**REPORT**

**[Lab3]**



**과 목 : 프로그래밍언어론01**

**담당교수 : 송수환 교수님**

**학 과 : 컴퓨터공학과**

**학 번 : 2021111971**

**이 름 : 이재혁**

텍스트, 클립아트이(가) 표시된 사진

자동 생성된 설명

**수정한 소스코드**

**allocate()**

**텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명**

ds 안의 decl을 꺼내 지역 변수를 추가합니다.

꺼낸 decl의 expr(변수에 할당할 값)이 있다면 expr을 계산해 id와 Pair로 만들어state에 push 합니다.

꺼낸 decl의 expr이 없다면 decl의 타입으로 기본 Value를 생성해 Pair로 만들어 state에 push 합니다.

**free()**

텍스트, 소프트웨어, 폰트, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

가장 최근에 생성된 지역변수가 스택의 상단에 위치합니다. stack은 Last – In – First – Out이기 때문에, ds만큼의 변수가 free 된다면, 가장 최근에 생성된 지역변수들 이므로, ds안의 decl 개수 만큼 pop을 시행합니다.

**binaryOperation()**

v1, v2의 타입에 따라, 다르게 연산합니다.

**case “==”**

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명**

**case “!=”**

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명**

int, bool 타입의 경우 기본 비교 연산이 존재하기 때문에, ==, != 연산자를 사용해 결과 값을 Value객체로 리턴합니다.

String type의 경우 자바환경에서 ==, != 연산자를 지원하지 않기 때문에, 클래스 메소드인 equals()를 사용해 두 String 객체를 비교합니다.

비교 결과를 그대로 리턴하면, 같을 때 true,

not연산으로 리턴하면 다를 때 true 값을 리턴하게 됩니다.

**case “<”**

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

**case “<=”**

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

**case “>”**

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

**case “>=”**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

비교 연산자는 int, String에 대해서만 연산을 시행합니다.

int 연산자의 경우 기본 기본연산자를 사용할 수 있기 때문에, 해당 연산자를 사용해 비교한 결과 값을 Value객체에 저장해 리턴합니다.

String 객체의 경우 compareTo() 메소드를 사용합니다.

str1.compareTo(str2)의 구조로,

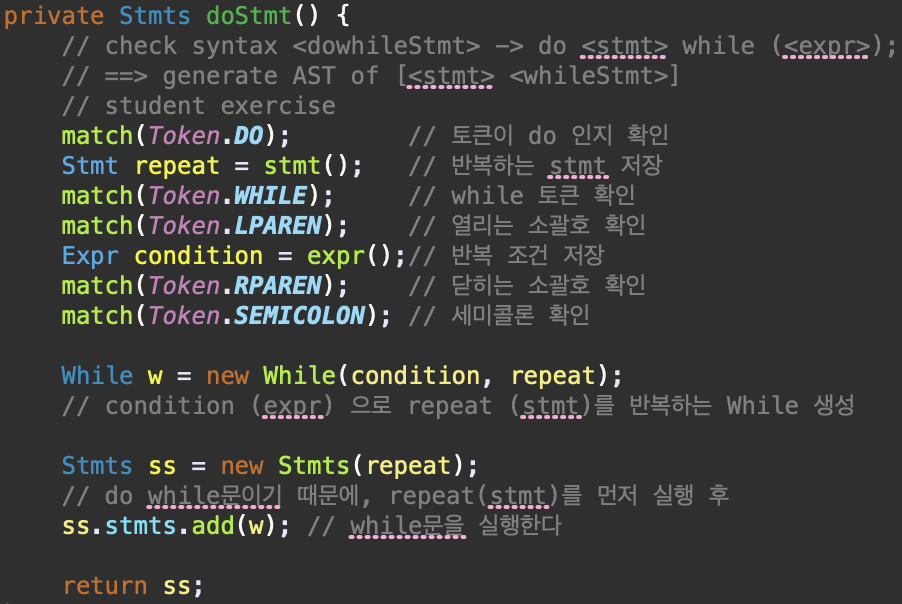
str1이 str2보다 크면, 사전순으로 뒤에 있는 의미이고 양수를 리턴합니다.

str1이 str2와 같다면, 같은 문자열로 0을 리턴합니다.

str1이 str2보다 작다면, 사전순으로 앞에 있는 의미이고 음수를 리턴합니다.

str1.compareTo(str2) 과 0을 주어진 연산자와 비교해 결과를 Value객체로 리턴합니다.

