---------------REQUIREMENTS FOR IMPLEMENTATION OF SYSTEM OF DYNAMIC CIRCULAR LINKED LISTS---------------

1. To implement a system of dynamic circular linked lists.

2. It is menu driven system

3. No global variables are to be used.

4. Options provided in the menu are listed below :

   1. Creation of lists
   2. Insert operations
   3. Delete operations
   4. Search operations
   5. Display operations
   6. Exit

==

```
 --- > Creation of Lists
       --- create a CDLL by inserting elements at the beginning
       --- create a CDLL by inserting elements at the end
 ----> Insert operations
       --- Insert a value to a particular list at the beginning of the list
       --- Insert a value to a particular list at the end of the list
       --- Insert a value to a particular list after a given value in that list
       --- Insert a value to a particular list before a given value in that list
       --- Insert a value to a particular list at a given position
 ----> Delete operations
       --- Delete a value in a particular list
       --- Delete a node after a particular value in the list
       --- Delete a node before a particular value in the list
       --- Delete first node of a particular list
       --- Delete last node of a particular list
       --- Delete node at a given position in a given list
       --- Delete the entire list
 ----> Search operation
       --- Search for a value in the list
       --- Search for a value in the system of lists
       --- Maximum in a list
       --- Maximum in a system
       --- Minimum in a list
       --- Minimum in system
 ----> Display operations
       --- Forward display
       --- Reverse display
       --- Display all the list in the system
 ----> Exit
```

==

FUNCTIONALITY :-

==

CREATION OF LIST

1. CREATE A CDLL BY INSERTING AT THE BEGINNING

Expected operation to be performed on function call :

```
 NULL<---BEGIN    1 <---BEGIN     1              1
                  2 <---BEGIN    2
                             3<---BEGIN
```

2. CREATE A CDLL BY INSERTING AT THE END

Expected operation to be performed on function call :

```
 NULL<---BEGIN    1 <---BEGIN     1 <---BEGIN    1<---BEGIN
```

```
                    2          2
                              3
```

==

INSERT OPERATION

NOTE : IF THE USER ENTERS A LIST NUMBER THAT IS NOT AVAILABLE OR LISTED , THEN THE USER WILL BE RETURNED TO THE MAIN MENU

1.Insert a value to a particular list at the beginning of the list

  Expected operation to be performed on function call :

```
1<---BEGIN  ------   0 IS INSERTED -------->   0<---BEGIN
2                             1
3                             2
4                             3
5                             4
                             5
```

NOTE : THERE IS NO EXCEPTIONAL CASES INVOLVED

2. Insert a value to a particular list at the end of the list

  Expected operation to be performed on function call :

```
1<---BEGIN  ------   0 IS INSERTED -------->   1<---BEGIN
2                             2
3                             3
4                             4
5                             5
                             0
```

NOTE : THERE IS NO EXCEPTIONAL CASES INVOLVED

3. Insert a value to a particular list after a given value in that list

  Expected operation to be performed on function call :

```
1<---BEGIN  ------   0 IS INSERTED -------->   1<---BEGIN
2            AFTER 3              2
3                             3
4                             0
5                             4
                             5
```

NOTE : IF THE VALUE AFTER WHICH THE ELEMENT HAS TO ENTERED IS NOT AVAILABLE ,PRINT ERROR MESSAGE AND RETURN TO MAIN MENU

4. Insert a value to a particular list before a given value in that list

  Expected operation to be performed on function call :

```
1<---BEGIN  ------   0 IS INSERTED -------->   1<---BEGIN
2            BEFORE 3              2
3                             0
4                             3
5                             4
                             5
```

NOTE : IF THE VALUE BEFORE WHICH THE ELEMENT HAS TO ENTERED IS NOT AVAILABLE ,PRINT ERROR MESSAGE AND RETURN TO MAIN MENU

5. Insert a value to a particular list at a given position

  Expected operation to be performed on function call :

```
1<---BEGIN  ------   0 IS INSERTED -------->   1<---BEGIN
2                  AT POSITION 4            2
3                                        3
4                                        4
5                                        0
                                        5
```

NOTE : IF THE POSITION IS NOT VALID ,PRINT ERROR MESSAGE AND RETURN TO MAIN MENU

==

DELETE OPERATION

NOTE : IF THE USER ENTERS A LIST NUMBER THAT IS NOT AVAILABLE OR LISTED , THEN THE USER WILL BE RETURNED TO THE MAIN MENU

1. Delete a value in a particular list

Expected operation to be performed on function call :

```
1<---BEGIN  ------   2 IS DELETED  -------->   1<---BEGIN
2                                        3
3                                        4
4                                        5
5
```

NOTE : THERE IS NO EXCEPTIONAL CASES INVOLVED

2. Delete a node after a particular value in the list

Expected operation to be performed on function call :

```
1<---BEGIN  ------   DELETE AFTER 2 -------->  1<---BEGIN
2                                        2
3                                        4
4                                        5
5
```

