↑.suivant ;   liberer(P) ;
            |     |       |     | |finsi ;
            |     |       |     |finsi ;
            |     |       |finsi ;
            |     |finsi ;
            |fin ;
```

0,5 pts

0,25

0,25

0,25

0,5 pts

0,25

0,25

0.5

0,25

0,25

0,25

0,25

1,75 pts

```
Procedure      acheter (var liste : pointeur, lib1: chaine[25] ; Nbboite :entier ; pri1 : reel) ;
    Var              P,pp   : pointeur ;
    Debut
    |     P← liste;
    |     tantque    (P<> Nil) et (lib1≠P↑.lib) faire
    |     |         pp ← P ;
    |     |         P ← P↑.suivant ;
    |     |fintantque ;
    |         Si    (P<>  Nil )  alors
    |         | P↑.nbb ← P↑.nbb + Nbboite ,P↑.pri ← pri1 ;
    |         |sinon
    |         |   allouer (Q)   ;
    |         |   avecQ↑faire
    |         |   |lib ← lib1 ; nbb ← Nbboite ; pri ← pri1 ;
    |         |   |suivant ←Nil ;
    |         |   |finavec ;
    |         |         pp↑.suivant ←Q ;
    |         |finsi ;
    |fin ;
```

0,5 0,5 0,5 0,25

1,75 pts

```
Fonction    PriStock ( liste : pointeur) : reel ;
    Var              P,pp   : pointeur ; S :reel ;
    Debut
    |     P← liste;    S← 0;
    |     tantque    (P<> Nil)  faire
    |     |         S ← S+ (P↑.nbb * P↑.pri) ;
    |     |         P ← P↑.suivant ;
    |     |fintantque ;
    |      PriStock ← S ;
    |fin ;
```

0.25 0,25 0,25

0,75 pts

```
Debut
|     creation (L) ;    lire (lib1,NBboite) ;
|     vendre (L, lib1,NBboite) ;   lire (lib1,NBboite,pri1)
|     acheter(L, lib1,NBboite,pri1) ;
|     ecrire ('le prix total des médicaments est=', PriStock ( L)) ;
|fin.
```

0,25 0,25 0,25

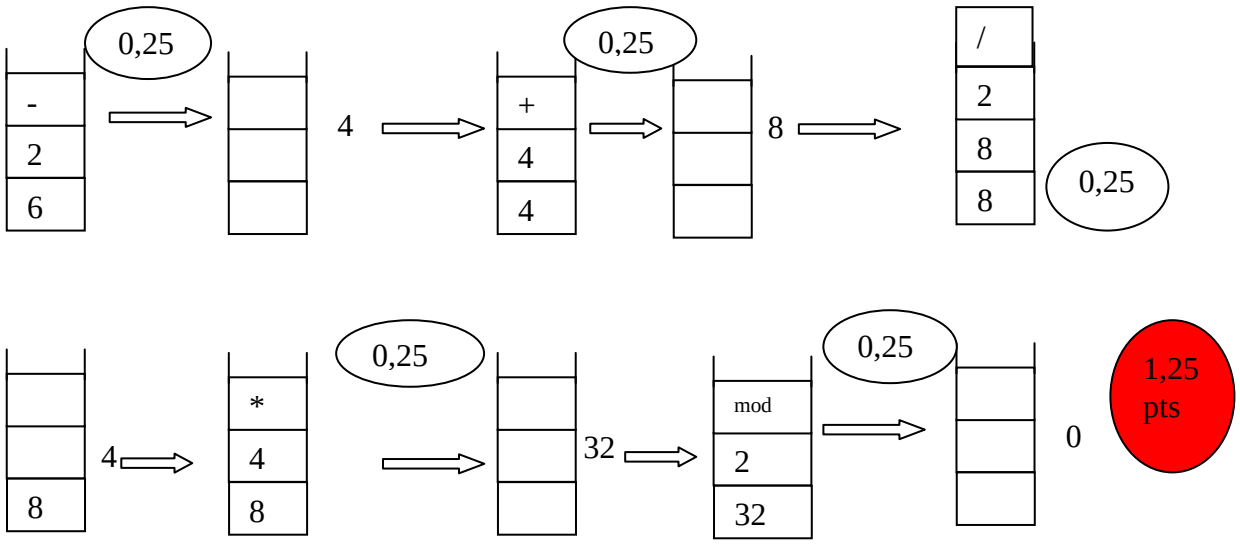
0,75 pts

**Exercice 3:**

**E = (((6-2) + 4) * (8/2)) mod 2**

1. Expression postfixée : E1= 6 2 – 4 + 8 2 / * 2 mod
2. Utilisation de la pile

0,75 pts




**algorithme** verifier ;

**type** pile= ↑ nœud ;

**type** nœud = **enregistrement**

| info :entier ;

| suivant :pile ;

|**fin ;**

**Var** A,B,B1 :pile ; bool1 : booleen ;

**fonction** calcul_taille (p:pile): entier ;

**var** c:entier; p1:pile;

**debut**

| **c** ← 0 ;

| **tantque non** (pile_vide(p)) **faire**

| | c ← c+1 ; 0,25

| | empiler (p1,sommet(p)) ; dépiler (p) ;

| |**fintantque ;**

| calcul_taille ← c ; 0.25

| **tantque non** (pile_vide(p1)) **faire**

| | empiler (p,sommet(p1)) ; dépiler (p1) ; 0.25

| |**fintantque ;**

|**fin ;**

0,75 pts

```
debut
|       bool1 ← faux ;
|   si (calcul_taille(A) > calcul_taille (B))   alors
|   |         ecrire (‘ A n’’est pas une partie de B’) ;
|   |sinon
|   |   repeter
|   |   |   tantque   (non (pile_vide(B)) et (sommet(A)≠sommet(B)) )   faire
|   |   |   |     empiler (B1,sommet(B)); depiler (B) ;
|   |   |   |fintantque ;
|   |   |   Si (pile_vide(B) = vrai) alors   bool1← vrai ;
|   |   |   |sinon
|   |   |   |   depiler (A) ;
|   |   |   |   tantque   (non (pile_vide(B) )   faire
|   |   |   |   |     empiler (B,sommet(B1)); depiler (B1) ;
|   |   |   |   |fintantque ;
|   |   |   |finsi ;
|   |   |jusquà  (pile_vide(A) ou (bool1=vrai) ;
|   |    Si (pile_vide(A) = vrai) alors   ecrire(‘A est une partie de B’) ;
|   |    |sinon
|   |    |   ecrire(‘A n’’est pas une partie de B’) ;
|   |    |finsi ;
|   |finsi ;
|fin.
```

0,5 — 0.25 — 0,25 — 0,25 — 0.25 — 0,25 — 0.25 — 0,25 — 2,25 pts

**Exercice 4:**

```
algorithme     permutation ;
    const taille= 100 ;
    type tab =  tableau  [1..taille] de entier ;
    var T : tab ; n : entier ;
    procedure permuter (var TT:tab; deb,fin: entier);
        var x: entier;
        debut
        |     si( deb<fin ) alors
        |     |  x←TT[deb];
        |     | TT[deb] ←TT[deb+1];
        |     | TT[deb+1]←x;
        |     | permuter(TT,deb+2,fin);
        |     |finsi ;
        |fin ;
```

0,25 — 0,75 — 0,75 — 0,75 — 2,5 pts

**debut** (pp) 0,25
| n ←1 ;
| permuter (T,n, Taille) ; 0,25
|**fin**.

0,5 pts