Lab 02

2023학년도 1학기 프로그래밍언어론

조교 송연주
ssyeonju@sookmyung.ac.kr

과제 02) 언어 S의 파서 구현

- 언어 S의 파서 구현 (Java)

  (1) 언어 S의 추상 구문 트리(AST)를 기초로 파서 구현 완성

  수식 또는 문장을 파싱하면서 AST를 구성하여 리턴

  비교 및 논리 연산 추가

  while, read, print문 추가

  (2) AST 노드를 트리 형태로 출력

  각 AST 노드에 대해서 들여쓰기 레벨을 이용하여 트리 형태로 출력하는 display 메소드 구현하여 AST 출력

과제 02) 언어 S의 파서 구현

- 언어 S의 파서 구현 (Java)
  - 언어 S의 문법 (EBNF)

<program> → {<command>}
<command> → <decl> | <stmt>
<decl> → <type> id [=<expr>];
<stmt> → id = <expr>;
| '{' <stmts> '}'
| if (<expr>) then <stmt> [else <stmt>]
| **while (<expr>) <stmt>**
| **read id;**
| **print <expr>;**
| let <decls> in <stmts> end;
<stmts> → {<stmt>}
<decls> → {<decl>}
<type> → int | bool | string

과제 02) 언어 S의 파서 구현

- 언어 S의 파서 구현 (Java)
  - 언어 S의 수식 문법 (EBNF)

<expr> → **<bexp> {& <bexp> | '|' <bexp>}** | !<expr> | true | false

<bexp> → **<aexp> [<relop> <aexp>]**

<relop> → == | != | < | > | <= | >=

<aexp> → <term> {+ <term> | - <term>}

<term> → <factor> {* <factor> | / <factor>}

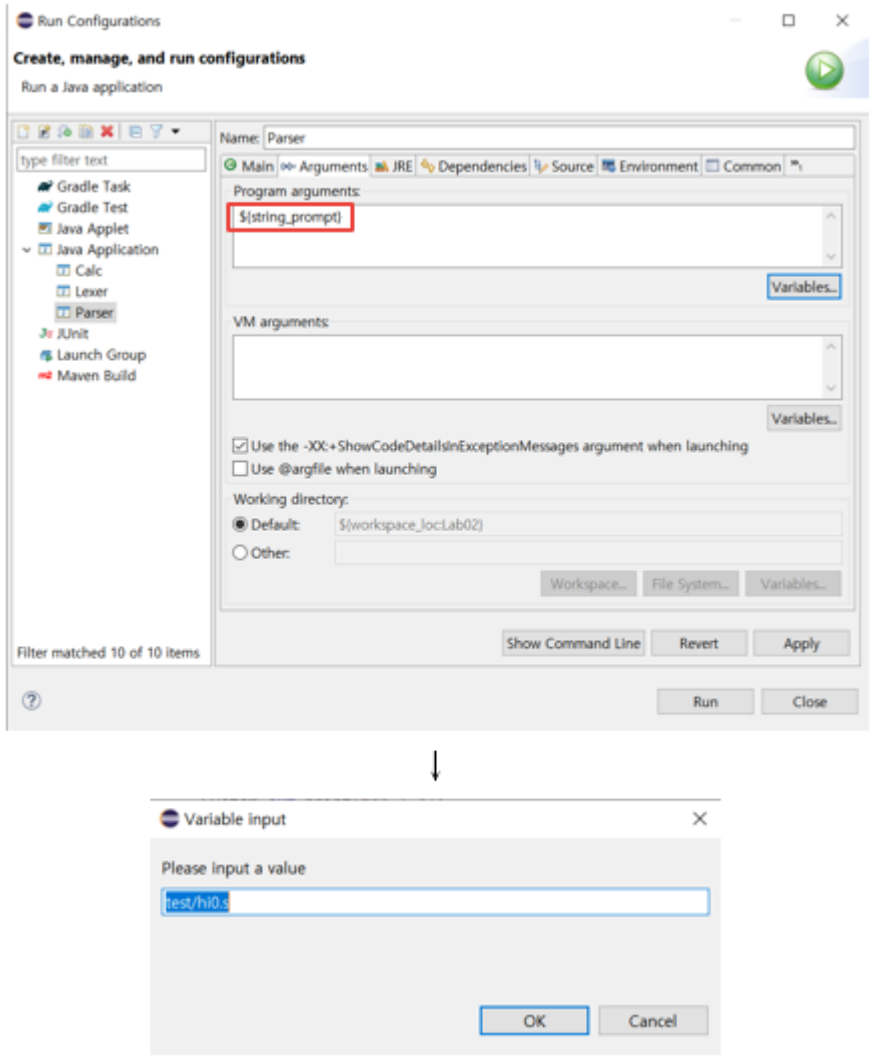
<factor> → [-] (<number> | (<aexp>) | id) | strliteral

