|  |  |  |  |
| --- | --- | --- | --- |
| File Name | Line (Final) | Why We Changed it | What we Changed it to |
| semantic.ssl | 18-90 | Updated the input semantic tokens to match parser.ssl | Input :  sIdentifier  firstInputToken = sIdentifier  sInteger  sLiteral  sProgram  sParmBegin  sParmEnd  sConst  sType  sVar  sProcedure  %%%%%%%%%%% ADDED  sPublic  %%%%%%%%%%%  sBegin  sEnd  sNegate  sArray  sPacked  sFile  sRange  sCaseStmt  %%%%%%%%%%% ADDED  sDefault  %%%%%%%%%%%  sCaseEnd  sLabelEnd  sExpnEnd  sNullStmt  sAssignmentStmt  sSubscript  sCallStmt  sFieldWidth  sIfStmt  sThen  sElse  %%%%%%%%%%% CHANGED  % sWhileStmt  % sRepeatStmt  % sRepeatEnd  %%%%%%%%%%%  sClass  sLoopStmt  sLoopBreakWhen  sLoopEnd  %%%%%%%%%%%  %%%%%%%%%%% ADDED 858  sReturn  %%%%%%%%%%%  sEq  sNE  sLT  sLE  sGT  sGE  sAdd  sSubtract  sMultiply  sDivide  sModulus  sInfixOr  sOr  sInfixAnd  sAnd  sNot  %%%%%%%%%%% ADDED  sSubstring  sLength  %%%%%%%%%%  sNewLine  sEndOfFile  lastInputToken = sEndOfFile; |
| semantic.ssl | 140-176 | Changed all the T-Codes for Char operations to be String operations and added T-Codes that are needed | tConcatenate  tSubstring  tLength  tStringEqual  %tWhileBegin  %tRepeatBegin  %tRepeatControl  tLoopBegin  tLoopBreakWhen  tLoopTest  tLoopEnd  %tStringDescriptor  %tSkipString  %tWhileTest  %tWhileEnd  %tRepeatTest |
| semantic.ssl | 239 | Added definition for stringSize to the type Integer and set value to 1024 | stringSize = 1024; |
| semantic.ssl | 286 | Updated StdType to contain stdString | stdString |
| semantic.ssl | 324-331 | Added oSymbolTblStripScope and oSymbolTblMergeScope | oSymbolTblStripScope  %Change all the identSymbolTableRefs for symbols in the  %top scope to their symbolTblLink values, (remove them from  %visibility even though they are still in table  oSymbolTblMergeScope;  %decrement the lexical level without changing any ident links |
| semantic.ssl | 346 | Add a kind for syClass and syPublicProcedure to the SymbolKind | syClass |
| semantic.ssl | 480-481 | Changed tpChar to be tpString | % tpChar  tpString |
| semantic.ssl | 596-603 | Changed the trReadString and trWriteString and removed trWriteString | % trWriteString = 7  trWriteInteger = 8  trReadInteger = 10  trReadString = 107  trWriteString = 108 %9  %trReadString  trAssign = 12; |
| semantic.ssl | 1447-1463 | Changed ClassDefinition to handle classes | ClassDefinitions :  sIdentifier  oSymbolStkPushLocalIdentifier  oSymbolStkSetKind(syClass)  oTypeStkPush(tpNull)  TypeTblEnter  oSymbolStkEnterTypeReference  oSymbolTblEnter  oSymbolTblPushScope  @Block  oSymbolTblUpdate  oSymbolStkPop  oTypeTblUpdate  oTypeStkPop  oSymbolTblStripScope  oSymbolTblMergeScope; |
