%{

#include <stdio.h>

#include <iostream>

int yylex(void);

void yyerror(const char \*s) {

fprintf (stderr, "%s %s\n", "Found error: ", s);

/\*printf("\n");

printf("\n");

printf(s);

printf("\n");

printf("\n");\*/

}

%}

%token END\_OF\_PRGM

%token END\_OF\_MAIN

%token MAIN

%token FUNCTION

%token FUNCT\_NAME

%token END\_OF\_DEF

%token USR\_DEF\_SHAPE

%token USR\_SHAPE\_NAME

%token INHERITANCE

%token ID

%token DRAW

%token SHAPE

%token END\_OF\_INPUT

%token POSITION

%token POS\_X\_POS\_Y

%token WIDTH

%token NUM

%token HEIGHT

%token STROKE

%token COLOR

%token RND\_CRNRS

%token NAME

%token START

%token END

%token ARROW\_STRT

%token ARROW\_END

%token DIR

%token DIR\_VAL

%token RED

%token BLUE

%token GREEN

%token YELLOW

%token BLACK

%token WHITE

%token IF

%token PARAMETER

%token AND

%token OR

%token GRTR

%token LESS

%token LOG\_EQ

%token GRTR\_EQ

%token LESS\_EQ

%token ELSE

%token ASSIGNMENT

%token BEGIN\_OF\_BLOCK

%token END\_OF\_BLOCK

%token WHILE

%token FOR

%token RETURN

%token TRUE

%token FALSE

%token USR\_DEF\_VAR

%token OP

%token ACCESS

%token PARAM\_DELIM

%token FILL\_STATE

%token FILL\_COLOR

%token BORDR\_STATE

%token BORDR\_COLR

%token TYPECAST

%token STRING

%token CHAR

%token RGB

%token USR\_DEF\_PARAM

%%

letter\_declaration:

main\_declaration letter\_tail

| error "." {yyerror ("No exclamation mark to denote the end of file!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

;

letter\_tail:

post\_declaration END\_OF\_PRGM

| END\_OF\_PRGM

| post\_declaration error {yyerror ("No exclamation mark to denote the end of file!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

| error {yyerror ("No exclamation mark to denote the end of file!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

;

main\_declaration:

MAIN block\_statement END\_OF\_MAIN

| block\_statement END\_OF\_MAIN {yyerror ("Main should start with Dear Kakubo!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

| MAIN block\_statement error {yyerror ("Main should end with Best Wishes!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

;

post\_declaration:

ps post\_tail

| pps post\_tail

;

post\_tail:

ps

| pps

|

;

statement:

functional\_statement statement\_tail

| if\_statement statement\_tail

| loop\_statement statement\_tail

| assignment statement\_tail

;

statement\_tail:

statement

|

;

ps:

FUNCTION FUNCT\_NAME ID expression END\_OF\_DEF ps\_statement

FUNCTION ID expression END\_OF\_DEF ps\_statement {yyerror ("P.S. should start with I defined!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

FUNCTION FUNCT\_NAME ID expression ps\_statement {yyerror ("P.S. should end with as the following!"); printf("P.S. should end with as the following!"); yyerrok;}

;

ps\_statement:

statement

| block\_statement

;

pps:

USR\_DEF\_SHAPE USR\_SHAPE\_NAME ID expression pps\_tail

| FUNCT\_NAME USR\_SHAPE\_NAME ID expression pps\_tail {yyerror ("P.S. used instead of P.P.S.!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

;

pps\_tail:

INHERITANCE ID END\_OF\_DEF pps\_statement

| END\_OF\_DEF pps\_statement

;

pps\_statement:

user\_def\_variable pps\_statement

| statement

| block\_statement

;

user\_def\_variable:

ID USR\_DEF\_VAR ID expression

|

;

functional\_statement:

ID expression

| DRAW functional\_tail

| functional\_tail {yyerror ("Draw keyword missing in draw function!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

;

functional\_tail:

ID expression

| SHAPE expression

;

expression:

parameter expression\_tail

;

expression\_tail:

expression END\_OF\_INPUT

| END\_OF\_INPUT

;

parameter:

POSITION POS\_X\_POS\_Y

| WIDTH term

| HEIGHT term

| STROKE term

| COLOR color

| RND\_CRNRS

| NAME ID

| START POS\_X\_POS\_Y

| END POS\_X\_POS\_Y

| ARROW\_STRT POS\_X\_POS\_Y

| ARROW\_END POS\_X\_POS\_Y

| DIR DIR\_VAL

| FILL\_STATE TRUE

| FILL\_STATE FALSE

| FILL\_COLOR color

| BORDR\_STATE TRUE

| BORDR\_STATE FALSE

| BORDR\_COLR color

| USR\_DEF\_PARAM ID

|

;

term:

NUM

| ID

;

color:

RED

| BLUE

| GREEN

| YELLOW

| BLACK

| WHITE

| RGB

;

if\_statement:

IF PARAMETER conditional PARAMETER if\_tail

;

conditional:

condition conditional\_tail

;

conditional\_tail:

AND conditional

| OR conditional

|

;

condition:

ID condition\_tail

;

condition\_tail:

GRTR condition

| LESS condition

| LOG\_EQ condition

| GRTR\_EQ condition

| LESS\_EQ condition

|

;

if\_tail:

statement else\_tail

| block\_statement else\_tail

;

else\_tail:

ELSE statement

ELSE block\_statement

|

;

assignment:

ID ASSIGNMENT assignment\_tail

| ID ACCESS ASSIGNMENT assignment\_tail

;

assignment\_tail:

NUM

| ID

| TRUE

| FALSE

| color

| OP operand

;

operand:

NUM

| ID

;

block\_statement:

BEGIN\_OF\_BLOCK statement END\_OF\_BLOCK

| statement END\_OF\_BLOCK {yyerror ("Block does not begin with Please!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

| BEGIN\_OF\_BLOCK statement {yyerror ("Block does not end with Thank You!"); printf("No exclamation mark to denote the end of file!"); yyerrok;}

;

loop\_statement:

WHILE PARAMETER conditional PARAMETER loop\_tail

| NUM FOR PARAMETER loop\_tail

;

loop\_tail:

statement

| block\_statement

;

return\_statement:

RETURN ID

| RETURN NUM

;

%%

int main(void) {

yyparse();

return 0;

}