과제 02) 언어 S의 파서 구현

- 언어 S의 파서 구현 (Java)
  - 예제 및 결과

    test 폴더에 있는 예제 파일

    ① hi0.s
    ② hi1.s
    ③ hi2.s
    ④ hi3.s
    ⑤ hi4.s
    ⑥ hi5.s
    ⑦ hi6.s
    ⑧ hi7.s



02) 언어 S의 파서 구현

hi0.s

```
Begin parsing... test/hi0.s

Print
        Value: hello world!
Decl
        Type: string
        Identifier: s
        Value: hello world!
Print
        Identifier: s
```

hi1.s

```
Begin parsing... test/hi1.s

Let
        Decls
                Decl
                        Type: string
                        Identifier: s
                        Value: hello
        Stmts
                Print
                        Identifier: s
                Print
                        Value: world !
                Read
                        Identifier: s
                Print
                        Identifier: s
```

hi2.s

```
Begin parsing... test/hi2.s

Let
        Decls
                Decl
                        Type: int
                        Identifier: i
                Decl
                        Type: int
                        Identifier: j
        Stmts
                Assignment
                        Identifier: i
                        Value: 1
                Print
                        Value: 2^n ?
                Read
                        Identifier: j
                While
                        Binary
                                Operator: >
                                Identifier: j
                                Value: 0
                        Stmts
                                Assignment
                                        Identifier: i
                                        Binary
                                                Operator: *
                                                Identifier: i
                                                Value: 2
                                Assignment
                                        Identifier: j
                                        Binary
                                                Operator: -
                                                Identifier: j
                                                Value: 1
                Print
                        Identifier: i
```

hi3.s

```
Begin parsing... test/hi3.s

Let
        Decls
                Decl
                        Type: int
                        Identifier: i
                        Value: 1
                Decl
                        Type: int
                        Identifier: sum
                        Value: 0
                Decl
                        Type: int
                        Identifier: n
        Stmts
                Print
                        Value: 1 + 2 + ... + n?
                Read
                        Identifier: n
                While
                        Binary
                                Operator: <=
                                Identifier: i
                                Identifier: n
                        Stmts
                                Assignment
                                        Identifier: sum
                                        Binary
                                                Operator: +
                                                Identifier: sum
                                                Identifier: i
                                Assignment
                                        Identifier: i
                                        Binary
                                                Operator: +
                                                Identifier: i
                                                Value: 1
                Print
                        Identifier: sum
```

hi4.s

```
Begin parsing... test/hi4.s

Let
        Decls
                Decl
                        Type: int
                        Identifier: i
                        Value: 0
        Stmts
                Let
                        Decls
                                Decl
                                        Type: int
                                        Identifier: i
                                Decl
                                        Type: int
                                        Identifier: j
                        Stmts
                                Assignment
                                        Identifier: i
                                        Value: 10
                                Assignment
                                        Identifier: j
                                        Value: 2
                                If
                                        Binary
                                                Operator: >
                                                Identifier: j
                                                Value: 0
                                        Assignment
                                                Identifier: i
                                                Binary
                                                        Operator: +
                                                        Identifier: i
                                                        Identifier: j
                                        Assignment
                                                Identifier: i
                                                Binary
                                                        Operator: -
                                                        Identifier: i
                                                        Identifier: j
                                Print
                                        Identifier: i
                Print
                        Identifier: i
```

02) 언어 S의 파서 구현

hi5.s

```
Begin parsing... test/hi5.s

Let
        Decls
                Decl
                        Type: int
                        Identifier: i
                Decl
                        Type: int
                        Identifier: j
                Decl
                        Type: int
                        Identifier: k
        Stmts
                Assignment
                        Identifier: i
                        Value: 1
                Assignment
                        Identifier: j
                        Value: 1
                While
                        Binary
                                Operator: <=
                                Identifier: i
                                Value: 3
                        Stmts
                                Assignment
                                        Identifier: j
                                        Value: 1
                                While
                                        Binary
                                                Operator: <=
                                                Identifier: j
                                                Value: 4
                                        Stmts
                                                Assignment
                                                        Identifier: k
                                                        Binary
                                                                Operator: *
                                                                Identifier: i
                                                                Identifier: j
                                                Print
                                                        Identifier: i
                                                Print
                                                        Identifier: j

                                                Print
                                                        Identifier: k
                                                Assignment
                                                        Identifier: j
                                                        Binary
                                                                Operator: +
                                                                Identifier: j
                                                                Value: 1
                                Assignment
                                        Identifier: i
                                        Binary
                                                Operator: +
                                                Identifier: i
                                                Value: 1
```

