Rapport du TP ALSDD

Les algorithmes écrits :

Définition des structures et types :

```
logement = record
                Type : string
                Street : string
                Surface : integer
                Distance : integer
                ID : integer
location = record
                logementID : integer
                locataireID : integer
                startDate : integer
                endDate : integer
locataire = record
                firstName : string
                lastName : string
                phoneNumber : string
                ID : integer

Node_logement = record
                Data : logement
                Next : pointeur
Node_location = record
                Data : location
                Next : pointeur
Node_locataire = record
                Data : location
                Next : pointeur
```

head_logement, head_location, head_locataire, head_archiveLogement, head_archiveLocation, head_archiveLocataire, head_nmbrCartier, head_plpmoc : pointeur

Les modules :

```
Function loyerLogement(p : pointeur)
Var
   temp : integer
Begin
```

```
    If (cell_value(p).type = “studio”) then
        temp <- 15000 + (cell_value(p).surface - 20) * 800
    Else If (cell_value(p).type = “F2”) then
        temp <- 20000 + (cell_value(p).surface - 20) * 800
    Else If (cell_value(p).type = “F3”) then
        temp <- 30000 + (cell_value(p).surface - 20) * 800
    Else If (cell_value(p).type = “F4”) then
        temp <- 45000 + (cell_value(p).surface - 20) * 800

    loyerLogement <- temp
End

Function loyerLocation(p : pointeur)
Var
    q    : pointeur
    temp : integer
Begin
    q <- head_logement

    While (p <> NULL) do
    Begin
        If (cell_value(p).logementID = cell_value(q).id) then
            temp <- loyerLogement(q)
        Else
            q <- next(q)
    End

    loyerLocation <- temp
End

Procedure addLogement()
Var
    temp : logement
    p    : pointeur
Begin
    write(“Enter the type of logement : “)
    read(temp.type)

    write(“Enter the street of logement : “)
    read(temp.street)

    write(“Enter the surface of logement : “)
    read(temp.surface)
```

```
    write(“Enter the distance from city : “)
    read(temp.distance)

    temp.id <- idLogement()

    allocate(p)
    aff_val(p, temp)
    aff_adr(p, head_logement)
    head_logement <- p
End

Procedure addLocation()
Var
    temp : location
    p    : pointeur
Begin
    write(“Enter the logement ID : “)
    read(temp.logementID)

    write(“Enter the locataire ID : “)
    read(temp.locataireID)

    write(“Enter the start date : “)
    read(temp.startDate)

    write(“Enter the end date : “)
    read(temp.endDate)

    allocate(p)
    aff_val(p, temp)
    aff_adr(p, head_location)
    head_location <- p
End

Procedure addLocataire()
Var
    temp : locataire
    p    : pointeur
Begin
    write(“Enter the first name : “)
    read(temp.firstName)

    write(“Enter the last name : “)
    read(temp.lastName)
```

```
    write(“Enter the phone number : “)
    read(temp.phoneNumber)

    temp.id <- idLocataire()

    allocate(p)
    ass_val(p, temp)
    ass_adr(p, head_locataire)
    head_locataire <- p
End


Procedure deleteLogement()

Var
deleteID : integer
p, q : pointeur

Begin
write("please enter the ID of logement to delete : ")
read(deleteID)

p <- head_logement
q <- NEXT(p)

If ( CELL_VALUE(head_logement).id = deleteID ) then
Begin
    ASS_ADR(head_logement, head_archiveLogement)
    head_archiveLogement <- head_logement
    head_logement <- q
End

While ( q <> NULL ) do
Begin
    If ( CELL_VALUE(q).id = deleteID ) then
    Begin
        ASS_ADR(p, NEXT(q))
        ASS_ADR(q, head_archiveLogement)
        head_archiveLogement <- q
        Break
    End
    Else
    Begin
```

```
      p <- q
      q <- NEXT(q)
   End
End
End