| semantic.ssl | 1466-1503 | Modified the handling of constant definitions to allow only one per definition | ProcedureDefinition :  sIdentifier oSymbolStkPushLocalIdentifier % procedure name  [ oSymbolStkChooseKind  | syUndefined:  | syExternal:  % Program parameters must be declared as file variables  #eExternalDeclare  | \*:  #eMultiplyDefined  % New definition obscures old one.  ]  [  | sPublic:  oSymbolStkSetKind(syPublicProcedure)  | \*:  oSymbolStkSetKind(syProcedure)  ]  .tSkipProc % Branch around the procedure definition  oFixPushForwardBranch  oEmitNullAddress  oSymbolStkEnterCodeAddress  oTypeStkPush(tpNull)  oTypeStkSetRecursionFlag(yes)  oTypeTblEnter  oSymbolStkEnterTypeReference  oSymbolTblEnter  oSymbolTblPushScope % procedure scope  @ProcedureHeading  @Block  .tProcedureEnd  oFixPopForwardBranch % fix skip around procedure  oSymbolTblUpdate  oSymbolStkPop  oTypeStkSetRecursionFlag(no)  oTypeTblUpdate  oTypeStkPop  oSymbolTblPopScope; |
| semantic.ssl | 992-1045 | Modified the handling of typed definitions to allow only one per definition | TypeDefinitions : % process named type definitions  %%%%%%%%%%%%% ALL I DID WAS REMOVE THE "{","}" TO MAKE IT ONLY ALLOW ONE PER DEFINITION%%%%%%%%%%%%%%%%%%%%  [  | sIdentifier:  oSymbolStkPushLocalIdentifier  @TypeBody  [ oSymbolStkChooseKind  | syUndefined:  | syExternal:  % Program parameters must be declared as file variables  #eExternalDeclare  % Ignore this definition  oSymbolStkPop  oTypeStkPop  oTypeStkPop  >>  | \*:  #eMultiplyDefined  % The new definition will obscure the old one  ]  oSymbolStkSetKind(syType)  % Create type table entries for the object as necessary.  % Where named types were used in the declaration, a type  % reference will be present, otherwise the type reference  % fields will be null.  oTypeStkSwap % structure on top  [ oTypeStkChooseKind  | tpArray:  oTypeStkSwap % component on top  [ oTypeStkChooseTypeReference  | no:  % No type table entry exists,  % so create one.  oTypeTblEnter  | \*:  ]  oTypeStkEnterComponentReference  oTypeStkSwap % structure on top  | \*:  ]  [ oTypeStkChooseTypeReference  | no:  oTypeTblEnter  | \*:  ]  oSymbolStkEnterTypeReference  oSymbolTblEnter  oSymbolStkPop  oTypeStkPop  oTypeStkPop  | \*:  % >  ]; |
| semantic.ssl |  | Changed handling of procedure definitions to recognize public procedures and store them with syPublicProcedure | ProcedureDefinition :  sIdentifier oSymbolStkPushLocalIdentifier % procedure name  [ oSymbolStkChooseKind  | syUndefined:  | syExternal:  % Program parameters must be declared as file variables  #eExternalDeclare  | \*:  #eMultiplyDefined  % New definition obscures old one.  ]  [  | sPublic:  oSymbolStkSetKind(syPublicProcedure)  | \*:  oSymbolStkSetKind(syProcedure)  ]  .tSkipProc % Branch around the procedure definition  oFixPushForwardBranch  oEmitNullAddress  oSymbolStkEnterCodeAddress  oTypeStkPush(tpNull)  oTypeStkSetRecursionFlag(yes)  oTypeTblEnter  oSymbolStkEnterTypeReference  oSymbolTblEnter  oSymbolTblPushScope % procedure scope  @ProcedureHeading  @Block  .tProcedureEnd  oFixPopForwardBranch % fix skip around procedure  oSymbolTblUpdate  oSymbolStkPop  oTypeStkSetRecursionFlag(no)  oTypeTblUpdate  oTypeStkPop  oSymbolTblPopScope; |
