Gramática JSON

```
Init            ::= Object;

Object          ::=
                    TK_LBRC TK_RBRC
                |   TK_LBRC Contents TK_RBRC


Contents        ::=
                    Contents Content
                |   Content


Content         ::=
                    TK_STRING TK_COLON ValueType TK_COMA
                |   TK_STRING TK_COLON ValueType



ValueType       ::=
                    TK_STRING:v
                |   TK_DOUBLE:v
                |   TK_INT:v
                |   TK_MINUS:s TK_INT:v
                |   TK_MINUS:s TK_DOUBLE:v
```

Gramática StatPy

```
Init            ::= Main

Main            ::= KW_VOID KW_MAIN TK_LPAR TK_RPAR TK_LBRC Instructions TK_RBRC
                |   KW_VOID KW_MAIN TK_LPAR TK_RPAR TK_LBRC TK_RBRC


Instructions    ::=
                    Instructions Instruction
                |   Instruction:i

GraphFuncs      ::=
                    BarGraph
                |   PieGraph


PieGraph        ::= KW_VOID KW_PIEGRPH TK_LPAR TK_RPAR TK_LBRC PieGraphInsts TK_RBRC

BarGraph        ::= KW_VOID KW_BARGRPH TK_LPAR TK_RPAR TK_LBRC BarGraphInsts TK_RBRC

PieGraphInsts   ::=
                    PieGraphInsts PieGraphInst
                |   PieGraphInst
```

```
PieGraphInst    ::=
                    KW_STRING KW_TITLE:t TK_EQ TK_ID:v TK_SCOLON
                |   KW_STRING KW_TITLE:t TK_EQ TK_STRING:v TK_SCOLON
                |   KW_STRING KW_TITLE:t TK_EQ CallJson:v TK_SCOLON
                |   KW_STRING TK_LBRK TK_RBRK KW_XAXIS:x TK_EQ TK_LBRC ArgsList:a TK_RBRC TK_SCOLON
                |   KW_DOUBLE TK_LBRK TK_RBRK KW_VALUES:v TK_EQ TK_LBRC ArgsList:a TK_RBRC TK_SCOLON

BarGraphInsts   ::=
                    BarGraphInsts:s BarGraphInst:i
                |   BarGraphInst:i

BarGraphInst    ::=
                    KW_STRING KW_TITLE:t TK_EQ TK_ID:id TK_SCOLON
                |   KW_STRING KW_TITLE:t TK_EQ TK_STRING:s TK_SCOLON
                |   KW_STRING KW_TITLE:t TK_EQ CallJson:v TK_SCOLON
                |   KW_STRING KW_TITLEX:tx TK_EQ TK_STRING:s TK_SCOLON
                |   KW_STRING KW_TITLEX:tx TK_EQ TK_ID:id TK_SCOLON
                |   KW_STRING KW_TITLEX:tx TK_EQ CallJson:v TK_SCOLON
                |   KW_STRING KW_TITLEY:ty TK_EQ TK_STRING:s TK_SCOLON
                |   KW_STRING KW_TITLEY:ty TK_EQ TK_ID:id TK_SCOLON
                |   KW_STRING KW_TITLEY:ty TK_EQ CallJson:v TK_SCOLON
                |   KW_STRING TK_LBRK TK_RBRK KW_XAXIS:x TK_EQ TK_LBRC ArgsList:a TK_RBRC TK_SCOLON
                |   KW_DOUBLE TK_LBRK TK_RBRK KW_VALUES:v TK_EQ TK_LBRC ArgsList:a TK_RBRC TK_SCOLON

GlobalVarsFunc  ::=
                    KW_VOID KW_DEFGLB:g TK_LPAR TK_RPAR GlobalVarsEnv:genv

GlobalVarsEnv   ::=
                    TK_LBRC GlobalVarsInsts:i TK_RBRC
                |   TK_LBRC TK_RBRC


GlobalVarsInsts ::=
                    GlobalVarsInsts:s GlobalVarsInst:i
                |   GlobalVarsInst:i
```

```
GlobalVarsInst  ::=
                    KW_STRING TK_ID:id TK_EQ TK_STRING:str TK_SCOLON
                |   KW_DOUBLE TK_ID:id TK_EQ TK_DOUBLE:db TK_SCOLON
                |   KW_STRING TK_ID:id TK_EQ CallJson:js TK_SCOLON
                |   KW_DOUBLE TK_ID:id TK_EQ CallJson:js TK_SCOLON


Instruction     ::=
                    DeclareVar:i TK_SCOLON
                |   DeclareArr:i TK_SCOLON
                |   CallFunc:i TK_SCOLON
                |   IfStatement:i
                |   SwitchStatement:i
                |   ForLoop:i
                |   WhileLoop:i
                |   Print:i
                |   DoWhileLoop:i TK_SCOLON
                |   ReasignVar:i TK_SCOLON
                |   IncreDecre:i TK_SCOLON
                |   KW_BREAK:i TK_SCOLON
                |   KW_RETURN:i TK_SCOLON
                |   KW_RETURN:i Expresion:exp TK_SCOLON
                |   KW_CONTINUE:i TK_SCOLON
                |   GlobalVarsFunc:i
                |   GraphFuncs:i
                |   error
                ;


DeclareVar      ::=