**doStmt()**



do는 stmt를 한번 실행하고 while문을 실행하는 구조입니다.

do 토큰을 확인하고 do ‘{’stmts‘}’ 의 구조이기 때문에 중괄호까지 Stmt로 저장합니다.

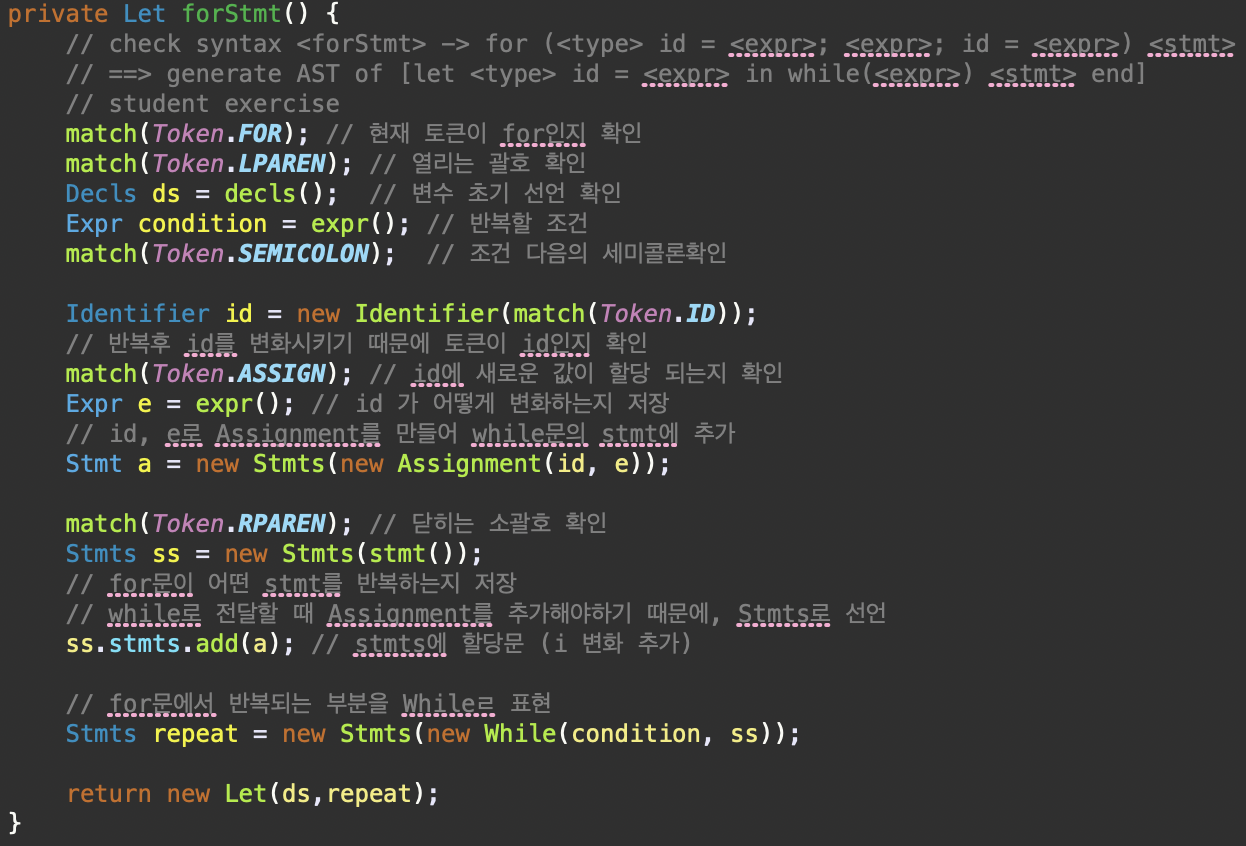
뒤에 나오는 while토큰과 ‘(’ 토큰을 확인하고 반복조건인 Expr을 저장합니다.