Procedure deleteLocation()
Var
deleteID1, deleteID2 : integer
p, q : pointeur

Begin
write("please enter the ID of logement of the location to delete : ")
read(deleteID1)
write("please enter the ID of locataire to delete : ")
read(deleteID2)

p <- head_location
q <- NEXT(p)

If ( CELL_VALUE(head_location).logementID = deleteID1 AND
CELL_VALUE(head_location).locataireID = deleteID2 ) then
Begin
   ASS_ADR(head_location, head_archiveLocation)
   head_archiveLocation <- head_location
   head_location <- q
End

While ( q <> NILL ) do
Begin
   If ( CELL_VALUE(q).logementID = deleteID1 AND CELL_VALUE(q).locataireID = deleteID2 )
then
   Begin
      ASS_ADR(p, NEXT(q))
      ASS_ADR(q, head_archiveLocation)
      head_archiveLocation <- q
      Break
   End
   Else
   Begin
      p <- q
      q <- NEXT(q)
   End
End
```

```
End

Procedure deleteLocataire()
Var
deleteID : integer
p, q : pointeur

Begin
write("please enter the ID of locataire to delete : ")
read(deleteID)

p <- head_locataire
q <- NEXT(p)

If ( CELL_VALUE(head_locataire).id = deleteID ) then
Begin
   ASS_ADR(head_locataire, head_archiveLocataire)
   head_archiveLocataire <- head_locataire
   head_locataire <- q
End

While ( q <> NILL ) do
Begin
   If ( CELL_VALUE(q).id = deleteID ) then
   Begin
      ASS_ADR(p, NEXT(q))
      ASS_ADR(q, head_archiveLocataire)
      head_archiveLocataire <- q
      Break
   End
   Else
   Begin
      p <- q
      q <- NEXT(q)
   End
End
End

Procedure searchByDate()
Var
date : integer
p, q : pointeur
found : integer
```

```
Begin
write("enter the date ")
read(date)

p <- head_logement
write("the logements occupied in this date are : ")

While ( p <> NULL ) do
Begin
   q <- head_location
   While ( q <> NULL ) do
   Begin
      If ( CELL_VALUE(p).id = CELL_VALUE(q).logementID and date >=
CELL_VALUE(q).startDate and date <= CELL_VALUE(q).endDate ) then
      Begin
         print_maillon_logement(p)
         q <- NULL
      End
      Else q <- NEXT(q)
   End
   p <- NEXT(p)
End

write("the logements not occupied in this date are : ")

p <- head_logement
While ( p <> NULL ) do
Begin
   q <- head_location
   found <- 0
   While ( q <> NULL ) do
   Begin
      If ( CELL_VALUE(p).id = CELL_VALUE(q).logementID and date >=
CELL_VALUE(q).startDate and date <= CELL_VALUE(q).endDate ) then
      Begin
         found <- 1
         q <- NULL
      End
      Else q <- NEXT(q)
   End
   If ( not found ) then
   Begin
      print_maillon_logement(p)
   End
```

```
    p <- NEXT(p)
End
End

Procedure sortLocation()
Var
p, q : pointeur
temp : location

Begin
p <- head_location
While ( p <> NULL ) do
Begin
    q <- NEXT(p)
    While ( q <> NULL ) do
    Begin
        If ( loyerLocation(p) > loyerLocation(q) ) then
        Begin
            temp <- CELL_VALUE(p)
            ASS_VAL(p, CELL_VALUE(q))
            ASS_VAL(q, temp)
        End
        q <- NEXT(q)
    End
    p <- NEXT(p)
End
End

Procedure listLocTypeLog()
Var
p, q, r : pointeur
head_F4, tail_F4, head_F3, tail_F3, head_F2, tail_F2, head_studio, tail_studio : pointeur

Begin

p <- head_location

While ( p <> NULL ) do
Begin
    r <- head_logement
    While ( r <> NULL and CELL_VALUE(r).id <> CELL_VALUE(p).logementID ) do
        r <- NEXT(r)

    If ( r <> NULL ) then
```

```
Begin
   ALLOCATE_CELL(q)
   ASS_VAL(q, CELL_VALUE(p))
   ASS_ADR(q, NULL)

   If ( CELL_VALUE(r).type = "F4" ) then
   Begin
      If ( head_F4 = NULL ) then
      Begin
         head_F4 <- q
         tail_F4 <- q
      End
      Else
      Begin
         ASS_ADR(tail_F4, q)
         tail_F4 <- q
      End
   End
   Else If ( CELL_VALUE(r).type = "F3" ) then
   Begin
      If ( head_F3 = NULL ) then
      Begin
         head_F3 <- q
         tail_F3 <- q
      End
      Else
      Begin
         ASS_ADR(tail_F3, q)
         tail_F3 <- q
