과제 03) 언어 S의 인터프리터 구현

- 언어 S의 인터프리터 구현 (Java)

  (1) Let문 구현을 위한 allocate 함수와 free 함수 구현

  선언된 변수들을 위한 엔트리들을 상태 state에 추가 (allocate)

  선언된 변수들을 위한 엔트리들을 상태 state에서 제거 (free)

  (2) 언어 S의 문법에 따라 관계 및 논리 연산 수행 기능 구현

  binaryOperation()을 확장하여 정수, 스트링 관계 연산 및 부울값의 논리연산 구현

과제 03) 언어 S의 인터프리터 구현

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

<stmt> → id = <expr>;
| '{' <stmts> '}'
| if (<expr>) then <stmt> [else <stmt>]
| while (<expr>) <stmt>
| read id;
| print <expr>;
| let <decls> in <stmts> end;

(3) 언어 S의 문장에 do-while문, for문을 추가하고 이를 해석하는 인터프리터 작성

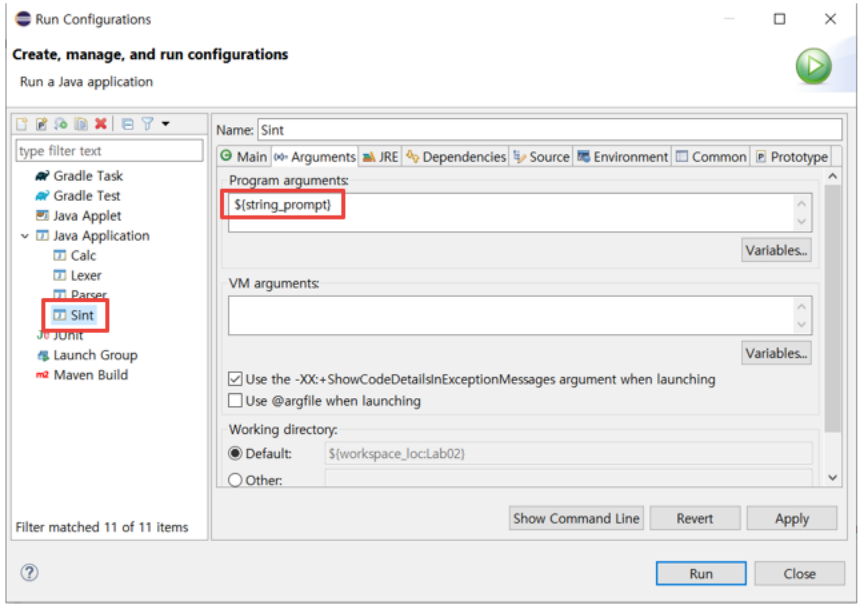
<stmt> → …
| do <stmt> while (<expr>);
| for (<type> id = <expr>; <expr>; id = <expr>) <stmt>

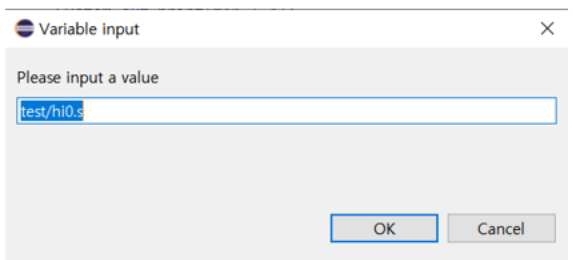
과제 03) 언어 S의 인터프리터 구현

- 언어 S의 인터프리터 구현 (Java)
  - 예제 및 결과

    test 폴더에 있는 예제 파일

    ① hi0.s
    ② hi2.s
    ③ hi3.s
    ④ hi4.s
    ⑤ hi5.s
    ⑥ hi6.s
    ⑦ hi7.s
    + ⑧, ⑨ String 관계 연산 테스트 2개
    + ⑩, ⑪ 논리 연산 테스트 2개
    + ⑫ for문, ⑬ do-while문 테스트 각각 1개





과제 03) 언어 S의 인터프리터 구현

hi0.s

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

Print
        Value: hello world!
Interpreting...test/hi0.s
hello world!

Decl
        Type: string
        Identifier: s
        Value: hello world!
Interpreting...test/hi0.s

Print
        Identifier: s
Interpreting...test/hi0.s
hello world!
```

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
Interpreting...test/hi2.s
2^n ?
10
1024
```

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
Interpreting...test/hi3.s
1 + 2 + ... + n?
10
55
```

hi4.s

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

        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
Interpreting...test/hi4.s
12
0
```

과제 03) 언어 S의 인터프리터 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
Interpreting...test/hi5.s
1
1
1
1
2
2
1
3
3
1
4
4
2
1
2
2
2
4
2
3
6
2
4
8
3
1
3
3
2
6
3
3
9
3
4
12
```

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
Interpreting...test/hi6.s
1
3
3
```

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
Interpreting...test/hi7.s
1
2
0
```

과제 03) 언어 S의 인터프리터 구현

관계 연산 테스트

stringrelop1.s

```
stringrelop1.s
1   string i = "apple";
2   string j = "banana";
3   if (i == j)
4       then print "strings are equal";
5   else
6       print "strings are not equal";
```

```
Begin parsing... test/stringrelop1.s

Decl
        Type: string
        Identifier: i
        Value: apple
Interpreting...test/stringrelop1.s

Decl
        Type: string
        Identifier: j
        Value: banana
Interpreting...test/stringrelop1.s