‘)’, ‘;’을 확인한 후 doStmt문이 종료되었으므로 하나의 Stmts문으로 묶습니다.

repeat (Stmt)로 Stmts를 생성 후, 그 뒤에 w (While)을 추가해 repeat 실행 후 w가 실행되도록 합니다.

즉 repeat이 한번 실행된 후 condition을 만족할 동안, repeat를 반복합니다.

**forStmt()**

for안의 요소를 while 구조를 포함한 Stmts로 바꾸는 과정입니다.

for ‘(’ 선언 ‘;’ 반복조건 ‘;’ 변수 최신화 ‘)’ 반복 Stmt 🡪 while문으로 재구조를 하면

let 선언 in while(반복조건) {반복 stmt + 변수 할당} 입니다

for 토큰과 ‘(’ 토큰을 확인 후 초기 ds(변수선언)을 저장합니다.

선언은 ‘;’으로 끝나기 때문에, 바로 다음에 오는 condition(반복조건) 저장

선언했던 변수를 변경(재할당) 합니다.

id와 할당 ‘=’ 토큰을 확인하고 어떤 값을 할당할지 e(표현식)을 저장합니다.

재할당 하는 과정을 하나의 Stmts로 저장합니다.

‘)’ 토큰을 확인하고 반복하는 Stmts를 저장합니다.

while에서 반복할 내용은 반복 Stmt + 변수 재할당이기 때문에,

하나의 Stmts, ss 로 합칩니다.

condition이 true인동안 ss를 반복하는 while문을 Stmts로 만든 뒤

ds와 합쳐 하나의 Let문으로 리턴합니다.

**실행결과**

**hi0.s**

|  |  |
| --- | --- |
| 텍스트, 폰트, 스크린샷이(가) 표시된 사진  자동 생성된 설명 | 문자열 “hello world!”를 출력합니다.  string type s에 “hello world!”를 저장합니다.  s에 저장된 문자열 “hello world!”를 출력합니다. |

**hi1.s**

|  |  |
| --- | --- |
|  | String변수 s를 “hello”로 선언합니다.  s안의 “hello”를 출력합니다.  “world !”를 출력합니다.  변수 s에 값을 읽습니다.  외부에서 실행한 파일로 입력을 받지는 않습니다  s안의 “hello”를 출력합니다. |

**hi2.s**

|  |  |
| --- | --- |
|  | int 변수 i,  int 변수 j가 기본값으로 초기화 됩니다.  i에 1을 할당합니다.  “2^n ?”를 출력합니다.  j를 입력받는데, 파일을 실행시켰기 때문에, 기본 값이 0으로 유지합니다.  j > 0인동안  i에 i\*2를 할당하고  j에 j-1을 할당합니다.  현재 i에 저장된 값을 출력합니다.  j가 0이므로, 2가 0번 곱해져 1을 출력합니다. |

**hi3.s**

|  |  |
| --- | --- |
|  | int 변수 i에 1을 할당합니다.  int 변수 sum에 0을 할당합니다.  int 변수 n을 기본값으로 선언합니다.  1 + 2 + … + n?을 출력합니다  n을 입력 받습니다. 외부에서 실행한 파일로, 입력 받지 않고 기본값인 0이 남아있습니다.  i <= n인 동안  sum에 sum + i를 할당합니다.  i에 i+1을 할당합니다.  sum을 출력합니다.  1~n까지의 합,  n은 0이므로 0이 출력됩니다. |

**hi4.s**

|  |  |
| --- | --- |
|  | int변수 i에 0을 할당합니다.  int변수 i가 기본값으로 할당됩니다.  int변수 j가 기본값으로 할당됩니다.  i에 10을 할당합니다.  j에 2를 할당합니다.  j가 0보다 크면,  i에 i+j를 할당합니다.  그렇지 않다면  i에 i-j를 할당합니다.  i를 출력합니다  두번째 Let문의 i로 12가 출력됩니다.  두번째 Let문이 종료되며 i와 j가 free됩니다.  남아있는 첫번째 Let문의 i를 출력합니다. |

