

```

● IMPORT namespace
● ENUM name
    member_def1    ,value
    member_def2    ,2
    member_defn    ,6
ENDENUM

```

## Data Division

```

● MAIN name
● .INCLUDE filename

● RECORD recordName
    ;var      ,typeSize ,literal
    alph     ,a3        ,'abc'
    dec      ,d5
    ● GROUP groupName ,type
        name      ,a20
        position  ,a20
    ENDGROUP
ENDRECORD

```

```

● STRUCTURE structureName
    var      ,typeSize ,literal
ENDSTRUCTURE

```

```

● COMMON commonName
    var      ,typeSize ,literal
ENDCOMMON

```

```

● LITERAL literalName
    var      ,typeSize ,literal
ENDLITERAL

```

```

● GLOBAL DATA SECTION gLobName ,INIT
    RECORD recordName
    ...
    ENDRECORD
ENDGLOBAL

```

```

● EXTERNAL FUNCTION
    functionName ,type
ENDEXTERNAL

```

- IMPORT** ● Import namespaces.
- ENUM** ● Declare an enumeration.
- MAIN - ENDMETHOD** ● Explicitly define a main routine.
- .INCLUDE** ● Include external source code.
- RECORD** ● Define a data record.
- GROUP** ● Define a group.
- STRUCTURE** ● Define how data is laid out.
- COMMON** ● Define a shared data record.
- LITERAL** ● Define a literal.
- GLOBAL** ● Define a global data section.
- EXTERNAL FUNCTION** ● Declare an external function.

## Basic Synergy Data Types

Data Type	Declaration	Literal	Parameter
Alpha	a2	"abc"	, a
Decimal	d5	"123"	, d
Implied-decimal	d8.2	"1.23"	, d.
Integer	i4	123	, i

## Declaring Variables

```

;variableName      ,type      ,initValue
alpha              ,a3        ,'abc'
emptyAlpha        ,a3
impliedDecimal    ,d5.2      ,123.45
simpleArray        ,2a5        ,"red", "blue"
complexArray      ,[4]d5     ,1 ,2 ,3 ,4
twoDimArray       ,[5,2]i4
dynamicArray      ,[#,#]struct1

```

*Access simple arrays with simpleArray(2).*

*Access complex arrays with complexArray[8].*

*Arrays in Synergy start at 1 by default.*

## Casting Data

To alpha	^a(expression)
To decimal	^d(expression)
To implied-decimal	^d(expression, precision)
To integer	^i(expression)
To string	%string(value ,format)

## Other Variable Statement Syntax

<b>INCR</b> var	Increment a numeric variable.
<b>DECR</b> var	Decrement a numeric variable.
<b>SET</b> var1 ,var2,... = 1	Assign a value to a set of variables.
<b>CLEAR</b> var1 ,var2,...	Set variable to default state.
<b>INIT</b> var1 ,var2,...	Set data structure to initial values.
<b>UPCASE</b> var	Convert characters to uppercase.



### Need more help?

Resource Center: [synergex.com/welcomeRC](http://synergex.com/welcomeRC)

Documentation: [synergex.com/docs](http://synergex.com/docs)

Email: [support@synergex.com](mailto:support@synergex.com)

Call: 800.366.3472

916.635.7300

## Procedure Division

```

● PROC
  IF(expression)
  ● BEGIN
    ● DATA var1, a5 , 'Hello'
    ● DATA var2 = 123
  ● END
  ● CALL internalSubroutine
  returnVar = %myFunc(arg1)
  ● XCALL mySubr(arg1, arg2)
  ● XCALL myFunc(rVar, arg1)
  ● STOP

● internalSubroutine,
  ...
● RETURN

ENDMAIN

● SUBROUTINE mySubr
  custNumber      ,n
  custName        ,a
  ● ENDPARAMS
  RECORD
    custType      ,a20

● PROC
  ...
  ● XRETURN
  ENDSUBROUTINE

● FUNCTION myFunc      ,a
  custNumber           ,n
  ● ENDPARAMS
  RECORD
    custName           ,a20

● PROC
  ...
  ● FRETURN custName
  ENDFUNCTION

;Comment with a semicolon

```

PROC-END ● Begin and end the procedure division.  
 BEGIN-END ● Begin and end a compound statement.  
 DATA ● Add a local stack variable declaration.  
 CALL ● Call an internal subroutine.  
 XCALL ● Call an external subroutine or function.  
 STOP ● Terminate program execution.  
 RETURN ● Return control from a subroutine.  
 SUBROUTINE ● Define an external subroutine.  
 ENDPARAMS ● End parameter list.  
 XRETURN ● Return control to a calling routine.  
 FUNCTION ● Define a function.  
 FRETURN ● Return a value to the calling routine.

## Control Flow

```

WHILE expression DO
  statement

DO statement
UNTIL expression

DO FOREVER
BEGIN
  statement
  IF expression
  EXITLOOP
END

REPEAT
BEGIN
  statement
  IF expression
  EXITLOOP
END

FOR count FROM initial THRU final BY incr
  statement

FOREACH variable IN collection AS type

IF expression
  statement

IF expression THEN
  statement
ELSE
  statement

CASE expression OF
BEGINCASE
  match_term1: statement
  match_term2: statement
  match_term3: statement
ENDCASE
ELSE statement

USING expression SELECT
  (match_term1), statement
  (match_term2), statement
  (match_term3), statement
ENDUSING

```

## Relational Operators

			String Relational
Equals to	==	.EQ.	.EQS.
Not equal to	!=	.NE.	.NES.
Greater than	>	.GT.	.GTS.
Less than	<	.LT.	.LTS.
Greater or equal to	>=	.GE.	.GES.
Less or equal to	<=	.LE.	.LES.

*String relational operators are used to compare alpha data or System.String data.*

## Boolean Operators

&&	.AND.
	.OR.
	.XOR.
!	.NOT.

IF var>0 && var<=10  
 XCALL mySubR

## Optional Parameter Definitions

REQUIRED	Required argument	IN	Read only
OPTIONAL	Optional argument	OUT	Write only
		INOUT	Read and write

```

XCALL exampleSubroutine(var1, var2, var3, var4)
...
SUBROUTINE exampleSubroutine
  REQUIRED IN      var1      ,d
  REQUIRED INOUT   var2      ,d.
  REQUIRED OUT     var3      ,n
  OPTIONAL OUT    var4      ,d
...

```