                    DataType TK_ID:id TK_EQ Expresion:exp
                |   DataType TK_ID:id

DeclareArr      ::=
                    DataType TK_LBRK TK_RBRK TK_ID:id TK_EQ TK_LBRC ArgsList:arg TK_RBRC

DataType        ::=
```

```
                    KW_INT
                |   KW_BOOL
                |   KW_DOUBLE
                |   KW_CHAR
                |   KW_STRING
                ;

ReasignVar      ::= TK_ID:id TK_EQ Expresion:exp

//DeclareFunc     ::= KW_VOID TK_ID TK_LPAR TK_RPAR LocalEnv;

CallFunc        ::=
                    TK_ID:id TK_LPAR ArgsList:args TK_RPAR
                |   TK_ID:id TK_LPAR TK_RPAR

ArgsList        ::=
                    ArgsList:l TK_COMA Expresion:e
                |   Expresion:e

IfStatement     ::=
                    KW_IF:f TK_LPAR Expresion:exp TK_RPAR LocalEnv:env KW_ELSE IfStatement:f2
                |   KW_IF:f TK_LPAR Expresion:exp TK_RPAR LocalEnv:env KW_ELSE LocalEnv:env2  {
                |   KW_IF:f TK_LPAR Expresion:exp TK_RPAR LocalEnv:env

LocalEnv        ::=
                    TK_LBRC:l Instructions:i TK_RBRC
                |   TK_LBRC:l TK_RBRC

SwitchStatement ::=
                    KW_SWITCH:s TK_LPAR Expresion:exp TK_RPAR LocalEnvs:env

LocalEnvs       ::=
                    TK_LBRC CaseDeft:cd TK_RBRC
                |   TK_LBRC TK_RBRC

CaseDeft        ::=
                    Cases:c Default:d
```

```
                |   Cases:c
                |   Default:d

Cases           ::=
                    Cases:cs Case:c
                |   Case:c


Case            ::=
                    KW_CASE:c Expresion:exp TK_COLON Instructions:i
                |   KW_CASE:c Expresion:exp TK_COLON


Default         ::=
                    KW_DEFAULT:d TK_COLON Instructions:i
                |   KW_DEFAULT:d TK_COLON

ForLoop         ::=
                    KW_FOR:f TK_LPAR ForArgs:fa TK_RPAR LocalEnv:env


ForArgs         ::= ForInitArg:fi TK_SCOLON Expresion:exp TK_SCOLON IncreDecre:inc


ForInitArg      ::= KW_INT TK_ID:id TK_EQ TK_INT:i
                |   ReasignVar:r


IncreDecre      ::=
                    TK_ID:id TK_INC
                |   TK_ID:id TK_DEC

WhileLoop       ::= KW_WHILE:w TK_LPAR Expresion:exp TK_RPAR LocalEnv:env

DoWhileLoop     ::= KW_DO:d LocalEnv:env KW_WHILE:w TK_LPAR Expresion:exp TK_RPAR

CallJson        ::=
```

```
                    TK_DOLL:d TK_LBRC KW_NVAL TK_COMA JsonArg:f TK_COMA JsonArg:k TK_RBRC {

JsonArg         ::=
                    TK_ID:id
                |   TK_STRING:s


Expresion       ::=
                    Math:e
                |   Relation:e
                |   Logic:e
                |   CallFunc:e
                |   CallJson:e
                |   IncreDecre:e
                |   TK_ID:e
                |   TK_STRING:e
                |   TK_CHAR:e
                |   TK_INT:e
                |   TK_DOUBLE:e
                |   KW_TRUE:e
                |   KW_FALSE:e
                |   TK_LPAR Expresion:e TK_RPAR


Math            ::=
                    Expresion:e1 TK_PLUS:s Expresion:e2
                |   Expresion:e1 TK_MINUS:s Expresion:e2
                |   Expresion:e1 TK_MULT:s Expresion:e2
                |   Expresion:e1 TK_DIV:s Expresion:e2
                |   TK_MINUS:s Expresion:e1


Relation        ::=
                    Expresion:e1 TK_GEQ:s Expresion:e2
                |   Expresion:e1 TK_LEQ:s Expresion:e2
                |   Expresion:e1 TK_EQEQ:s Expresion:e2
                |   Expresion:e1 TK_NOTEQ:s Expresion:e2
```

```
                |   Expresion:e1 TK_GREATER:s Expresion:e2
                |   Expresion:e1 TK_LESS:s Expresion:e2

Logic           ::=
                    Expresion:e1 TK_AND:s Expresion:e2
                |   Expresion:e1 TK_OR:s Expresion:e2
                |   TK_NOT:s Expresion:e


Print           ::=
                    KW_PRINT:p TK_LPAR Expresion:exp TK_RPAR TK_SCOLON
                |   KW_PRINT:p TK_LPAR TK_RPAR TK_SCOLON
```