**스택을 이용한 중위표기식을 후위표기식으로 전환하는 알고리즘 구현**

**학교: 서울잠원초등학교**

**학 번: 6학년 6반 51번**

**이 름: 강서희**

**수험번호: A20226001**

**요 약**

|  |
| --- |
| 중위 표기법은 사람들이 보통 계산할 때 사용하는 두 피연산자 사이에 연산자를 표기하는 방법입니다. 반면에 후위표기법은 두 피연산자의 뒤에 연산자를 표기하는 방법입니다. 예를 들어, A와 B를 더하는 식을 쓸 때 중위 표기법을 이용하여 쓰면 A+B라고 표기하고, 후위 표기법을 이용하면 AB+라고 표기합니다. 중위 표기법을 후위 표기법으로 바꾸려면 기호를 뒤쪽으로 옮겨야 하기 때문에 연산자를 기억해 놔야 합니다. 중위표기법을 후위 표기법으로 바꾸는 과정에서 가장 중요한 것은 우선 순위입니다. 피연산자 사이에 연산자가 있으면 피연산자 바로 뒤로 옮겨야 하는데 다음 연산에서 우선 순위가 더 높은 연산자가 있으면 나중에 나왔지만 먼저 출력해줘야 하기에 스택을 사용하면 편리합니다. 예를 들어, A+B/C를 ABC/+로 바꿀 때 +를 뒤로 옮기기 전에 /를 먼저 출력해야 합니다. +와 /를 둘 다 스택에 넣어주면 스택의 특징에 따라서 /가 먼저 출력되는 것입니다. |

**1.스택**

* 1. 스택의 정의
  2. 스택의 특징
  3. 스택에 쓰이는 함수

**2. 후위표기법**

2.1. 후위표기법의 계산 방법

2.2. 후위표기법이 필요한 까닭

2.3. 중위표기법을 후위표기법으로 전환하는 방법

**3. 알고리즘 구현 및 탐구보고서 작성 후 느낀 점**

**4. 참고자료**

**1. 스택**

1.1 스택의 정의

: 스택(stack)은 모든 원소들의 삽입(insert)과 삭제(delete)가 리스트의 한쪽 끝에서만 수행되는 제한 조건을 가지는 선형 자료 구조(linear data structure)로서, 삽입과 삭제가 일어나는 리스트의 끝을 top이라 하고, 다른 한쪽 끝을 bottom이라 합니다.

1.2. 스택의 특징

: 스택은 한 쪽 끝에서만 자료를 넣거나 뺄 수 있는 후입 선출 구조(LIFO - Last In First Out)으로 되어 있습니다. 자료를 넣는 것을 '밀어 넣는다' 라고 하여 푸쉬(push)라고 하고 반대로 넣어둔 자료를 꺼내는 것을 팝(pop)이라고 하는데, 이때 꺼내지는 자료는 가장 최근에 푸쉬한 자료부터 나오게 됩니다.

1.3. 스택에 사용한 함수

: -선언: stack<자료형> 변수 명(스택 이름);

-(스택 이름).push(element): 스택에 데이터 삽입

-(스택 이름).pop(): 데이터 삭제

-(스택 이름).top(): top에 있는 데이터 반환

-(스택 이름).empty(): 스택이 비어 있으면 1(True)아니면0(False)를 반환

-(스택 이름).size(): 스택의 사이즈(데이터 개수)를 변환

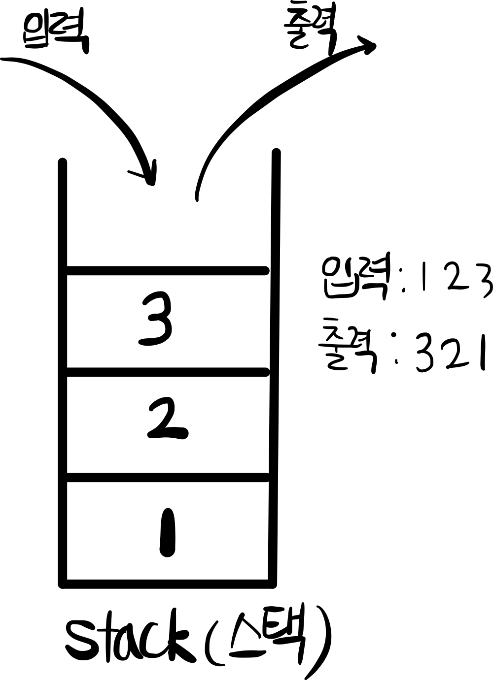


그림 1 스택

1. **후위표기법**

2.1. 특징 및 계산 방법

: 후위 표기법은 연산자를 연산 대상의 뒤에 쓰는 연산 표기법입니다.

