



Object-Oriented Programming Report

Assignment 3-1

Professor	Donggyu Sim
Department	Computer engineering
Student ID	2022202065
Name	박나림
Class (Design / Laboratory)	2 / C
Submission Date	2023. 5. 4

Program 1

□ 문제 설명

Merge_List(Node*p1, Node*p2, Node*p3)의 함수를 가지는 문자열 정렬 프로그램을 구현하는 문제이다. 사용자로부터 두개의 문자열을 입력 받으면 공백을 기준으로 단어를 구분하여 한 문자열씩 연결리스트로 병합 정렬한 다음, 정렬된 두 문자열을 위 함수를 이용하여 다시 병합 정렬을 하여 최종적으로 정렬된 하나의 문자열을 출력하는 프로그램인 것이다. 알파벳은 대소문자를 구분하지 않으며 오름차순으로 정렬해야 한다.

-구현 방법

먼저 사용자로부터 공백을 포함한 문자열을 입력 받은 뒤 공백을 토큰으로 설정하여 분리시키면서 연결리스트에 전달한다. 전달받을 때마다 새로운 노드를 생성하여 연결시키고, 한 문자열 내에서 노드끼리 알파벳을 비교하여 더 작은 값을 새 연결리스트에 저장한다. 각각 완성된 새 연결리스트를 인자로 전달받아서 다시 병합 정렬시킨 뒤 완성된 문자열을 출력시키도록 만든다.

□ 결과 화면

```
C:\Users\박나림\OneDrive\바탕 화면\과제\객체프\과제 3\3-1\Assignment 3-1\Debug\Assignment 3-1_1.exe
Input>>
Input list 1: Well done is better than well said
list 1 정렬: Well better done is said than well

Input list 2: Blaze with the fire that is never extinguished
list 2 정렬: Blaze extinguished fire is never that the with

Output>>
Well better done is said than well Blaze extinguished fire is never that the with
```

전체 연결리스트를 다시 정렬하는 것은 구현을 못하여 일단 각각 정렬하는 것으로 출력하였다. 하지만 대소문자를 구분하는 것도 어려움이 있어 우선 대문자를 앞으로 출력되게끔 하였다.

□ 고찰

파일 안에서 주석처리로 한 부분이 구현을 하다 실패한 부분인데, 일단 각 문자열을 정렬하는 것까지는 노드로 구성하여 구현하였지만 완성된 연결리스트를 다시 정렬하는 방법을 모르겠다. 그래서 연결리스트를 매개변수로 받아서 정렬하는 방법도 시도해봤지만 이 역시 병합 정렬로 구현하기가 아직 실력이 부족한 것 같아 완성을 못하였다. 또한 대소문자도 어떻게 반복시킬지 잘 생각을 못했으므로 이 문제를 풀면서 아직 연결리스트에 대해 더욱 공부할 필요가 있다는 것을 느꼈다.

Program 2

□ 문제 설명

내장된 함수 strtok()과 비슷하게 작동되는 my_strtok()함수를 구현하는 문제이다. 사용자로부터 이메일 주소를 입력 받으면, 이 주소를 토큰화 하기 위해 함수가 쓰이는 것인데, '@'와 '.'을 기준으로 문자열을 분리하여 한 줄씩 출력할 수 있도록 해야 한다. 이때 이 함수는 분리한 문자열의 첫번째 주소 값을 반환한다. 또한 가능한 모든 입력 예외 처리까지 구현하도록 한다.

-구현 방법

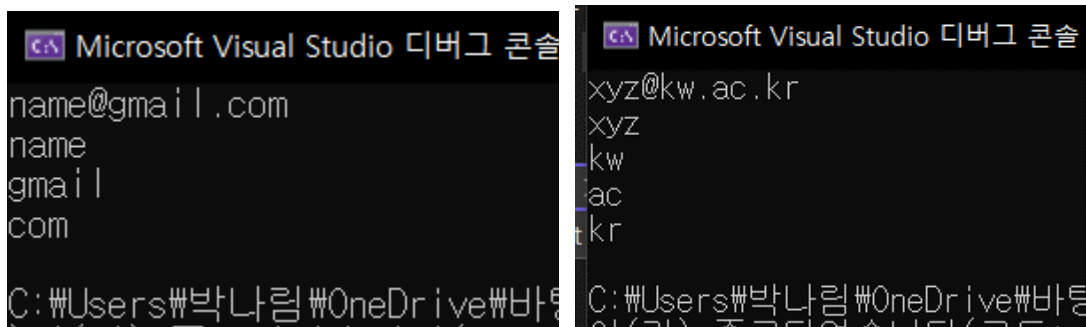
먼저 함수 안에서, 문자열을 매개변수로 전달받으면 static 문자열 포인터에 저장하도록 한다. 이 함수를 여러 번 호출할 때마다 분리한 문자열의 다음 칸 정보를 유지해야 하므로 정적 변수로 해야 구현할 수 있다. 이때 예외처리를 위해 문자열이 NULL 값이 아닌 경우에만 포인터에 저장하며, 빈 문자열일 경우에는 문자열에 포인터를 저장한다. 또한 포인터 자체가 NULL 값인 경우에는 NULL 값을 반환한다.

