자료구조 실습과제

배열 리스트 프로그램

5671144 강범창

1. 배열을 사용한 리스트를 매뉴로 조작하는 프로그램을 작성한다.

1.1 배열을 사용한 리스트를 매뉴로 조작하는 프로그램을 작성하기위해 리스트에 사용할 구조체와 element를 선언합니다.

```
#define MAX_LIST_SIZE 100 // list 의 최대크기

typedef int element;

// array list 구조체
typedef struct
{
	element array[MAX_LIST_SIZE];
	int size; // length
} ArrayListType;
```

1.2 error가 일어나면 에러 메시지를 출력하기위해 error 함수를 선언합니다.

```
// error 함수
void error(char *message)
{
		fprintf(stderr, "%s₩n", message);
		return;
}
```

1.3 배열 리스트를 구현합니다.

Init 함수입니다. 배열 리스트기에 list를 받아 list size를 초기화 시켜줍니다.

```
// create
// init -> list 를 주고 list 를 초기화
void init(ArrayListType* list)
{
	list->size = 0;
}
```

isEmpty 함수입니다. 배열리스트의 사이즈가 0 이라면 true 를 return하여 비어 있다는 것을 알 수 있습니다.

```
bool isEmpty(ArrayListType* list)
{
	return list->size == 0 ? true : false;
}
```

isFull 함수입니다. 배열 리스트의 size가 MAX_LIST_SIZE 즉 배열의 최대 크기 와 같다면 true를 return 합니다. 아니라면 false를 return를 합니다.

```
bool isFull(ArrayListType* list)
{
	return list->size == MAX_LIST_SIZE ? true : false;
}
```

getEntry 함수입니다 list 의 pos 즉 list 에 있는 ~~ 번째에 있는 값을 확인하는 데 사용합니다. 이때 0보다 작거나 size 보다 크다면 error를 출력합니다. 이후 -1 을 return 해줍니다.

```
element getEntry(ArrayListType* list, int pos)
{
		if (pos < 0 || pos >= list->size)
```

```
        {
                error("position error");
                return -1;
        }
        return list->array[pos];
}
```

printList 함수입니다. 리스트를 받아 리스트 에 있는 값들을 출력합니다.

```
void printList(ArrayListType* list)
{
        for (int i = 0; i < list->size; i++)
        {
                printf("%d->", list->array[i]);
        }
        printf("₩n");
}
```

insertLast 함수입니다. List 와 item 을 받으며 list 의 size 가 MAX SIZE 보다 크면 error 를 출력하고 함수를 종료를 하고, 아니라면 size를 늘리고 그 size 에 item 을 넣어 마지막에 insert를 완료합니다.

```
void insertLast(ArrayListType* list, element item)
{
        if (list->size >= MAX_LIST_SIZE)
        {
                error("list overflow");
                return;
        }
        list->size++;
        list->array[list->size] = item;
}
```

Insert 함수입니다. List 와 , 넣을 위치 , item을 입력으로 받으며 배열이 가득 차 차 있을 때, position 이 0 보다 작거나 list 의 최대 사이즈보다 클때 에러처리를 해줍니다. 그리고 움직인 횟수를 확인하기위해 moveCount 를 선언해준 다음, insert 를 해준 위치 까지 i의 위치를 빼면서 한칸씩 한칸씩 뒤쪽으로 밀어 원하는 위치에 item을 넣어줍니다.

```
void insert(ArrayListType* list, int pos, element item)
{
	// full 이거나 postion 이 0 보다 작거나 , list 의 최대 사이즈보다 클때 에러처리
	if (isFull(list) || pos < 0 || pos > list->size)
	{
		error("insert error");
		return;
	}
	int moveCount = 0;
	// ex) size = 3 이면 0 1 2의 2 를 가르키게 합니다.
	for (int i = (list->size - 1); i >= pos; i--)
	{
		list->array[i + 1] = list->array[i];
		moveCount++;
	}
	printf("Move : %d₩n", moveCount);
	list->array[pos] = item;
	list->size++;
}
```