이와 같은 방식은 우선 순위가 가장 높은 연산부터 계산이 되기 때문에 수식을 계산할 때 특별한 변환이 필요없이, 수식을 앞에서부터 읽어 계산하면 된다는 장점이 있습니다. 또한, 중위 표기법에서는 연산자의 우선순위가 모호해서 괄호가 필요한 경우가 있지만, 후위 표기법에서는 그러한 문제점이 발생하지 않습니다. 그러나 인간의 눈으로 쉽게 이해하거나 계산하기 힘들다는 문제점이 있어 눈에 보이는 표기보다는 주로 프로그램 내부의 표기법으로 사용합니다.

2.2. 후위표기법이 필요한 까닭

: 컴퓨터에게 후위 표기법이 필요한 까닭의 중점은 우선 순위에 있습니다. 사람들은 계산을 할 때 \*나 /가 보이면 그 부분부터 계산을 한 뒤 나머지 계산을 한다는 점을 잘 알기에 중위 표기법을 사용하곤 합니다. 하지만, 컴퓨터가 중위 표기법을 이용하여 계산하게 된다면 계산에 오차가 생기거나 비효율적인 방법을 이용하여 식을 계산해야 합니다. 예를 들어, 3+5\*7이라는 식이 있다고 하겠습니다. 이때, 우리는 5\*7을 먼저 계산한 후에 3을 더해주어 38이라는 올바른 결과를 도출해낼 수 있지만, 컴퓨터는 앞에서 뒤로 쭉 보고 계산하는 방식이기에 우선순위를 다 참고해 계산하려면 복잡한 방법을 사용해야합니다. 하지만, 후위 표기식은 앞에서부터 보면서 연산자가 나오면 그 앞의 두 숫자끼리 묶어주어, 손쉽게 계산할 수 있기 때문에 후위 표기식은 컴퓨터에게 꼭 필요합니다.

2.3 중위 표기식을 후위표기식으로 전환하기

: . 중위 표기법을 후위 표기법으로 바꾸려면 기호를 뒤쪽으로 옮겨야 하기 때문에 연산자를 기억해놔야 합니다. 중위표기법을 후위 표기법으로 바꾸는 과정에서 가장 중요한 것은 우선 순위입니다. 피연산자 사이에 연산자가 있으면 피연산자 바로 뒤로 옮겨야 하는데 다음 연산에서 우선 순위가 더 높은 연산자가 있으면 나중에 나왔지만 먼저 출력해줘야 하기에 스택을 사용하면 편리합니다. 예를 들어, A+B/C를 ABC/+로 바꿀 때 +를 뒤로 옮기기 전에 /를 먼저 출력해야 합니다. +와 /를 둘 다 스택에 넣어주면 스택의 특징에 따라서 /가 먼저 출력되는 것입니다. 연산자가 나오면 스택에 넣어줘야 하는데 우선 순위가 더 높거나 같은 연산자가 있다면 빼야 합니다.4가지의 경우로 나누어 보면 다음과 같습니다.

기본적으로 현재 연산자는 스택에 넣어주어야 합니다.

1. 연산자가 ‘\*’나’/’일 때

-> \*와/는 우선 순위가 가장 높습니다. 자신과 우선 순위가 같은 \*나/만 출력해주면 해결이 가능합니다.

2. 연산자가 ‘+’나’-‘일 때

-> \*와/보다 우선 순위가 낮기 때문에 \*와/먼저 출력해주고 연산자 우선 순위가 같은 +와-도 출력합니다. 괄호가 있는 경우에는 ‘(‘가 나올 때까지 출력해줍니다.

3. ‘(‘가 나올 때

->스택에 ‘(‘를 넣어줍니다.

4. ‘)’가 나올 때

-> ‘(‘가 나올 때까지 값들을 출력해줍니다.

