```
{ Algorithme A* appliqué sur le jeu de Taquin}
   SOIENT
       G UN GRAPHE DE VECTEUR ( 3 , 3 ) DE ENTIERS ;
       _noeud , N1 , N2 DES NOEUDS ( G ) ;
       Lo , Lf DES LISTES DE NOEUDS ( G ) ;
       { (g, h) }
       Valo : LISTE DE ( ENTIER , ENTIER ) ;
       V : ( ENTIER , ENTIER ) ;
       Gden2 : ENTIER ;
       H UNE FONCTION ( ENTIER ) ;
       Generer_suc UNE ACTION ;
       Nlo , Nlf DES ENTIERS ;
       Continue UN BOOLEEN ;
       L UNE LISTE DE NOEUD ( G ) ;
       Conf_fin : VECTEUR ( 3 , 3 ) DE ENTIERS ;
       Exist_dans_lo , Exist_dans_lf , Egal DES FONCTIONS ( BOOLEENS ) ;
       Rang_plus_petit UNE FONCTION ( ENTIER ) ;
       I UN ENTIER ;

   DEBUT
       INIT_VECT ( Conf_fin , [ 1 , 2 , 3 , 8 , 0 , 4 , 7 , 6 , 5 ] ) ;
       CREERLISTE ( Lo ) ;
       CREERLISTE ( Valo ) ;
       Nlo := 0 ;
       CREERLISTE ( Lf ) ;
       Nlf := 0 ;
       CREERNOEUD ( G , N1 ) ;
       INIT_VECT ( N1 , [ 2 , 3 , 4 , 5 , 6 , 8 , 7 , 1 , 0 ] ) ;
       Nlo := Nlo + 1 ;
       INSERER ( Lo , N1 , Nlo ) ;
       INIT_STRUCT ( V , [ 0 , 0 ] ) ;
       INSERER ( Valo , V , Nlo ) ;
       Continue := VRAI ;
       TQ NON FINLISTE ( Lo ) ET Continue
      {retirer un noeud de Lo}
           N_IEME ( Lo , Rang_plus_petit ( Valo ) , N2 ) ;
           ECRIRE ( 'boucle' ) ;
           N_IEME ( Valo , Rang_plus_petit ( Valo ) , V ) ;
           Gden2 := STRUCT ( V , 1 ) ;
           SUPPRIMER ( Lo , Rang_plus_petit ( Valo ) ) ;
           SUPPRIMER ( Valo , Rang_plus_petit ( Valo ) ) ;
          {est-ce un r,sulat}
           SI Egal ( N2 , Conf_fin )
               Continue := FAUX
           SINON
          {Lf <----- n}
               Nlf := Nlf + 1 ;
               INSERER ( Lf , N2 , Nlf ) ;
              {g,n,rer les successeur de n2}
```

```
            APPEL Generer_suc ( N2 , L ) ;
            POUR I := 1 , NBRLISTE ( L )
                N_IEME ( L , I , _noeud ) ;
                SI NON Exist_dans_lo ( _noeud ) ET NON Exist_dans_lf ( _noeud )
                    Nlo := Nlo + 1 ;
                    INSERER ( Lo , _noeud , Nlo ) ;
                    INIT_STRUCT ( V , [ Gden2 + 1 , H ( N2 ) ] ) ;
                    INSERER ( Valo , V , Nlo ) ;
                    CREERARC ( _noeud , N2 ) ;

                FSI ;
                SI Exist_dans_lo ( _noeud ) OU Exist_dans_lf ( _noeud )

                FSI
            FPOUR
        FSI
    FTQ
FIN
ACTION Generer_suc ( N1 , L )
SOIT
    L UNE LISTE DE NOEUD ( G ) ;
    N , X , Y DES ENTIERS ;
    N1 , N2 DES NOEUDS ( G ) ;

DEBUT
    CREERLISTE ( L ) ;
    N := 0 ;
    POUR X := 1 , 3
        POUR Y := 1 , 3
            SI ( ( X - 1 ) >= 1 )
                SI ELEMENT ( N1 [ X - 1 , Y ] ) = 0
                    CREERNOEUD ( G , N2 ) ;
                    N2 := N1 ;
                    AFF_ELEMENT ( N2 [ X - 1 , Y ] , ELEMENT ( N2 [ X , Y ] ) ) ;
                    AFF_ELEMENT ( N2 [ X , Y ] , 0 ) ;
                    N := N + 1 ;
                    INSERER ( L , N2 , N ) ;

                FSI
            FSI ;
            SI ( ( X + 1 ) <= 3 )
                SI ELEMENT ( N1 [ X + 1 , Y ] ) = 0
                    CREERNOEUD ( G , N2 ) ;
                    N2 := N1 ;
                    AFF_ELEMENT ( N2 [ X + 1 , Y ] , ELEMENT ( N2 [ X , Y ] ) ) ;
