Date : 11.02.2025
Lakshana Baskaran
24011103026

LAB - 3
List ADT - Array Implementation

QUESTION 1:

1. Write a C++ menu-driven program to implement List ADT using an array of size 5. Maintain proper boundary conditions and follow good coding practices. The List ADT has the following operations,

1. Insert Beginning
2. Insert End
3. Insert Position
4. Delete Beginning
5. Delete End
6. Delete Position
7. Search
8. Display
9. Rotate
10. Exit

SOURCE CODE:

```
//list ADT
#include <stdio.h>
#include <iostream>
using namespace std;
#define SIZE 5

class list {
    private:
        int arr[SIZE];
        int curr;

    public:
        list() {curr = -1}

        void insert_beginning(int*, int);
```

```
        void insert_end(int*, int);
        void insert_position(int*, int, int);
        int delete_beginning(int*);
        int delete_ending(int*);
        int delete_position(int*, int);
        int search(int*, int);
        void rotate(int*, int);

        bool empty();
        void print_list(int*);

    int getCurr() const {
        return curr;
    }
};

//INSERTIONS
void list::insert_beginning(int arr[SIZE], int x) {
    if (curr >= SIZE-1) {
        return;
    }

    for (int i = curr; i > -1; i--) {
        arr[i+1] = arr[i];
    }
    arr[0] = x;
    curr = curr + 1;
}

void list::insert_end(int arr[SIZE], int x) {
    printf("curr: %d\n", curr);
    if (curr < SIZE-1){
        arr[curr+1] = x;
        printf("The end: %d\n", arr[curr+1]);
        print_list(arr);
    }
    curr = curr + 1;
    print_list(arr);
}

void list::insert_position(int arr[SIZE], int x, int pos) {
    if (pos < 0 || pos > curr) {
        printf("invalid pos\n");
        return;
```

```
    }

    if (curr < SIZE-1) {
        for (int i = curr; i >= pos; i--) {
            arr[i+1] = arr[i];
        }
        arr[pos] = x;
        curr = curr + 1;
    }
    else{
        printf("OverFlowError Detected: Ignoring Request.\n");
    }

    print_list(arr);

}

//DELETIONS
int list::delete_beginning(int arr[SIZE]) {
    if (empty()){
        printf("UnderflowError: List is empty.\n");
        return 0;
    }
    int key = arr[0];

    for (int i = 0; i < (curr); i++) {
        arr[i] = arr[i+1];
    }
    curr = curr - 1;

    return key;
}

int list::delete_ending(int arr[SIZE]) {
    if (empty()){
        printf("UnderflowError: List is empty.\n");
        return 0;
    }
    int key = arr[curr];

    curr = curr - 1;

    return key;
}
```

Date : 11.02.2025
Lakshana Baskaran
24011103026

```
int list::delete_position(int arr[SIZE], int pos) {

    if (pos > curr || pos < 0 || empty() == true)
        printf("\nPossible Errors:\nUnderflowError: List is empty
\n(or)\nPosition value is Invalid\n");
        return 0;

    int key = arr[pos];
    printf("key : %d\n", key);
    for (int i = pos; i < (curr); i++) {
        arr[i] = arr[i+1];
    }
    curr = curr - 1;
    printf("key : %d\n", key);
    return key;
}

//SEARCH
int list::search(int arr[SIZE], int x) {
    for (int i = 0; i < curr; i++) {
        if (arr[i] == x) {
            return 1;
        }
    }
    return 0;
}

//ROTATE
void list::rotate(int arr[SIZE], int k) {
    int temp[SIZE];
    for (int i = 0; i < SIZE; i++) {
        temp[i] = arr[i];
    }

    for (int i = curr; i >= 0; i--) {
        if ((i+k) >= (curr+1)) {
            arr[i+k-curr-1] = temp[i];
        }
        else {
            arr[i+k] = temp[i];
        }
    }
}
```

Date : 11.02.2025
Lakshana Baskaran
24011103026

```
#include <stdbool.h>
bool list::empty() {
    return curr == -1;
}

void list::print_list(int A[]) {
    for (int i = 0; i <= curr; i++) {
        printf("%d ", A[i]);
    }
    printf("\n");
}

int main() {
    int A[SIZE] = {0};
    list L;
    int x, pos;
    int choice;
    printf("\n1 - Insert beginning\n2 - Insert positon\n3 - Insert
End\n");
    printf("\n4 - Delete beginning\n5 - Delete positon\n6 - Delete
End\n");
    printf("\n7 - Search\n8 - Rotate\n9 - Exit\n");
    while (choice != 9) { //menu
        printf("\nEnter you choice:");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter the element to be added: ");
                scanf("%d", &x);
                L.insert_beginning(A, x);
                break;

            case 2:
                printf("Enter the element to be added: ");
                scanf("%d", &x);
                printf("%d", x);
                L.insert_end(A, x);
                break;

            case 3:
                printf("Enter the element to be added and its
position: ");
```

```
            scanf("%d", &x);
            scanf("%d", &pos);
            L.insert_position(A, x, pos);
            break;

        case 4:
            printf("%d\n", L.delete_beginning(A));
            break;

        case 5:
            printf("%d\n", L.delete_ending(A));
            break;

        case 6:
            printf("Enter the position to delete: ");
            scanf("%d", &pos);
            printf("%d\n", L.delete_position(A, pos));
            break;

        case 7:
            printf("Enter the element to be searched: ");
            scanf("%d", &x);
            if (L.search(A, x) == 1)
                printf("Element found!\n");
            else
                printf("Element Not found!\n");
            break;

        case 8:
            int k;
            printf("Enter the number of time to be rotated: ");
            scanf("%d", &k);

            L.rotate(A, k);
            break;

        case 9:
            printf("Exiting...\n");
            return 0;
            break;

        default:
            break;
    }
```

```
        printf("\nThe list: ");
        L.print_list(A);
    }
```

OUTPUT:

```
lemon@jupiter:~/workspace/college/DSA/Lab-3$ ./out

1 - Insert beginning
2 - Insert positon
3 - Insert End

4 - Delete beginning
5 - Delete positon
6 - Delete End

7 - Search
8 - Rotate
9 - Exit

Enter you choice:1
Enter the element to be added: 2

The list: 2

Enter you choice:2
Enter the element to be added: 3
3curr: 0
The end: 3
2
2 3

The list: 2 3

Enter you choice:3
Enter the element to be added and its position: 1
1
2 1 3

The list: 2 1 3

Enter you choice:1
Enter the element to be added: 5

The list: 5 2 1 3

Enter you choice:8
Enter the number of time to be rotated: 3

The list: 2 1 3 5

Enter you choice:5
5

The list: 2 1 3

Enter you choice:4
2

The list: 1 3

Enter you choice:7
Enter the element to be searched: 4
Element Not found!

The list: 1 3

Enter you choice:9
Exiting...
```