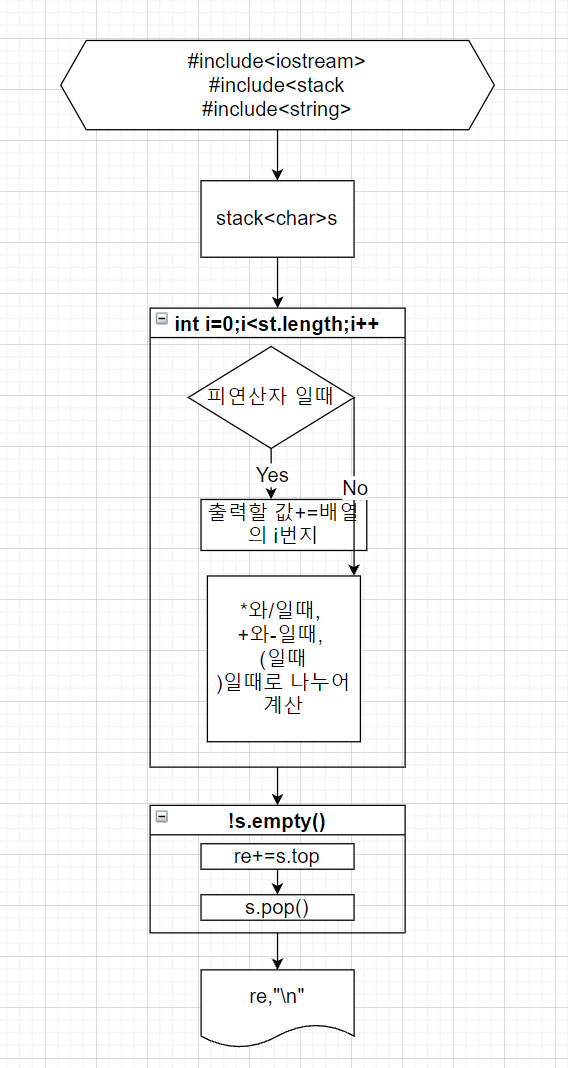
 (도 스택에서 빼줍니다.

그림2 순서도

|  |
| --- |
| #include <iostream>  #include<stack>  #include<string>  using namespace std;  int main() {  ios\_base::sync\_with\_stdio(0);  cin.tie(0);  string st,re;  stack<char>s;  cin>>st;  for(int i=0;i<st.length();i++)  {  if(('A'<=st[i]&&st[i]<='Z')||('1'<=st[i]&&st[i]<='9'))  {  re+=st[i];  }  else if(st[i]=='(')  {  s.push(st[i]);  }  else if(st[i]=='\*'||st[i]=='/')  {  while(!s.empty()&&(s.top()=='\*'||s.top()=='/'))  {  re+=s.top();  s.pop();  }  s.push(st[i]);  }  else if(st[i]=='+'||st[i]=='-')  {  while(!s.empty()&&s.top()!='(')  {  re+=s.top();  s.pop();  }  s.push(st[i]);  }  else if(st[i]==')')  {  while(!s.empty()&&s.top()!='(')  {  re+=s.top();  s.pop();  }  s.pop();  }  }  while(!s.empty())  {  re+=s.top();  s.pop();  }  cout<<re<<"\n";  } |

중위표기식을 후위표기식으로 바꾸는 코드는 다음과 같습니다.

1. **알고리즘 구현 및 탐구보고서 작성 후 느낀 점**

이번에 중위 표기법을 후위표기법으로 바꾸는 알고리즘을 짜보았습니다. 후위 표기법과 스택이 처음 접하는 이론이었고 사람들이 사용하는 계산 방식인 중위 표기법과는 표기하는 방식이 달라서 어려움이 있었지만, 이 어려움으로 인해 알고리즘 구현에 대한 해결방법을 더 주의 깊게 살펴보며 순서도를 먼저 그려보고 코드를 짜게 되어서 실수 없이 정확한 코드를 짤 수 있었습니다.

이번 탐구보고서를 작성하면서 순서도를 이용해 직접 설계하고 손으로 디버깅하여 검토하는 것이 정말 중요하다는 것을 또 다시 깨닫게 되는 가치 있는 시간이 되었습니다. 마지막으로, 좀 더 심화된 학습과 융합적 사고력의 성장을 할 수 있는 교육을 받고 의미 있는 공부를 이어갈 수 있는 영재교육원에 꼭 합격하고 싶습니다.

**4. 참고자료**

1.컴퓨터인터넷IT용어대사전-2011.1.20

2.위키백과 역 폴란드 표기법

3. 위키백과 스택