If
        Binary
                Operator: ==
                Identifier: i
                Identifier: j
        Print
                Value: strings are equal
        Print
                Value: strings are not equal
Interpreting...test/stringrelop1.s
strings are not equal
```

stringrelop2.s

```
stringrelop2.s
1   string i = "apple";
2   string j = "banana";
3   if (i < j)
4       then print "banana is located behind the dictionary";
5   else
6       print "apple is located behind the dictionary";
```

```
Begin parsing... test/stringrelop2.s

Decl
        Type: string
        Identifier: i
        Value: apple
Interpreting...test/stringrelop2.s

Decl
        Type: string
        Identifier: j
        Value: banana
Interpreting...test/stringrelop2.s

If
        Binary
                Operator: <
                Identifier: i
                Identifier: j
        Print
                Value: banana is located behind the dictionary
        Print
                Value: apple is located behind the dictionary
Interpreting...test/stringrelop2.s
banana is located behind the dictionary
```

과제 03) 언어 S의 인터프리터 구현

논리 연산 테스트

logicalop1.s

logicalop2.s

logicalop1.s

```
1   bool i = true;
2   bool j = true;
3   if (i & j)
4       then print "both are true";
5   else
6       print "one or both are false";
```

```
Begin parsing... test/logicalop1.s

Decl
        Type: bool
        Identifier: i
        Value: true
Interpreting...test/logicalop1.s

Decl
        Type: bool
        Identifier: j
        Value: true
Interpreting...test/logicalop1.s

If
        Binary
                Operator: &
                Identifier: i
                Identifier: j
        Print
                Value: both are true
        Print
                Value: one or both are false
Interpreting...test/logicalop1.s
both are true
```

logicalop2.s

```
1   bool i = true;
2   bool j = false;
3   if (i | j)
4       then print "one or both are true";
5   else
6       print "both are false";
```

```
Begin parsing... test/logicalop2.s

Decl
        Type: bool
        Identifier: i
        Value: true
Interpreting...test/logicalop2.s

Decl
        Type: bool
        Identifier: j
        Value: false
Interpreting...test/logicalop2.s

If
        Binary
                Operator: |
                Identifier: i
                Identifier: j
        Print
                Value: one or both are true
        Print
                Value: both are false
Interpreting...test/logicalop2.s
one or both are true
```

과제 03) 언어 S의 인터프리터 구현

for문, do-while문 테스트

for.s

for.s

```
1  for (int i=0; i<10; i = i+1) print i;
```

```
Begin parsing... test/for.s

Let
        Decls
                Decl
                        Type: int
                        Identifier: i
                        Value: 0
        Stmts
                While
                        Binary
                                Operator: <
                                Identifier: i
                                Value: 10
                        Stmts
                                Print
                                        Identifier: i
                                Assignment
                                        Identifier: i
                                        Binary
                                                Operator: +
                                                Identifier: i
                                                Value: 1

Interpreting...test/for.s
0
1
2
3
4
5
6
7
8
9
```

dowhile.s

```
Begin parsing... test/dowhile.s

Let
        Decls
                Decl
                        Type: int
                        Identifier: i
                        Value: 5
        Stmts
                Stmts
                        Stmts
                                Print
                                        Identifier: i
                                Assignment
                                        Identifier: i
                                        Binary
                                                Operator: -
                                                Identifier: i
                                                Value: 1
                        While
                                Binary
                                        Operator: >
                                        Identifier: i
                                        Value: 0
                                Stmts
                                        Print
                                                Identifier: i
                                        Assignment
                                                Identifier: i
                                                Binary
                                                        Operator: -
                                                        Identifier: i
                                                        Value: 1

Interpreting...test/dowhile.s
5
4
3
2
1
```

dowhile.s

```
1  let
2      int i = 5;
3  in
4      do {
5          print i;
6          i = i - 1;
7      }
8      while (i > 0);
9  end;
```

과제 03) 언어 S의 인터프리터 구현

- 언어 S의 인터프리터 구현 (Java)
  - 팁 (Sint.java - Let)

```
State Eval(Let l, State state) {
    State s = allocate(l.decls, state);
    s = Eval(l.stmts,s);
    return free(l.decls, s);
}

State allocate (Decls ds, State state) {
    if (ds != null) {
        // add entries for declared variables on the state
    }
    return null;
}

State free (Decls ds, State state) {
    if (ds != null) {
        // free the entries for declared variables from the state
     }
     return null;
}
```

과제 03) 언어 S의 인터프리터 구현

- 언어 S의 인터프리터 구현 (Java)
  - 팁 (Sint.java – binaryOperation)

```
Value binaryOperation(Operator op, Value v1, Value v2) {
    check(!v1.undef && !v2.undef,"reference to undef value");
    switch (op.val) {
    case "+":
        return new Value(v1.intValue() + v2.intValue());
    case "-":
        return new Value(v1.intValue() - v2.intValue());
    case "*":
        return new Value(v1.intValue() * v2.intValue());
    case "/":
        return new Value(v1.intValue() / v2.intValue());

    // relational operations

    // logical operations and or not

    default:
        throw new IllegalArgumentException("no operation");
    }
}
```

과제 03) 언어 S의 인터프리터 구현

- 언어 S의 인터프리터 구현 (Java)
  - 팁 (Parser.java – dowhile, for)

dowhile : 1번 이상 반복

do <stmt> while (<expr>);

=

{
<stmt>
  while (<expr>)
    <stmt>
}

cf. while : 0번 이상 반복

for

for (<type> id = <expr>; <expr>; id=<expr>) <stmt>

=

let
  <type> id = <expr>
in
  while (<expr>)
    <stmt>
end