

2024 자료구조

[과제 4 트리그래프]



과목명 : 자료구조

학과 : 소프트웨어학과

학번 : 20231371

이름 : 김혜린

I 사전 탐색 트리 만들기

: randdict.txt를 이용해 이진 탐색 트리를 구현한다.

사전 데이터 파일에 저장되어 있는 데이터를 한 줄씩 읽어와 노드로 만든다. 그 후 노드에 저장된 word의 사전식 순서를 비교하여 트리에 노드를 추가한다. 이는 정렬되지 않은 데이터를 이용하여 구성하였기 때문에 트리의 높이가 최소가 아니다.

사전 데이터 파일의 모든 정보를 이진 트리로 구현한 다음 사용자가 단어를 입력하면 그에 맞는 뜻을 알려주는 코드를 추가한다. 이는 이진 트리를 중위 순회의 방식을 이용한다.

// 실행결과 : 단어의 개수 = 49434, 트리 높이 = 37, 노드의 개수 = 49267

```
Microsoft Visual Studio 디버그 콘솔
단어 : 49435개
트리 전체 높이 : 37
트리 노드 수 : 49267개
검색할 단어를 입력하세요 : lumen
의미 : n.(광속의 단위)루멘 (레벨 : 28)
계속하려면 1, 그만하려면 0을 입력하세요 : 1
검색할 단어를 입력하세요 : pilous
의미 : adj.=pilose (레벨 : 37)
계속하려면 1, 그만하려면 0을 입력하세요 : 0
```

II 사전 탐색 트리 개선하기

: 트리의 높이가 높으면 자료 검색에 시간이 오래 걸리므로 높이 낮추는 방법을 고안한다.

I에서 정렬되지 않은 데이터를 이용하여 단어의 사전식 순서를 비교하여 트리에 노드를 추가하는 방식을 사용했는데 이렇게 했을 경우에는 불균형 이진 트리가 형성될 수 있다. 이를 해결하기 위해 사전에서 데이터를 읽어올 때 배열에 저장한 뒤 퀵 정렬을 통해 먼저 사전식으로 정렬하고 이진 트리로 구현하였다. 정렬된 데이터의 처음이나 끝 원소를 root로 사용할 경우 한쪽으로 매우 치우친 편향 트리가 된다. 그러므로 root는 배열의 중간값을 사용하도록 한다. 이렇게 하면 균형 잡힌 트리를 만듦으로써 이진 트리의 높이를 최소한으로 만들 수 있다.

// 실행결과 : 단어의 개수 = 49435, 트리 높이 = 16, 노드의 개수 = 49435

```
Microsoft Visual Studio 디버그 콘솔
단어 : 49435개
트리 전체 높이 : 16
트리 노드 수 : 49435개
검색할 단어를 입력하세요 : lumen
의미 : n.(광속의 단위)루멘 (레벨 : 3)
계속하려면 1, 그만하려면 0을 입력하세요 : 1
검색할 단어를 입력하세요 : pilous
의미 : adj.=pilose (레벨 : 1)
계속하려면 1, 그만하려면 0을 입력하세요 : 0
```

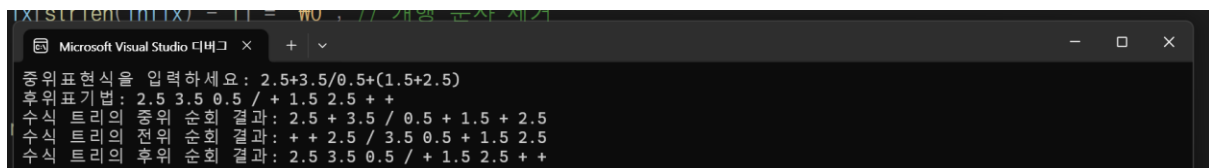
앞서 I에서의 트리 높이 37보다 훨씬 줄어든 트리로 구현하였음을 알 수 있다.

Ⅲ 수식 트리 만들기

: 중위표기법으로 작성된 수식으로 수식 트리 만든다.

먼저 사용자로부터 수식(중위표기법)을 입력 받는다. 이 수식 배열의 원소 하나하나에 접근하여 원소가 숫자이거나 '.'이면 피연산자로, 그렇지 않으면 연산자로 인식한다. 이를 스택을 이용하여 후위표기법으로 변환하고 또다시 스택을 이용하여 후위표기법을 수식 트리로 변환한다. 후위표기법에서 수식트리로 변환할 때 피연산자는 스택에 저장, 연산자는 스택에서 두 개의 노드를 꺼내와 각각을 자식 노드로 연결한다. 수식 트리를 완성한 뒤 중위순회, 전위순회, 후위순회를 하여 중위표기법, 전위표기법 후위표기법으로 만드는 코드도 추가한다.

// 수식 2.5 + 3.5 / 0.5 + (1.5 + 2.5)를 입력하였을 때



```
Microsoft Visual Studio 디버그 콘솔
중위표현식을 입력하세요 : 2.5+3.5/0.5+(1.5+2.5)
후위표기법 : 2.5 3.5 0.5 / + 1.5 2.5 + +
수식 트리의 중위 순회 결과 : 2.5 + 3.5 / 0.5 + 1.5 + 2.5
수식 트리의 전위 순회 결과 : + + 2.5 / 3.5 0.5 + 1.5 2.5
수식 트리의 후위 순회 결과 : 2.5 3.5 0.5 / + 1.5 2.5 + +
```

IV 미로 분석기

: 주어진 미로를 분석하여 길의 여부를 확인한다.

// 실행결과

실행 결과 : 길 이 2개 이상 있음																			
+	-	-	-	-	-	-	-	-	+	-	-	-	-	-	-	-	+	-	+
	0																		
	0	+	-	-	-	+			+	-	-	-	+		+				
	0	0	0	0	0														
+	-	-	-	+	0				+		+	-	+						
	0	0	0	0	0														
	0	+	-	+		+	-	+	-	+		+	-	+		+		+	
	0																		
	0			+	-	+				+	-	+		+	-	+		+	
	0				0	0	0												
	0	+			0	+	0	+	-	-	-	+	-	+		+		+	
	0				0		0												
	0	+	-	+	0		0	+	-	-	-	+	-	-	-	+		+	
	0	0	0	0	0		0		0	0	0	0	0	0	0	0			
+	-	+	-	-	-	+	0		0	+	-	-	-	+	0	+			
			0	0	0	0	0		0		0	0	0	0	0				
			0	+	-	-	-	+	0		0	+	-	-	-	+		+	
			0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	
		+	-	-	-	+		+	-	+	-	+		+	-	+	0		
																	0		
		+	-	+				+				+	-	+			0		
																	0		
+	-	+				+				+		+	-	-	-	+	0		
																	0		
+	-	-	-	+	-	-	-	+	-	-	-	+	-	-	-	-	-	+	

V 지하철 짧은 길찾기

: 사용자가 선택한 2개의 역 사이의 가장 짧은 길을 찾아 표시한다.

// 출발역 : 시청, 도착역 : 남영을 입력하였을 때

총 거리 : 4.5km, 총 소요시간 : 6.5분, 환승횟수 0



```
Microsoft Visual Studio 디버그 콘솔
출발역 입력: 시청
도착역 입력: 남영
경로: 시청 -> 남영
시청 (2.0 km, 2.0 분) -> 충정로 (0.8 km, 1.5 분) -> 서울역 (0.9 km, 1.5 분) -> 남영 (0.8 km, 1.5 분)
총 거리: 4.5 km
총 소요 시간: 6.5 분
환승 횟수: 0
```