hi6.s

```
Begin parsing... test/hi6.s

Let
        Decls
                Decl
                        Type: int
                        Identifier: i
                        Value: 0
        Stmts
                Let
                        Decls
                                Decl
                                        Type: int
                                        Identifier: i
                                        Value: 1
                                Decl
                                        Type: int
                                        Identifier: j
                                        Value: 2
                        Stmts
                                Print
                                        Identifier: i
                                If
                                        Binary
                                                Operator: >
                                                Identifier: i
                                                Value: 0
                                        Assignment
                                                Identifier: i
                                                Binary
                                                        Operator: +
                                                        Identifier: i
                                                        Identifier: j
                                        Assignment
                                                Identifier: i
                                                Binary
                                                        Operator: -
                                                        Identifier: i
                                                        Identifier: j
                                Print
                                        Identifier: i
                Let
                        Decls
                                Decl
                                        Type: int
                                        Identifier: k
                                        Value: 3
                        Stmts
                                Assignment
                                        Identifier: i
                                        Identifier: k
        Print
                Identifier: i
```

hi7.s

```
Begin parsing... test/hi7.s

Let
        Decls
                Decl
                        Type: int
                        Identifier: i
                        Value: 0
        Stmts
                Let
                        Decls
                                Decl
                                        Type: int
                                        Identifier: i
                                        Value: 1
                                Decl
                                        Type: int
                                        Identifier: j
                                        Value: 1
                                Decl
                                        Type: bool
                                        Identifier: k
                                        Value: true
                        Stmts
                                Print
                                        Identifier: i
                                If
                                        Identifier: k
                                        Assignment
                                                Identifier: i
                                                Binary
                                                        Operator: +
                                                        Identifier: i
                                                        Identifier: j
                                        Assignment
                                                Identifier: i
                                                Binary
                                                        Operator: -
                                                        Identifier: i
                                                        Identifier: j
                                Print
                                        Identifier: i
                Let
                        Decls
                                Decl
                                        Type: int
                                        Identifier: k
                                        Value: 0
                        Stmts
                                Assignment
                                        Identifier: k
                                        Binary
                                                Operator: +
                                                Identifier: i
                                                Identifier: k
        Print
                Identifier: i
```

과제 02) 언어 S의 파서 구현

- 언어 S의 파서 구현 (Java)
  - 팁 (AST.java)

```
class Indent {
    public static void display(int level, String s) {
        String tab = "";
        System.out.println();
        for (int i=0; i<level; i++)
            tab = tab + "    ";
        System.out.print(tab + s);
    }
}

abstract class Command {
    // Command = Decl | Function | Stmt
    Type type =Type.UNDEF;
    public void display(int l) { }
}
```

과제 02) 언어 S의 파서 구현

- 언어 S의 파서 구현 (Java)
  - 팁 (AST.java – 문장의 AST)

```
>> int x = 0;

Decl
        Type: int
        Identifier: x
        Value: 0
>> x = x + 10;

Assignment
        Identifier: x
        Binary
                Operator: +
                Identifier: x
                Value: 10
```

```
class Assignment extends Stmt {
    // Assignment = Identifier id; Expr expr
    Identifier id;
    Expr expr;

    Assignment (Identifier t, Expr e) {
        id = t;
        expr = e;
    }

    public void display(int level) {
        Indent.display(level, "Assignment");
        id.display(level+1);
        expr.display(level+1);
    }
}
```

과제 02) 언어 S의 파서 구현

- 언어 S의 파서 구현 (Java)
  - 팁 (AST.java – 수식의 AST)

```
>> int x = 1;

Decl
        Type: int
        Identifier: x
        Value: 1
>> int y = 2;

Decl
        Type: int
        Identifier: y
        Value: 2
>> z = - (x + y);

Assignment
        Identifier: z
        Unary
                Operator: -
                Binary
                        Operator: +
                        Identifier: x
                        Identifier: y
```

```
class Binary extends Expr {
// Binary = Operator op; Expr expr1; Expr expr2;
    Operator op;
    Expr expr1, expr2;

    Binary (Operator o, Expr e1, Expr e2) {
        op = o; expr1 = e1; expr2 = e2;
    } // binary

    public void display(int level) {
        Indent.display(level, "Binary");
        op.display(level+1);
        expr1.display(level+1);
        expr2.display(level+1);
    }
}
```