                    AFF_ELEMENT ( N2 [ X + 1 , Y ] , 0 ) ;
                    N := N + 1 ;
                    INSERER ( L , N2 , N ) ;
```

```
                FSI
            FSI ;
            SI ( ( Y - 1 ) >= 1 )
                SI ( ELEMENT ( N1 [ X , Y - 1 ] ) = 0 )
                    CREERNOEUD ( G , N2 ) ;
                    N2 := N1 ;
                    AFF_ELEMENT ( N2 [ X , Y - 1 ] , ELEMENT ( N2 [ X , Y ] ) ) ;
                    AFF_ELEMENT ( N2 [ X , Y - 1 ] , 0 ) ;
                    N := N + 1 ;
                    INSERER ( L , N2 , N ) ;

                FSI
            FSI ;
            SI ( ( Y + 1 ) <= 3 )
                SI ( ELEMENT ( N1 [ X , Y + 1 ] ) = 0 )
                    CREERNOEUD ( G , N2 ) ;
                    N2 := N1 ;
                    AFF_ELEMENT ( N2 [ X , Y + 1 ] , ELEMENT ( N2 [ X , Y ] ) ) ;
                    AFF_ELEMENT ( N2 [ X , Y + 1 ] , 0 ) ;
                    N := N + 1 ;
                    INSERER ( L , N2 , N ) ;

                FSI
            FSI
        FPOUR
    FPOUR
FIN
{Nombre de pièces mal placéss }
FONCTION H ( N ) : ENTIER
SOIT
    N UN NOEUD ( G ) ;
    X , Y DES ENTIERS ;

DEBUT
    H := 0 ;
    POUR X := 1 , 3
        POUR Y := 1 , 3
            SI ELEMENT ( N [ X , Y ] ) <> ELEMENT ( Conf_fin [ X , Y ] )
                H := H + 1
            FSI
        FPOUR
    FPOUR
FIN
FONCTION Rang_plus_petit ( L ) : ENTIER ;
SOIT
    Val UN ( ENTIER , ENTIER ) ;
    L UNE LISTE DE ( ENTIER , ENTIER ) ;
    Plus_petit UN ENTIER ;
    I UN ENTIER ;
```

```
DEBUT
    N_IEME ( Valo , 1 , Val ) ;
    Rang_plus_petit := 1 ;
    Plus_petit := STRUCT ( Val , 1 ) + STRUCT ( Val , 2 ) ;
    I := 1 ;
    TQ NON FINLISTE ( Valo )
        I := I + 1 ;
        SUIVANT ( Valo , Val ) ;
        SI STRUCT ( Val , 1 ) + STRUCT ( Val , 2 ) < Plus_petit
            Plus_petit := STRUCT ( Val , 1 ) + STRUCT ( Val , 2 ) ;
            Rang_plus_petit := I
        FSI ;

    FTQ
FIN
FONCTION Egal ( N1 , N2 ) : BOOLEEN
SOIT
    N1 , N2 DES NOEUDS ( G ) ;
    X , Y DES ENTIERS ;
    Trouv UN BOOLEEN ;

DEBUT
    X := 1 ;
    Trouv := FAUX ;
    TQ ( X <= 3 ) ET ( NON Trouv )
        Y := 1 ;
        TQ ( Y <= 3 ) ET ( NON Trouv )
            SI ELEMENT ( N1 [ X , Y ] ) <> ELEMENT ( N2 [ X , Y ] )
                Trouv := VRAI
            SINON
                Y := Y + 1
            FSI
        FTQ ;
        X := X + 1
    FTQ
FIN
FONCTION Exist_dans_lo ( N ) : BOOLEEN
SOIT
    I UN ENTIER ;
    N , _noeud DES NOEUDS ( G ) ;
    Trouv : BOOLEEN ;

DEBUT
    I := 1 ;
    Trouv := FAUX ;
    SUIVANT ( Lo , _noeud ) ;
    TQ NON Trouv ET ( NON FINLISTE ( Lo ) )
        SI Egal ( N , _noeud )
```

```
            Trouv := VRAI
        SINON
            SUIVANT ( Lo , _noeud )
        FSI
    FTQ ;
    Exist_dans_lo := Trouv
FIN
FONCTION Exist_dans_lf ( N ) : BOOLEEN
SOIT
    I UN ENTIER ;
    N , _noeud DES NOEUDS ( G ) ;
    Trouv : BOOLEEN ;

DEBUT
    I := 1 ;
    Trouv := FAUX ;
    SUIVANT ( Lf , _noeud ) ;
    TQ NON Trouv ET ( NON FINLISTE ( Lf ) )
        SI Egal ( N , _noeud )
            Trouv := VRAI
        SINON
            SUIVANT ( Lf , _noeud )
        FSI
    FTQ ;
    Exist_dans_lf := Trouv
FIN
```