NOTE : IF THE VALUE IS NOT FOUND , RETURNS ERROR

3. Delete a node before a particular value in the list

Expected operation to be performed on function call :

```
1<---BEGIN  ------   DELETE BEFORE 2------->   2<---BEGIN
2                                        3
3                                        4
4                                        5
5
```

NOTE : IF THE VALUE IS NOT FOUND , RETURNS ERROR

4. Delete first node of a particular list

Expected operation to be performed on function call :

```
1<---BEGIN  ------  DELETE FIRST NODE------>   2<---BEGIN
2                                        3
3                                        4
4                                        5
5
```

NOTE : THERE IS NO EXCEPTION INVOLVED

5. Delete last node of a particular list

Expected operation to be performed on function call :

```
1<---BEGIN  ------  DELETE LAST NODE------->   1<---BEGIN
2                                        2
3                                        3
```

```
4                                        4
5
```

NOTE : THERE IS NO EXCEPTIONAL CASES INVOLVED

6. Delete node at a given position in a given list

Expected operation to be performed on function call :

```
1<---BEGIN  ------  DELETE AT POSITION 3--->   1<---BEGIN
2                                        2
3                                        3
4                                        5
5
```

NOTE : IF THE POSITION IS OUT OF RANGE , THEN ERROR MESSAGE IS PRINTED

7. Delete a given list

Expected operation to be performed on function call :

```
1<---BEGIN  ------   DELETE LIST (NUMBER)------->   NULL<---BEGIN
2
3
4
5
```

NOTE : THERE IS NO EXCEPTIONAL CASES INVOLVED

==

SEARCH OPERATION

NOTE : IF THE USER ENTERS A LIST NUMBER THAT IS NOT AVAILABLE OR LISTED , THEN THE USER WILL BE RETURNED TO THE MAIN MENU

1. Search for a value in the list

```
1<---BEGIN  ------ SEARCH 2  ------->   1<--BEGIN
2                                  2  <----- FOUND
3                                  3
4                                  4
5                                  5
```

NOTE : POSITION NUMBER WILL BE DISPLAYED ALONG WITH THE NUMBER OF TIMES IT APPEARS

NOTE : INCASE OF THE NIL APPEARANCE , THEN VALUE NOT FOUND MESSAGE IS PRINTED

2. Search for a value in the system of lists

```
LIST 1
1<---BEGIN  ------ SEARCH 2  ------->   1<--BEGIN
2                                  2  <----- FOUND
3                                  3
4                                  4
5                                  5

LIST 2
1<---BEGIN  ------ SEARCH 2  ------->   1<--BEGIN
4                                  4
3                                  3
6                                  6
2                                  2 <---FOUND
```

NOTE : POSITION NUMBER WILL BE DISPLAYED ALONG WITH THE NUMBER OF TIMES IT APPEARS

NOTE : INCASE OF THE NIL APPEARANCE , THEN VALUE NOT FOUND MESSAGE IS PRINTED

3. Maximum in a list

```
1<---BEGIN  ------ SEARCH FOR  ----->   1<--BEGIN
2                  MAX              2
3                                3
4                                4
5                                5 <--MAXIMUM
```

4. Maximum in a system

```
LIST 1
1<---BEGIN  ------ SEARCH 2  ------->   1<--BEGIN
2                                2
3                                3
4                                4
5                                5

LIST 2
1<---BEGIN  ------ SEARCH 2  ------->   1<--BEGIN
4                                4
3                                3
6                                6 <- MAXIMUM
2                                2
```

5. Minimum in a list

```
1<---BEGIN  ------ SEARCH FOR  ----->   1<--BEGIN<-----MINIMUM
2                  MAX              2
3                                3
4                                4
5                                5
```

6. Minimum in system

```
LIST 1
1<---BEGIN  ------ SEARCH 2  ------->   1<--BEGIN
2                                2
3                                3
4                                4
5                                5

LIST 2
1<---BEGIN  ------ SEARCH 2  ------->   1<--BEGIN<---MIN
4                                4
3                                3
6                                6
2                                2
```

===

DISPLAY OPERATION

NOTE : IF THE USER ENTERS A LIST NUMBER THAT IS NOT AVAILABLE OR LISTED , THEN THE USER WILL BE RETURNED TO THE MAIN MENU

1. forward display

ACCEPT VALUE FROM USER...VAL =1

DISPLAY :

```
LIST 1
1
2
3
4
5
```

2. reverse display

ACCEPT VALUE FROM USER...VAL =1

DISPLAY :

LIST 1
5
4
3
2
1

3. display system

NOTE : ENTIRE SYSTEM OF LISTS ARE DISPLAYED

DISPLAY

LIST 1

1
2
3
4
5

LIST 2

2
6
5
1
2

LIST 3

0
4
2
3
1

NOTE : INCASE THERE ARE NO LISTS CREATED, NOTHING IS DISPLAYED AND MAIN MENU IS DISPLAYED
==

6. EXIT

ALL THE LISTS CREATED ARE FREED
==