COLON => :
SC => ;
LSQ => [
RSQ =>]
LCB => {
RCB => }
e => epsilon
num => constant value
var_name => variable

For Bhavya and Manas : Do not modify the grammar. If any problem is there comment there or msg in group

ISSUES/DOUBTS for Vandana Madam : next line? Do we need to explicitly account for delimiter (SPACE) ?

- Array type operations (addition allowed with same subranges / different subranges / same variables declared together or not)?
- Array type assignment (a1=a2 allowed or not? a1=b1(same subranges) ? with different subranges)
- Comment symbol in this language?
- Clarify page 6 last para of assignment
- Can declaration and assignment statements come alternatively? Or first block will have all declaration statements and then all the assignment statements
- Same variable name declarations ? error or not. If no error final declaration to be used?
- More than one Blank space ? error or not
- Brackets operations in assignment
- Should we care about some type errors in the grammar itself ? or only during the type checking (currently we separated boolean and arithmetic operations)
- Type errors vs syntax errors (a+b ||| c)? Whether we should make any distinction?
- How to initialize boolean var , is it by integer 0 1
- Jagged2d and Jagged3d separate or not ? which would be better

Linking both declaration and assignment statements

```
<start>-><gen-dec_block> <assign_block>
<gen-dec_block> -> <gen-dec><gen-dec_block>|<gen-dec>
<assign_block> -> <assign_stmnt><assign_block>|<assign_stmnt>
```

Declaration Statements
<gen-dec> -> declare var_name COLON <type> | declare list of variables <var_names> COLON <type>

<var_names> -> var_name | var_name <var_names>

<type> -> integer SC | real SC | Boolean SC | <Jagarr-type> | <Rectarr-type>

<Jagarr-type> -> jagged array <dims_J> of integer SC <populate>

<dims_J> -> LSQ num..num RSQ LSQ RSQ <brackets>

<brackets> -> LSQ RSQ | e

<populate> -> R1 LSQ num RSQ COLON size num COLON values <vals> <populate>

<vals> -> LCB <val_ext> num <nex> RCB

<nex> -> num <nex> | e

<val_ext> -> num <nex> SC <val_ext> | e

<Rectarr-type> -> array <dims_R> of integer SC

<dims_R> -> LSQ <var_Ind>..<var_Ind> RSQ | LSQ <var_Ind>..<var_Ind> RSQ <dims_R>

<var_Ind> -> var_name | num

Assignment statements

<assign_stmt> -> <arithmetic_expr> | <bool_expr>
<arithmetic_expr> -> <gen_var_name> = <expr1> SC

<expr1> -> <expr1> + <term1>
<expr1> -> <expr1> - <term1>
<expr1> -> <term1>

<term1> -> <term1> * <var>
<term1> -> <term1> / <var>
<term1> -> <var>

<bool_expr> -> var_name = <expr2> SC
<expr2> -> <expr2> ||| <term2>
<expr2> -> <term2>

<term2> -> <term2> &&& var_name
<term2> -> var_name

<var> -> <gen_var_name> | num
<gen_var_name> -> var_name | var_name LSQ num <nex> RSQ

COLON => :
SC => ;
LSQ => [
RSQ =>]
LCB => {
RCB => }
e => epsilon
num => constant value
var_name => variable

For Bhavya and Manas : Do not modify the grammar. If any problem is there comment there or msg in group

ISSUES/DOUBTS for Vandana Madam : next line? Do we need to explicitly account for delimiter (SPACE) ?
Array type operations (addition allowed with same subranges / different subranges / same variables declared together or not)?
Array type assignment (a1=a2 allowed or not? a1=b1(same subranges) ? with different subranges)
Comment symbol in this language?
Clarify page 6 last para of assignment
Can declaration and assignment statements come alternatively? Or first block will have all declaration statements and then all the assignment statements
Same variable name declarations ? error or not. If no error final declaration to be used?
More than one Blank space ? error or not
Brackets operations in assignment
Should we care about some type errors in the grammar itself ? or only during the type checking (currently we separated boolean and arithmetic operations)
Type errors vs syntax errors (a+b ||| c)? Whether we should make any distinction?
How to initialize boolean var , is it by integer 0 1
Jagged2d and Jagged3d separate or not ? which would be better

Linking both declaration and assignment statements

<start>-><gen-dec_block> <assign_block>
<gen-dec_block> -> <gen-dec><gen-dec_block>|<gen-dec>
<assign_block> -> <assign_stmnt><assign_block>|<assign_stmnt>

Declaration Statements
<gen-dec> -> declare var_name COLON <type> | declare list of variables <var_names> COLON <type>