그 다음으로, 이 포인터를 가리킬 또다른 char*형 토큰 포인터를 만들어준다. 여기서 문자열 포인터가 정적 변수이므로 함수를 재호출 할 때마다 이 라인에서 값이 유지된 채로 다음 위치가 저장될 수 있게 하는 것이다. 그리고 전체 while 문으로 문자열

포인터가 NULL 이 아닐 때까지 반복하여, 해당 포인터가 '@'와 '.'일 때마다 그 위치를 NULL 로 바꿔준다. 그리고 포인터를 증가시켜 다음 위치로 이동시킨 뒤 토큰 포인터를 반환한다. while 문이 끝나고 다 자른 경우에도 토큰 포인터를 반환한다.

main 함수에서는 사용자로부터 이메일 주소를 입력받아서 그 문자열을 my_strtok 함수에 전달하여 출력한다. 그리고 두번째부터는 이전 호출 값 이후부터 다시 자르기 위해 NULL 값을 인자로 전달한다. 이러한 함수 호출을 전체 이메일 문자열 끝까지 반복하며 출력해주는 것으로 구현한다.

□ 결과 화면



```
Microsoft Visual Studio 디버그 콘솔
name@gmail.com
name
gmail
com
C:\Users\박나림\OneDrive\바탕화면\...

Microsoft Visual Studio 디버그 콘솔
xyz@kw.ac.kr
xyz
kw
ac
kr
C:\Users\박나림\OneDrive\바탕화면\...
```

예제의 이메일을 입력했을 때 기호를 기준으로 분리되어 한 줄씩 출력된 모습이다.

□ 고찰

처음에 문자열 포인터를 while 문 형태로 반복을 안 시키고 for 문에서 배열형태로 만들어서 검사를 돌렸더니 문자열이 제대로 나뉘어지지 않았다. 값을 유지시키는 변수이다 보니 배열 형태로 하면 오류가 나는 것 같았다. 그래서 while 에서 포인터를 증가시키는 형태로 하였더니 제대로 결과 값이 나올 수 있었다. 또한 제일 시행착오를 겪었던 부분은 main 함수 내에서 값을 출력할 때였다. 두번째부터는 NULL 값을 전달해야 하는데, 그걸 첫번째부터 같이 반복시키다 보니 무한 반복하는 등 여러 오류들이 생겼었다. 그래서 우선 첫번째만 먼저 출력하고 두번째부터는 반복시키는 형태로 고쳤더니 잘 출력될 수 있었다.

Program 3

□ 문제 설명

주어진 큐 클래스의 멤버함수들을 구현하여 최종적으로 큐 연결리스트를 구현하는 것이 목적인 문제이다. 이 클래스 안에는 반드시 비어있는지 확인하는 `IsEmpty()`, 다 찼는지 확인하는 `IsFull()`, 제거하는 `Pop()`, 새로 추가하는 `Push()` 함수들이 포함되어야 한다.

-구현 방법

먼저 사용자로부터 사이즈를 입력 받으면 큐 클래스 객체를 생성하여 `SetSize()` 함수에 인자로 전달한다. 그 후 노드 클래스 포인터 객체를 생성한 다음, 명령어 문자열 내용에 따라 if 문으로 각 명령을 실행한다. Push 일 경우, 노드 객체에 새 노드를 동적할당하여 생성한 다음 정수값을 입력 받으면 노드의 `SetData` 에 인자로 전달하고나서 큐 클래스 `Push()` 함수에 해당 노드를 인자로 전달하여 추가하는 형식으로 한다. 해당 함수 내에서는 큐가 비어있을 시와 꽉 차 있을 시에 대한 예외처리를 해주며, 나머지의 경우에는 마지막 포인터의 다음 포인터를 새 노드와 연결시키고 마지막을 그 노드로 설정시키는 형식으로 이어 나간다. 또한 노드 개수 변수도 같이 증가시키는 것으로 한다. Pop 일 경우, 노드에 해당 함수를 호출한 결과를 저장하며 만약 그 값이 null 포인터가 아닌 경우에만 그 노드의 `GetData()`를 출력하고 동적할당 된 노드를 지워주는 것으로 한다. `Pop()` 함수 내에서는 마찬가지로 비어있는 경우에 대한 예외처리를 먼저 해주고, 아닌 경우에 새 노드에 헤드 포인터 값을 저장한 뒤 헤드 포인터에는 그 다음 포인터의 값으로 옮겨 저장한다. 그리고 새 노드를 삭제하는 것으로 지워준다. 여기서 삭제했더니 다 삭제된 경우에는 마지막 포인터를 null 포인터로 설정하는 것으로 하며, 이 외에 노드 개수 변수를 감소시키고 지운 노드를 반환하는 것으로 구현한다. `IsEmpty()`는 헤드 포인터가 null 일 경우에 큐가 비어있는 것이니 그때 true 를 반환하도록 하고, `IsFull()`은 노드 개수 변수가 큐의 사이즈랑 같아질 때 가득 찬 상태이므로 그때 true 를 반환하는 것으로 한다. 마지막으로 `print()` 함수에서는 현재 노드를 생성하여 헤드 포인터 값으로 초기화 시킨 뒤 그 노드가 null 이 아닌 동안, 즉 끝에 도달하기 전까지 반복하면서 현재 값을 출력하고 다음 칸으로 옮기는 형식을 반복하게 만든다.

