start : 시작변수

main\_statement : main함수를 포함한 stmt

global\_statement : main함수 밖의 stmt

global\_assign\_statement: main함수 밖의 stmt에서 전역변수 선언 부분

func\_statement : 함수의 선언.

returnable\_statement : return이 나타날 수 있는 statement

else\_statement : else가 나타날 수 있는 statement

var\_statement : 매개변수를 선언하는 statement

def\_statement : 정의문 statement( := )

for\_statement : continue, break가 나타날 수 있는 statement

case\_statement : case, default가 나타날 수 있는 statement

case\_without\_default\_statement : case가 나타날 수 있는 statement(default 사용 X)

start : KPACKAGE KMAIN KIMPORT string main\_statement

시작변수 -> package main \n import “fmt” 를 해야 main함수에 접근

main\_statement : main함수 전후로 전역변수 선언, 함수 선언이 이루어질 수 있음. main함수 밖의 코드는 gloabl\_statament

global\_statement: main함수 밖에서 이루어질 수 있는 것은 전역변수 선언과 함수 선언

global\_assign\_statement: 전연변수 선언을 위한 stmt.

전역변수 선언은 var x int와 같이 단순 선언과 var x int = 2와 같이 선언 후 대입으로 나뉨.

전역변수를 선언 하면, global\_names에 저장됨. 지역변수로 선언되지 않더라도 전역변수로 선언되었다면 해당 변수와 value를 사용할 수 있음

함수 선언

함수 선언은 func 함수이름 ( 매개변수 선언 ) { 함수 내용 }으로 이루어져 있음

매개변수 선언은 0~N개의 매개변수가 각각 ID type , 의 형태로 나열됨. (x int, y int)

함수 내용은 일반적인 stmt와 동일함. 다만, return을 사용할 수 있음.

지역변수 선언

지역변수는 전역변수와 같이 단순 선언과 선언 후 대입 모두 가능함.

그리고 이미 선언된 지역변수에 새로운 값을 대입할 수 있음.

문자열 처리

문자열이란 큰 따옴표 두 개 사이에 놓여진 임의의 문자열을 말함. ( “ statement “ )

그러나 직접 돌려보았을 때, 잘 인식되지 않는 것 같음. 수정필요

if문

if문은 if 조건문 { statement } else\_statement로 구성됨

else\_statement란, else가 나올 수 있는 statment임. ~~else는 if 에서 파생된 ‘}’ 와 같은 줄에 있어야 함(미구현)~~

마찬가지로 조건문과 ‘{’은 같은 줄에 있어야 함(미구현)

switch문

switch문은 크게 세 가지 용법이 있음.

1. switch { case 조건문 }

2. switch ID { case ID가 취할 수 있는 값 }

3. switch ID { case ID가 취할 수 있는 type }

세 경우 모두 하나의 switch문 안에 하나의 default만 사용할 수 있음.

2번과 3번의 경우, 하나의 case문에 여러 개의 값과 type이 들어갈 수 있음.

정의문

:=를 이용한 stmt를 정의문이라 함.

ID := expression으로 구현하였으며, 일단 expression만 들어갈 수 있도록 하였음(원래 함수의 리턴값 등 다양한 값에 들어갈 수 있음)

for문

for문은 크게 네 가지 용법이 있음

1. for 초기 조건 ; 조건문 ; 종단 조건 { for문 내용 }

2. for 조건문 { for문 내용 }

3. for range문 { for문 내용 }

4. for { for문 내용 } // while(true)와 같은 효과

초기 조건, 종단 조건, range문 처리 미구현

for문 내용은 일반적인 statement에 continue와 break를 사용할 수 있음.

for문 안에 정의문이 같이 들어가는 경우 미구현

BLANK

BLANK = ( |\t)\*

NEWLINE = BLANK \n BLANK

이렇게 정의하고 lex를 돌려보니 오류... 좀 더 생각해 봐야 할 것 같음