      End
   End
   Else If ( CELL_VALUE(r).type = "F2" ) then
   Begin
      If ( head_F2 = NULL ) then
      Begin
         head_F2 <- q
         tail_F2 <- q
      End
      Else
      Begin
         ASS_ADR(tail_F2, q)
         tail_F2 <- q
      End
   End
```

```
      Else If ( CELL_VALUE(r).type = "studio" ) then
      Begin
         If ( head_studio = NULL ) then
         Begin
            head_studio <- q
            tail_studio <- q
         End
         Else
         Begin
            ASS_ADR(tail_studio, q)
            tail_studio <- q
         End
      End
   End

   p <- NEXT(p)
End

If ( head_F4 <> NULL ) then sortLocation(head_F4)
If ( head_F3 <> NULL ) then sortLocation(head_F3)
If ( head_F2 <> NULL ) then sortLocation(head_F2)
If ( head_studio <> NULL ) then sortLocation(head_studio)

write("Locations F4 logements in order of price:")
write("ID-LOG  ID-LOC  START-D  END-D")
print_location(head_F4)

write("Location F3 logements in order of price:")
write("ID-LOG  ID-LOC  START-D  END-D")
print_location(head_F3)

write("Location F2 logements in order of price:")
write("ID-LOG  ID-LOC  START-D  END-D")
print_location(head_F2)

write("Location Studio logements in order of price:")
write("ID-LOG  ID-LOC  START-D  END-D")
print_location(head_studio)

End

Procedure listLocataireTypeLog()
Var
p, q, r, t : pointeur
```

```
head_F4, tail_F4, head_F3, tail_F3, head_F2, tail_F2, head_studio, tail_studio : pointeur

Begin

p <- head_location

While ( p <> NULL ) do
Begin
   r <- head_logement
   While ( r <> NULL and CELL_VALUE(r).id <> CELL_VALUE(p).logementID ) do
      r <- NEXT(r)

   If ( r <> NULL ) then
   Begin
      t <- head_locataire
      While ( t <> NULL and CELL_VALUE(t).id <> CELL_VALUE(p).locataireID ) do
         t <- NEXT(t)

      If ( t <> NULL ) then
      Begin
         ALLOCATE_CELL(q)
         ASS_VAL(q, CELL_VALUE(t))

         If ( ( CELL_VALUE(r).type = "F4" ) and ( CELL_VALUE(r).surface > 85 ) ) then
         Begin
            If ( head_F4 = NULL ) then
            Begin
               head_F4 <- q
               tail_F4 <- q
            End
            Else
            Begin
               ASS_ADR(tail_F4, q)
               tail_F4 <- q
            End
         End
         Else If ( ( CELL_VALUE(r).type = "F3" ) and ( CELL_VALUE(r).surface > 65 ) ) then
         Begin
            If ( head_F3 = NULL ) then
            Begin
               head_F3 <- q
               tail_F3 <- q
            End
            Else
```

```
            Begin
               ASS_ADR(tail_F3, q)
               tail_F3 <- q
            End
         End
         Else If ( ( CELL_VALUE(r).type = "F2" ) and ( CELL_VALUE(r).surface > 45 ) ) then
         Begin
            If ( head_F2 = NULL ) then
            Begin
               head_F2 <- q
               tail_F2 <- q
            End
            Else
            Begin
               ASS_ADR(tail_F2, q)
               tail_F2 <- q
            End
         End
         Else If ( ( CELL_VALUE(r).type = "studio" ) and ( CELL_VALUE(r).surface > 20 ) ) then
         Begin
            If ( head_studio = NULL ) then
            Begin
               head_studio <- q
               tail_studio <- q
            End
            Else
            Begin
               ASS_ADR(tail_studio, q)
               tail_studio <- q
            End
         End
      End
   End
   p <- NEXT(p)
End

write("Locataire F4 logements sont : ")
print_locataire(head_F4)

write("Locataire F3 logements sont : ")
print_locataire(head_F3)

write("Locataire F2 logements sont : ")
print_locataire(head_F2)
```

```
write("Locataire Studio logements sont : ")
print_locataire(head_studio)

End

Procedure addSmallestToList()
Var
current, prev, smallest, smallestPrev : pointeur
currentScore, smallestScore : entier

Begin