<var_names> -> var_name | var_name <var_names>

<type> -> integer SC | real SC | Boolean SC | <Jagarr-type> | <Rectarr-type>

<Jagarr-type> -> jagged array <dims_J> of integer SC <populate>

<dims_J> -> LSQ num..num RSQ LSQ RSQ <brackets>

<brackets> -> LSQ RSQ | e

<populate> -> R1 LSQ num RSQ COLON size num COLON values <vals> <populate>

<vals> -> LCB <val_ext> num <nex> RCB

<nex> -> num <nex> | e

<val_ext> -> num <nex> SC <val_ext> | e

<Rectarr-type> -> array <dims_R> of integer SC

<dims_R> -> LSQ <var_Ind>..<var_Ind> RSQ | LSQ <var_Ind>..<var_Ind> RSQ <dims_R>

<var_Ind> -> var_name | num

Assignment statements

```
<assign_stmt> -> <arithmetic_expr> | <bool_expr>
<arithmetic_expr> -> <gen_var_name> = <expr1> SC

<expr1> -> <expr1> + <term1>
<expr1> -> <expr1> - <term1>
<expr1> -> <term1>

<term1> -> <term1> * <var>
<term1> -> <term1> / <var>
<term1> -> <var>

<bool_expr> -> var_name = <expr2> SC

<expr2> -> <expr2> ||| <term2>
<expr2> -> <term2>

<term2> -> <term2> &&& var_name
<term2> -> var_name

<var> -> <gen_var_name> | num
<gen_var_name> -> var_name | var_name LSQ num <nex> RSQ
```

```
COLON => :
SC => ;
LSQ => [
RSQ => ]
LCB => {
RCB => }
 e => epsilon
 num => constant value
 var_name => variable
```

For Bhavya and Manas : Do not modify the grammar. If any problem is there comment there or msg in group

ISSUES/DOUBTS for Vandana Madam : next line? Do we need to explicitly account for delimiter (SPACE) ?
Array type operations (addition allowed with same subranges / different subranges / same variables declared together or not)?

Array type assignment (a1=a2 allowed or not? a1=b1(same subranges) ? with different subranges)
Comment symbol in this language?
Clarify page 6 last para of assignment
Can declaration and assignment statements come alternatively? Or first block will have all declaration statements and then all the assignment statements
Same variable name declarations ? error or not. If no error final declaration to be used?
More than one Blank space ? error or not
Brackets operations in assignment
Should we care about some type errors in the grammar itself ? or only during the type checking (currently we separated boolean and arithmetic operations)
Type errors vs syntax errors (a+b ||| c)? Whether we should make any distinction?
How to initialize boolean var , is it by integer 0 1
Jagged2d and Jagged3d separate or not ? which would be better

Linking both declaration and assignment statements

<start>-><gen-dec_block> <assign_block>
<gen-dec_block> -> <gen-dec><gen-dec_block>|<gen-dec>
<assign_block> -> <assign_stmnt><assign_block>|<assign_stmnt>

Declaration Statements
<gen-dec> -> declare var_name COLON <type> | declare list of variables <var_names> COLON <type>

<var_names> -> var_name | var_name <var_names>

<type> -> integer SC | real SC | Boolean SC | <Jagarr-type> | <Rectarr-type>

<Jagarr-type> -> jagged array <dims_J> of integer SC <populate>

<dims_J> -> LSQ num..num RSQ LSQ RSQ <brackets>

<brackets> -> LSQ RSQ | e

<populate> -> R1 LSQ num RSQ COLON size num COLON values <vals> <populate>

```
<vals> -> LCB <val_ext>  num <nex> RCB

<nex> -> num <nex> | e

<val_ext> -> num <nex> SC <val_ext> | e

<Rectarr-type> -> array <dims_R> of integer SC

<dims_R> -> LSQ <var_Ind>..<var_Ind> RSQ | LSQ <var_Ind>..<var_Ind> RSQ <dims_R>

<var_Ind> -> var_name | num
```

Assignment statements

```
<assign_stmt> -> <arithmetic_expr> | <bool_expr>
<arithmetic_expr> -> <gen_var_name> = <expr1> SC

<expr1> -> <expr1> + <term1>
<expr1> -> <expr1> - <term1>
<expr1> -> <term1>

<term1> -> <term1> * <var>
<term1> -> <term1> / <var>
<term1> -> <var>

<bool_expr> -> var_name = <expr2> SC

<expr2> -> <expr2> ||| <term2>
<expr2> -> <term2>

<term2> -> <term2> &&& var_name
<term2> -> var_name

<var> -> <gen_var_name> | num
<gen_var_name> -> var_name | var_name LSQ num <nex> RSQ
```

```
<expr2> -> <expr2> ||| <term2>
```

```
<expr2> -> <term2>

<term2> -> <term2> &&& var_name
<term2> -> var_name

<var> -> <gen_var_name> | num
<gen_var_name> -> var_name | var_name LSQ num <nex> RSQ
```