과제 02) 언어 S의 파서 구현

- 언어 S의 파서 구현 (Java)
  - 팁 (Parser.java)

```
Command command = null;
try {
    command = parser.command();
if (command != null)
    command.display(0);
} catch (Exception e) {
    System.err.println(e);
}
```

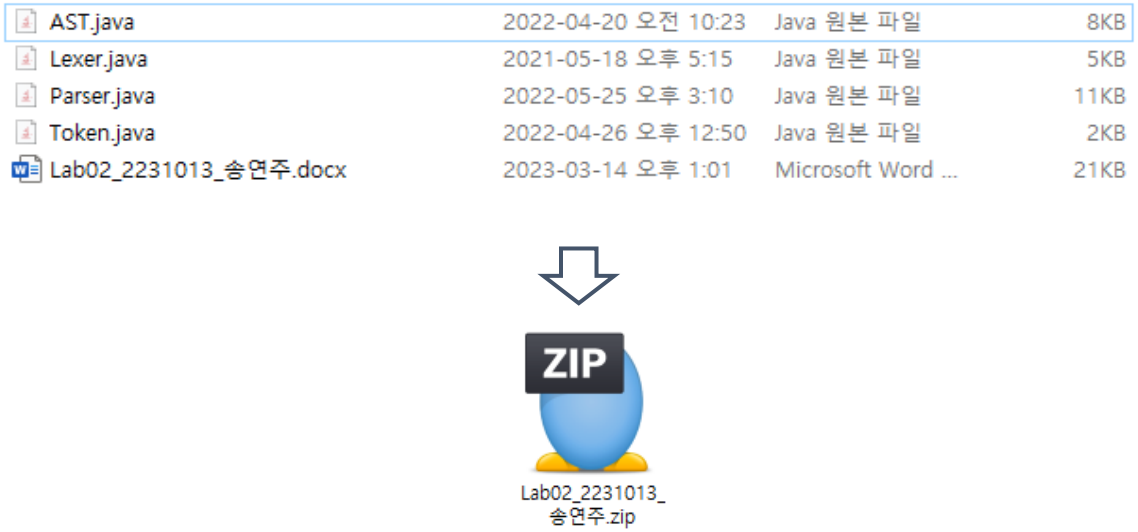
```
private Stmt assignment() {
// <assignment> -> id = <expr>;
    Identifier id = new Identifier(match(Token.ID));
    match(Token.ASSIGN);
    Expr e = expr();
    match(Token.SEMICOLON);
    return new Assignment(id, e);
}
```

+) 과제 제출 기한

- 과제 제출 기한
  - 01분반 : 4월 11일 화요일 오후 11:59
  - 02분반 : 4월 12일 수요일 오후 11:59
- 제출 장소
  - 스노우보드 과제 제출 페이지에 업로드
- 추가 제출
  - 제출 기한 이후 24시간 이내 조교 메일로 전송 : 10% 감점
  - 그 이후는 20% 감점

+) 과제 제출 양식

- 소스파일(.java)과 과제보고서(.docx)가 담기 압축파일(.zip 등) 제출
  - 압축 파일 이름 : Lab과제번호_학번_이름
    예) Lab02_2231013_송연주



- 소스파일 이름
  - PPT에 제시

- 과제보고서 양식
  - 스노우보드에서 다운로드
  - 프로그램 전체 코드 및 프로그램 설명
  - 실행 결과 화면을 캡처한 이미지 첨부

+) 과제 질문

- ssyeonju@sookmyung.ac.kr (조교 메일로 질문 보내기)
- 질문 시 주의사항

  - 메일에 반드시 과목, 분반, 전공, 이름, 학번 명시
  - 몇 번 과제에서 어떤 부분이 막혔는지, 어떤 과정이 문제인지 설명 첨부
    (그냥 코드만 보내면 어디가 문제인지 알 수 없어요)
  - 답장이 늦을 수 있으니 이 점 고려하여 미리 질문 (특히 과제 제출 마지막날 유의!)
  - 그 외 출석 등 다른 질문들도 메일로

+) 참고

- 채점시 고려하는 기본 사항
  - 보고서 구성에 맞게 보고서를 작성
    ① 소스코드 프로그램 전체 코드
    ② 프로그램 설명
    ③ 실행 화면 캡처
  - 확장
    ① 빌드(컴파일)가 가능한지 테스트
    ② 제시된 예제가 출력되는지 테스트
    ③ 의도에 맞게 출력되는지 테스트