current <- head_logement
prev <- NULL
smallest <- head_logement
smallestPrev <- NULL

While ( current <> NULL ) do
Begin
   currentScore <- ( CELL_VALUE(current).distance * 10 ) / 2
   smallestScore <- ( CELL_VALUE(smallest).distance * 10 ) / 2

   If ( currentScore < smallestScore ) then
   Begin
      smallest <- current
      smallestPrev <- prev
   End

   prev <- current
   current <- NEXT(current)
End

If ( smallestPrev = NULL ) then
   head_logement <- NEXT(smallest)
Else
   ASS_ADR(smallestPrev, NEXT(smallest))

ASS_ADR(smallest, head_plpmoc)
head_plpmoc <- smallest
End

Procedure searchPlpMoc()
Var i : entier
Begin
```

```
head_plpmoc <- NULL

For i <- 0 to 4 do
   addSmallestToList()

write("Les logements les plus proches avec le loyer minimal sont : ")
print_logement(head_plpmoc)
End

Procedure historyNumCartier()
Var
year : entier
p : pointeur
Begin
write("Please, enter the year:")
read(year)

p <- head_archiveLocation

While ( p <> NULL ) do
Begin
   If ( ( CELL_VALUE(p).startDate / 10000 <= year ) and ( CELL_VALUE(p).endDate / 10000 >=
year ) ) then
   Begin
      q <- head_archiveLogement

      While ( q <> NULL ) do
      Begin
         If ( CELL_VALUE(q).id = CELL_VALUE(p).logementID ) then
         Begin
            r <- head_nmbrCartier
            prev <- NULL

            While ( ( r <> NULL ) and ( CELL_VALUE(r).name <> CELL_VALUE(q).street ) ) do
            Begin
               prev <- r
               r <- NEXT(r)
            End

            If ( r = NULL ) then
            Begin
               ALLOCATE_CELL(newNode)
               ASS_VAL(newNode, ( CELL_VALUE(q).street, 1 ))
               ASS_ADR(newNode, NULL)
```

```
                If ( prev = NULL ) then
                    head_nmbrCartier <- newNode
                Else
                    ASS_ADR(prev, newNode)
            End
            Else
            Begin
                CELL_VALUE(r).frequency <- CELL_VALUE(r).frequency + 1
            End
        End
        q <- NEXT(q)
    End
  End
  p <- NEXT(p)
End

print_nmbrCartier(head_nmbrCartier)
End

Procedure historyNumLogement()
Var
year, studio, F2, F3, F4 : entier
p, r : pointeur
Begin
write("please, enter the year:")
read(year)

studio <- 0
F2 <- 0
F3 <- 0
F4 <- 0

p <- head_archiveLocation

While ( p <> NULL ) do
Begin
  If ( ( CELL_VALUE(p).startDate / 10000 <= year ) and ( CELL_VALUE(p).endDate / 10000 >=
year ) ) then
  Begin
    r <- head_archiveLogement

    While ( ( r <> NULL ) and ( CELL_VALUE(r).id <> CELL_VALUE(p).logementID ) ) do
      r <- NEXT(r)
```

```
      If ( r <> NULL ) then
      Begin
         If ( CELL_VALUE(r).type = "F4" ) then
            F4 <- F4 + 1
         Else If ( CELL_VALUE(r).type = "F3" ) then
            F3 <- F3 + 1
         Else If ( CELL_VALUE(r).type = "F2" ) then
            F2 <- F2 + 1
         Else If ( CELL_VALUE(r).type = "studio" ) then
            studio <- studio + 1
      End
   End
   p <- NEXT(p)
End

write("F4: ", F4)
write("F3: ", F3)
write("F2: ", F2)
write("studio: ", studio)
End

Procedure print_logement(p : pointeur)
Var q : pointeur

Begin
   q <- p
   While ( q <> NULL ) do
   Begin
      write(CELL_VALUE(q).type, " ")
      write(CELL_VALUE(q).street, " ")
      write(CELL_VALUE(q).surface, " ")
      write(CELL_VALUE(q).distance, " ")
      write(CELL_VALUE(q).id, " ")
      q <- NEXT(q)
   End
End

Procedure print_location(p : pointeur)
Var q : pointeur

Begin
   q <- p
   While ( q <> NULL ) do
```

```
    Begin
        write(CELL_VALUE(q).logementID, " ")
        write(CELL_VALUE(q).locataireID, " ")
        write(CELL_VALUE(q).startDate, " ")
        write(CELL_VALUE(q).endDate, " ")
        q <- NEXT(q)
    End
End

Procedure print_locataire(p : pointeur)
Var q : pointeur

Begin
    q <- p
    While ( q <> NULL ) do
    Begin
