**2017년 1학기**

**프로그래밍언어개론 설계 계획서**

**(팀이름) : 자... 이제 누가 버스기사지???**

<작업의 분할>

Item 1

-각자 구현

Item 2

a. define 테이블 생성 및 구현 (동하)

b. insertTable 구현 (동하)

c. lookupTable 구현 (진수)

d.기존 코드에서 INT Type 대신 ID Type입력시 Table에서 꺼내오도록 수정

(cons ~ plus 부분 -> 동하) (minus ~ eq 부분 -> 진수)

=> run\_expr 수정으로 모두 적용

e. lambda 구현

(평일 짬짬이 하며 주말에 같이 동방에서 구현)

Symbol Table 추가

Define에서 Lambda 처리

Lambda 변수 바인딩 및 삭제

Lambda 변수 여러 개 처리

Lambda 옵션 적용

구현 일정 계획

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 작업내용 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 6.1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 팀결성 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Item1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Item2 a |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Item2 b |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Item2 c |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Item2 d |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Item2 e |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| refactoring |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Bug fix |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

* 1. 팀 구성

|  |  |  |  |
| --- | --- | --- | --- |
| **학번** | **이름** | **역할** | |
| **Item1** | **Item2** |
| **201202141** | **김동하** | **All** | **defTable ,insertTable구현, Symbol Table 생성, Lambda 기본구현, Lambda 옵션 처리,**  **test, refactoring** |
| **201203391** | **임진수** | **All** | **lookupTable구현, run\_expr 수정, Lambda 변수 바인딩 및 삭제. Lambda 다중 변수 처리,**  **test, refactoring** |

\* 팀장 (조교가 연락을 취할 수 있는 대표) : 김동하

1-3 수정 로그

17.05.21 Item1 구현 Test 코드 삭제 및 input 추가

17.05.21 define 구현

17.05.22 insertTable 구현

17.05.22 lookupTable 구현

* + 1. 최종 완료
  1. 문제 해결방식

Define )

l\_node은 define의 key 값이 되고 r\_node의 값은 value 값이 됩니다. Define을 할 때는 기본적을 value 값을 run\_expr 해서 리스트 내부의 값을 실행한 후 quote 된 값만 저장을 합니다. 하지만 저장해야할 value가 lambda 라면 실행을 하지 않고 lambda 식을 저장합니다.

insertTable )

defTable[id] = value 이런 식으로 l\_node의 value(해당 ID 노드의 값)을 id로 받아 key로 사용하고 r\_node를 value로 해서 저장합니다.

lookupTable )

defTable에 id라는 key 값을 가진 페어가 있다면 저장된 value를 리턴하고 없으면 none을 리턴하도록 했습니다.

ID값 복원 )

Define으로 정의한 ID를 표현식에서 다시 복원을 하도록 run\_expr 함수에서 기존 id 타입이면 해당 노트를 리턴하고 끝이였는데 그 곳에서 id이면 lookupTable 함수를 이용하여 값이 있을 경우 해당 value를 리턴하여 각 오퍼레이션 함수 내부에서 run\_expr()을 실행하여 new\_node를 만들 때 자동으로 치환 되도록 구현을 하였습니다.

Lambda )

Lambda의 기본 모형은 argument까지 포함하였을 때 ((lambda (x) (defines) (expression)) argument) 이런 모형입니다. 여기서 기존 다른 오퍼레이션과 다르게 lambda는 리스트 내부가 아닌 밖에 argument를 가져와서 함수에서 사용을 합니다. 그래서 lambda 함수를 실행하기전에 run\_list에서 내부에 lambda가 있는지 검사를 한 후 argument에 대하여 처리를 한 후 lambda 식으로 넘어가서 argument를 받아서 사용하도록 구현하였습니다.

insertValue

root\_node에서 변수 부분과 값 부분, 빈테이블 saveTable을 파라미터로 받습니다.

받은 변수와 값을 defTable에 넣습니다. 넣기 전에 이미 defTable에 변수의 값이 존재한다면 saveTable에 값을 저장해두고 defTable에 넣습니다.

removeValue

root\_node에서 변수 부분과 값 부분,insertValue에서 사용한 saveTable을 파라미 터로 받습니다. saveTable에 변수의 값이 존재하지 않는다면 defTable에서 변수 값을 지우고 존재한다면 saveTable의 값으로 변경합니다.

Func

Opcode가 미리 정의된 Symbol table에 존재하지 않는다면 func로 넘어가도록 symbol table에 값을 넣습니다. 그 값에 의해 func가 실행됩니다.

func에서는defTable에 node의 value의 value 즉 Symbol table의 key 값이 defTable에 존재 한다면 그 값을 꺼내와 치환 후 run\_expr을 실행합니다.

존재하지 않는다면 Error를 출력합니다.

동작과정

입력한 값이 Node로 변환되어 run\_expr로 넘어올 경우 제일 첫 노드의 값에 따라서 반환하게 됩니다. 그중 List일 경우에는 run\_list를 반환합니다.

run\_list 에서는 list 내부의 첫 값이 list이고 또 그 list안의 value가 lambda일 경우 insertValue를 실행 후 lambda의 식의 결과값을 run\_expr을 통해 받아옵니다. 그후에는 removeValue를 실행하여 lambda에서만 사용되는 변수 값을 제거합니다. 제거후에는 run\_expr의 결과를 반환합니다.

그 외의 경우에는 첫 값을 opcode로 넘겨 심볼 테이블에 의해 처리된 결과를 반환합니다.

Lambda 구현

전역 함수 호출이 가능

함수 내에서 전역 함수 호출 가능

변수 scope 구현

Nested 함수 구현 가능

Recursion 함수 구현 가능

함수에서 함수를 인자로 사용가능