Delete함수입니다. List와 지울 위치를 받으며 list 가 비어 있거나 지울 위치가 0 보다 작거나 pos의 값이 list의 size보다 클 경우 에러처리를 해준 후, 지워질 위치에 있는 값을 return 하는 함수이기에 return 값인 item 에 값을 저장해둔 후, 값을 덮어씌워 삭제를 구현합니다. 이때도 insert 와 같이 움직인 횟수를 확인하기 위해 moveCount 를 사용합니다. 이후 item을 return해줍니다.

```
element delete(ArrayListType* list, int pos)
```

```
{
	element item;
	if (isEmpty(list) || pos < 0 || pos >= list->size)
	{
		error("delete error");
		return -1;
	}
	// 값을 return 하기위해 복사해둡니다.
	item = list->array[pos];
	// 덮어 씌워 값을 삭제시켜줍니다.
	int moveCount = 0;
	for (int i = pos; i < (list->size - 1); i++)
	{
		list->array[i] = list->array[i + 1];
		moveCount++;
	}
	printf("Move : %d₩n", moveCount);
	list->size--;
	return item;
}
```

1.3 메인에서 입력을 받아 처리하는 부분입니다.
입력을 받아 0라면 프로그램을 종료하고, 아니라면 그에 맞는 메뉴를 조건에 맞춰 실행 시켜줍니다.

```
int main()
{

	ArrayListType list;
	int menu;
	int num1, num2;

	init(&list);
	while (1)
	{
```