        write(CELL_VALUE(q).firstName, " ")
        write(CELL_VALUE(q).lastName, " ")
        write(CELL_VALUE(q).phoneNumber, " ")
        write(CELL_VALUE(q).id, " ")
        q <- NEXT(q)
    End
End

Procedure print_nmbrCartier(p : pointeur)
Var q : pointeur

Begin
    q <- p
    While ( q <> NULL ) do
    Begin
        write(CELL_VALUE(q).name, " ")
        write(CELL_VALUE(q).frequency, " ")
        q <- NEXT(q)
    End
End

Procedure print_maillon_logement(p : pointeur)
Begin
    write(CELL_VALUE(p).type, " ")
    write(CELL_VALUE(p).street, " ")
    write(CELL_VALUE(p).surface, " ")
    write(CELL_VALUE(p).distance, " ")
    write(CELL_VALUE(p).id, " ")
```

```
End

Procedure print_maillon_location(p : pointeur)
Begin
   write(CELL_VALUE(p).logementID, " ")
   write(CELL_VALUE(p).locataireID, " ")
   write(CELL_VALUE(p).startDate, " ")
   write(CELL_VALUE(p).endDate, " ")
End

Procedure print_maillon_locataire(p : pointeur)
Begin
   write(CELL_VALUE(p).firstName, " ")
   write(CELL_VALUE(p).lastName, " ")
   write(CELL_VALUE(p).phoneNumber, " ")
   write(CELL_VALUE(p).id, " ")
End

Procedure print_maillon_nmbrCartier(p : pointeur)
Begin
   write(CELL_VALUE(p).name, " ")
   write(CELL_VALUE(p).frequency, " ")
End
```

Program principal :

```
Var
Choice, addchoice, deletechoice, searchchoice, historychoice, displaychoice : integer

Begin
While ( True ) do
   write("1- Add new")
   write("2- Delete")
   write("3- Search")
   write("4- Display")
   write("5- Exit")
   write("Enter your choice: ")
   read(choice)

   If ( choice = 1 ) then
      write("1- Add new logement")
      write("2- Add new locataire")
      write("3- Add new location")
```

```
    write("Enter your choice: ")
    read(addChoice)

    If ( addChoice = 1 ) then
        add_Logement()
    Else If ( addChoice = 2 ) then
        add_Locataire()
    Else If ( addChoice = 3 ) then
        add_Location()

Else If ( choice = 2 ) then
    write("1- Delete logement")
    write("2- Delete locataire")
    write("3- Delete location")
    write("Enter your choice: ")
    read(deleteChoice)

    If ( deleteChoice = 1 ) then
        delete_Logement()
    Else If ( deleteChoice = 2 ) then
        delete_Locataire()
    Else If ( deleteChoice = 3 ) then
        delete_Location()

Else If ( choice = 3 ) then
    write("1- Search by date")
    write("2- List locations by type of logement")
    write("3- List locataire by type of logement")
    write("4- Search the closest logement with minimal price")
    write("5- Consult history by year")
    write("Enter your choice: ")
    read(searchChoice)

    If ( searchChoice = 1 ) then
        searchByDate()
    Else If ( searchChoice = 2 ) then
        listLocTypeLog()
    Else If ( searchChoice = 3 ) then
        listLocataireTypeLog()
    Else If ( searchChoice = 4 ) then
        searchPlpMoc()
    Else If ( searchChoice = 5 ) then
        write("1- Number of logements rented this year by name of street")
        write("2- Number of logements rented this year by type of logement")
```

```
            write("Enter your choice: ")
            read(historyChoice)

            If ( historyChoice = 1 ) then
               historyNumCartier()
            Else If ( historyChoice = 2 ) then
               historyNumLogement()

      Else If ( choice = 4 ) then
         write("1- Display logement")
         write("2- Display locataire")
         write("3- Display location")
         write("Enter your choice: ")
         read(displayChoice)

         If ( displayChoice = 1 ) then
            print_logement(head_logement)
         Else If ( displayChoice = 2 ) then
            print_locataire(head_locataire)
         Else If ( displayChoice = 3 ) then
            print_location(head_location)

      Else If ( choice = 5 ) then
         write("Exiting...")
End
```