□ 결과 화면

```
Microsoft Visual Studio 디버그 콘솔
Queue의 크기 입력 : 3
-----
push 3
push 5
push 2
push 7
큐가 가득 찼습니다.

print
3 5 2

pop
3
pop
5
pop
2
pop
큐가 비어있습니다.

push 4
push 29
print
4 29

exit
C:\Users\박나림\OneDrive\바탕 화면\과제\객포트
```

차례대로 각 명령어들을 수행한 결과이다. 예외처리까지 검사할 수 있도록 해당 경우에는 안내문구를 출력하도록 구현하였다. 큐는 처음 들어온 요소가 처음으로 나가는 형태이므로 순서가 똑같이 출력된다.

□ 고찰

처음에 push 나 pop 을 구현할 때 예외처리를 생각 못하고 바로 구현했다가 결과 값이 제대로 안 나와서 시행착오들을 겪었었다. 특히 push 에서 큐가 비어있을 경우를 먼저 생각하고 head 포인터와 tail 포인터를 설정한 다음 그 다음으로 들어올 시 SetNext 로 맞춰줘야 한다는 걸 나중에 깨달았던 것 같다. 또한 pop 에서도 과정을 수행한 다음 마지막에 다 비어있는 경우엔 tail 포인터를 null 로 설정해야 된다는 것도 이번에 깨달았다.

Program 4

□ 문제 설명

3 번 문제와 비슷한 문제 유형으로, 이번에는 큐 대신에 스택을 연결리스트로 구현하는 문제이다. 마찬가지로 IsEmpty(), IsFull(), Pop(), Push() 함수들은 반드시 포함해야 한다.

-구현 방법

큐 연결리스트와 비슷하게 구현하지만, 이제 스택에서는 하나의 포인터로만 진행되기 때문에 약간의 차이가 있다. 마찬가지로 각 함수 내에서는 예외처리를 해주면서, push 를 할 때에는 새 노드의 포인터를 헤드 포인터로 설정하고 헤드 포인터에 현재 노드를 대입함으로써 현재 포인터를 다음 노드로 옮기는 과정을 수행한다. 또한 노드 개수 변수도 같이 증가시킨다. pop 함수 내에서는 스택이 비어있을 경우에만 예외처리를 해주고, 나머지 경우에는 지울 노드를 생성하여 헤드 포인터로 초기화 시킨 뒤 헤드 포인터를 다음 노드로 옮긴다. 스택은 마지막으로 들어온 요소가 맨 처음에 나가게 되므로 이런 연산만 해주면 제대로 작동될 수 있다. 그리고 노드 개수도 감소시키고 지울 노드를 반환해주는 것으로 한다. 이 외에 IsFull(), IsEmpty()는 큐에서 구현한 함수들과 똑같이 구현해주며, print() 함수도 이와 동일하게 진행해준다.

□ 결과 화면

```
Microsoft Visual Studio 디버그 콘솔
Stack의 크기 입력: 3
-----
push 3
push 5
push 2
push 7
스택이 가득 차 있습니다.

print
2 5 3

pop
2
pop
5
pop
3
pop
스택이 비어있습니다.

push 4
push 29
print
29 4

exit
C:\Users\박나림\OneDrive\바탕 화면\과제\객체프고
```

차례대로 각 명령어들을 수행한 결과이다. 예외처리까지 검사할 수 있도록 해당 경우에는 안내문구를 출력하도록 구현하였다. 스택은 마지막에 들어온 요소가 처음으로 나가는 형태이므로 순서가 반대로 출력된다.

□ 고찰

큐 연결리스트와 비슷한 원리로 구현되니 나머지 부분은 괜찮았지만, 이제 Pop 함수를 구현하는 부분이 어려웠다. 두개의 포인터를 사용하여 head 포인터를 다음 노드로 이동시키고 그 전 노드를 지우면 되는 큐와 달리, 스택은 하나의 head 포인터로 뒤로가기 동작을 수행해야 된다고 하니 아이디어가 바로 안 떠올랐었던 것 같다. 그래서 스택 연결리스트에 대해 공부를 더 해보다가, head 포인터를 다음 포인터로 설정하면 자연스럽게 지워진다는 것을 깨닫게 되었다. 하지만 이 역시 정확한 원리를 자세하게는 이해하지 못했으므로, 조금 더 공부를 해봐야 할 것 같다.