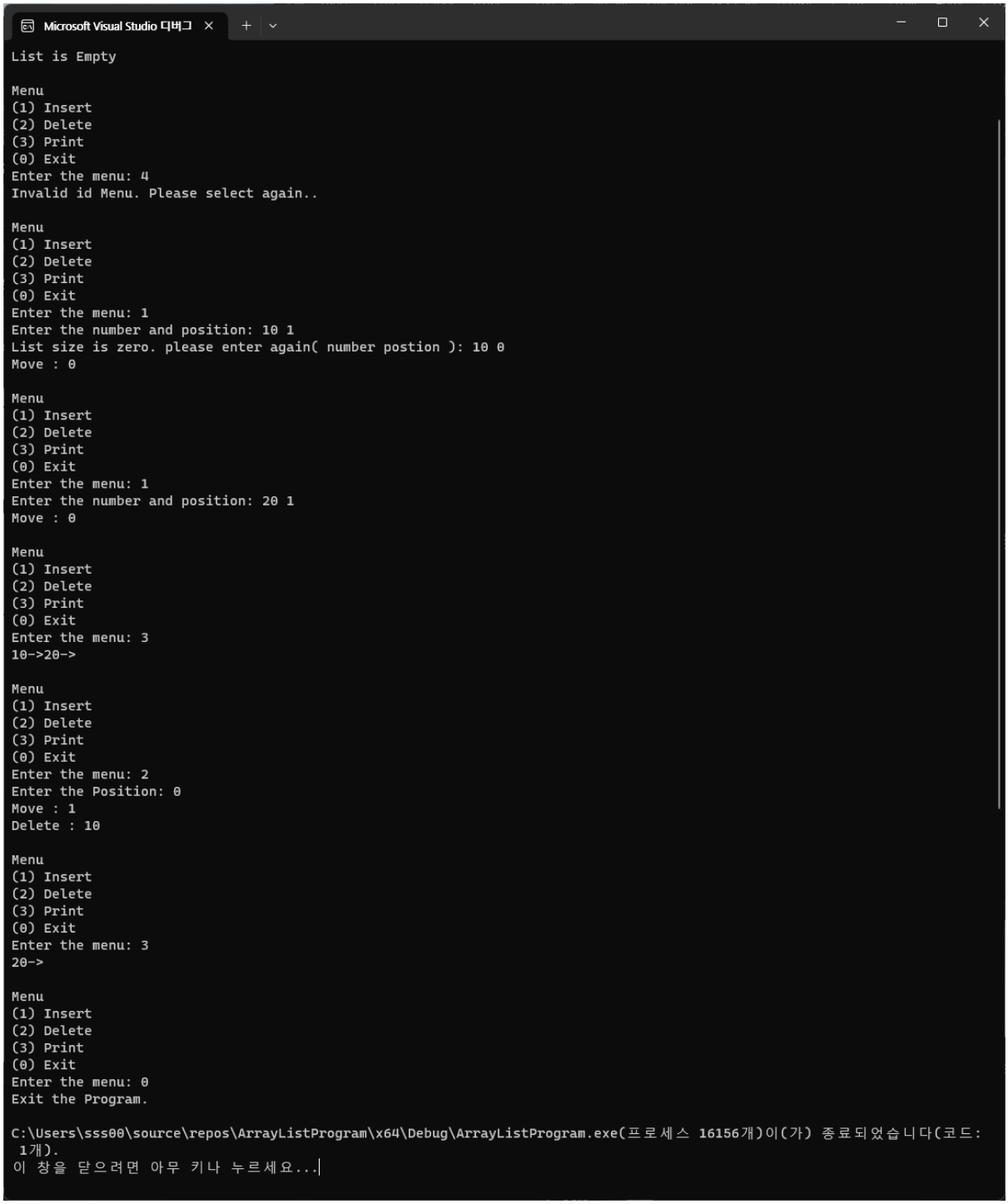
```
		printf("Menu₩n");
		printf("(1) Insert₩n");
		printf("(2) Delete₩n");
		printf("(3) Print₩n");
		printf("(0) Exit₩n");
		printf("Enter the menu: ");
		scanf("%d", &menu);

		switch (menu)
		{
		case 0:
			printf("Exit the Program.₩n");
			exit(1);
			break;
		case 1:
			printf("Enter the number and position: ");
			scanf("%d %d", &num1, &num2); // 값을 두 개 입력받아서 변수 두 개에 저장
			if (list.size == 0 && num2 != 0)
			{
				printf("List size is zero. please enter again( number postion ): ");
				scanf("%d %d", &num1, &num2);
			}
			insert(&list, num2, num1);

			break;
		case 2:
			if (isEmpty(&list))
			{
				printf("List is Empty₩n");
				break;
			}
			printf("Enter the Position: ");
```

```
                    scanf("%d", &num2);
                    int item = delete (&list, num2);
                    printf("Delete : %d₩n", item);
                    break;
                case 3:
                    printList(&list);
                    break;
                default:
                    printf("Invalid id Menu. Please select again..₩n");
                    break;
                }
                printf("₩n");
        }
}
```

아래는 실행 결과입니다.


Microsoft Visual Studio 디버그
List is Empty
Menu
(1) Insert
(2) Delete
(3) Print
(0) Exit
Enter the menu: 4
Invalid id Menu. Please select again..
Menu
(1) Insert
(2) Delete
(3) Print
(0) Exit
Enter the menu: 1
Enter the number and position: 10 1
List size is zero. please enter again(number postion): 10 0
Move : 0
Menu
(1) Insert
(2) Delete
(3) Print
(0) Exit
Enter the menu: 1
Enter the number and position: 20 1
Move : 0
Menu
(1) Insert
(2) Delete
(3) Print
(0) Exit
Enter the menu: 3
10->20->
Menu
(1) Insert
(2) Delete
(3) Print
(0) Exit
Enter the menu: 2
Enter the Position: 0
Move : 1
Delete : 10
Menu
(1) Insert
(2) Delete
(3) Print
(0) Exit
Enter the menu: 3
20->
Menu
(1) Insert
(2) Delete
(3) Print
(0) Exit
Enter the menu: 0
Exit the Program.
C:\Users\sss00\source\repos\ArrayListProgram\x64\Debug\ArrayListProgram.exe(프로세스 16156개)이(가) 종료되었습니다(코드:
1개).
이 창을 닫으려면 아무 키나 누르세요...