**hi5.s**

|  |  |
| --- | --- |
| 텍스트, 스크린샷, 메뉴, 폰트이(가) 표시된 사진  자동 생성된 설명  텍스트, 스크린샷이(가) 표시된 사진  자동 생성된 설명 | int 변수 i, j, k를 선언합니다.  i와 j에 1을 할당합니다.  i가 3보다 작은동안,  j에 1을 할당합니다.  j가 4보다 작은동안  k에 i\*j를 할당합니다.  3개의 print를 파싱해  각각 i, j, k를 출력합니다.  j에 j+1을 할당합니다.  i에 i+1을 할당합니다.  i는 1부터 3까지  j는 1부터 4까지  k는 1부터 12까지 출력됩니다.  k는 1\*1 ~ 3\*4의 값이며  j가 5가 되었을 때 i를 하나 증가시키고 j를 1로 할당합니다 |

**hi6.s**

|  |  |
| --- | --- |
| 텍스트, 메뉴, 스크린샷, 폰트이(가) 표시된 사진  자동 생성된 설명  텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진  자동 생성된 설명 | int 변수 i를 0으로 선언합니다.  int 변수 i를 1로 선언합니다.  int 변수 j를 2로 선언합니다.  i를 출력합니다.  가장 최근 선언된 지역변수의 값,  1이 출력됩니다.  i가 0보다 크다면  i에 i+j를 할당합니다.  그렇지 않다면,  i에 i-j를 할당합니다  i가 1이므로  i에 i+j 즉 3이 할당됩니다.  i를 출력합니다.  3이 출력됩니다.  int 변수 k를 3으로 선언합니다.  i에 k를 할당합니다.  k가 free됩니다.  i를 출력합니다.  k의 값이었던 3이 출력됩니다.  i, j가 free됩니다.  i가 free됩니다. |

**hi7.s**

|  |  |
| --- | --- |
|  | int 변수 i를 0으로 선언합니다.  int 변수 i를 1로 선언합니다.  int 변수 j를 1로 선언합니다.  bool 변수 k를 true로 선언합니다.  i를 출력합니다 나중에 선언된 지역변수,  1이 출력됩니다.  k가 true면 i에 i+j를 할당하고  그렇지 않다면 i에 i-j를 할당합니다.  k가 true이므로 i는 i+j 즉 2를 할당받습니다.  i를 출력합니다  2가 출력됩니다.  i, j, k가 free됩니다.  int변수 k를 0으로 선언합니다  i에 i+k를 할당합니다.  0+0인 0이 할당됩니다  i를 출력합니다.  0이 출력됩니다. |

**stringrelop1.s stringrelop2.s**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

String변수 i에 apple, j에 banana를 할당해 비교합니다

stringrelop1.s에서 두 문자열이 다르므로 strings are not equal이 출력됩니다.

stringrelop2.s에서 banana가 사전상 뒤에 있으므로,  
banana is located behind the dictionary가 출력됩니다.

**logicalop1.s logicalop2.s**

텍스트, 스크린샷, 폰트, 메뉴이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

logicallop1.s에서 i, j는 모두 true입니다. 두 변수 를 and 연산을 실행해 true이므로

both are true가 출력됩니다

logicallop2.s에서 i는 true, j는 false입니다. 두 변수 를 or 연산을 실행해 true이므로

one or both are true가 출력됩니다.

**for.s**

|  |  |
| --- | --- |
| 텍스트, 스크린샷, 폰트, 메뉴이(가) 표시된 사진  자동 생성된 설명 | for 문이 while문으로 재구성 되어 실행됩니다.  for(int i =0; i < 10; i++) print i  ->  let int i = 0 in while(i<10){print i; i = i+1} |

**dowhile.s**

|  |  |
| --- | --- |
|  | do while문이  {stmt while}으로 재구성되어 실행됩니다.  do {  print i;  i = i - 1;  } while (i > 0);  ->  {  print i;  i = i-1;  while (i > 0) {  print i;  i = i-1;  }  } |