| semantic.ssl | 1854-1898 | Remove the handling of loop statements and add handling the switch statement | %WhileStmt :  % .tWhileBegin  % oFixPushTargetAddress % top-of-loop branch target  % @BooleanControlExpression  % .tWhileTest  % oFixPushForwardBranch  % oEmitNullAddress % exit branch  % oFixSwap % top-of-loop target back on top  % @Statement  % .tWhileEnd  % oFixPopTargetAddress  % oFixPopForwardBranch;  LoopStmt :  sBegin  .tLoopBegin  oFixPushTargetAddress  @BooleanControlExpression  @Statement  sLoopBreakWhen %%%%%%%%%%%%  .tLoopTest  oFixPushForwardBranch  oEmitNullAddress  oFixSwap  @Statement  .tLoopEnd  oFixPopTargetAddress  oFixPopForwardBranch;  %RepeatStmt :  % .tRepeatBegin  % oFixPushTargetAddress % top-of-loop branch target  % {[  % | sRepeatEnd:  % >  % | \*:  % @Statement  % ]}  % .tRepeatControl  % @BooleanControlExpression  % .tRepeatTest  % oFixPopTargetAddress; |
| semantic.ssl | 2186-2195 | Added handling of ternary operator substring | TernaryOperator:  [  | sSubstring:  .tSubstring  oTypeStkPush(tpString)  @CompareTernaryOperandAndResultTypes  | \*:  ]; |
| Semantic.ssl | 1995-2013 | Handle calling the ternary operator | Expression :  % Expressions have been converted to postfix form by the  % previous pass with the exceptions noted below. This rule  % pushes symbol and type table entries for the expression result.  {  @Operand  @UnaryOperator  @BinaryOperator  %%%ADDED BY BEN AND STORUS  @TernaryOperator  [  | sExpnEnd:  >  | sParmBegin, sSubscript:  #eInvalidExpn  @FlushExpn  | \*:  ]  }; |
| Semantic.ssl | 2095-2116 | Changed UnaryOperator to handle the length operation | UnaryOperator :  [  | sNegate:  .tNegate  oTypeStkPush(tpInteger) % result type  @CompareAndSwapTypes  | sNot:  .tNot  oTypeStkPush(tpBoolean)  @CompareAndSwapTypes  |sLength:  .tLength  oTypeStkPush(tpInteger)  @StringLengthCompareTypes  | \*:  >> % bypass type check  ]  % If an operator is present the result is an expression  oSymbolStkSetKind(syExpression)  @CompareAndSwapTypes  oTypeStkPop; |
| semantic.ssl | 2137-2123 | Added handling of concatenation to the sAdd part of the Binary Operator rule | | sAdd:  [oTypeStkChooseKind  | tpString :  .tConcatenate  oTypeStkPush(tpString)  | tpPackedArray :  .tConcatenate  oTypeStkPush(tpPackedArray)  |\*:  .tAdd  oTypeStkPush(tpInteger)  ]  %%%TypeStkPush(tpInteger) % result type  @CompareOperandAndResultTypes |
| semantic.ssl | 956-985 | Removed the tSkipString and tStringDescriptor and Changed handling of string constants to act like vars | | sLiteral:  oValuePush(one) % implicit lower bound of string index type  oValuePushStringLength  [ oValueChoose  | zero:  #eNullString  @SymbolStkPushCharConstant  | one:  oSymbolStkLinkToStandardType(stdString)  oSymbolStkEnterStringValue  | \*:  % Emit a branch around the string since  % it is stored in the code area  %.tSkipString  %oFixPushForwardBranch  %oEmitNullAddress  .tAssignBegin  .tLiteralString  oEmitValue % string length  % The string "address" is the location (offset)  % of the first character of the string  oSymbolStkEnterCodeAddress  oEmitString  .tAssignString  %oFixPopForwardBranch  oTypeStkPush(tpPackedArray) % means literal string  oTypeStkEnterBounds  oTypeStkLinkToStandardComponentType(stdString)  oTypeTblEnter  oSymbolStkEnterTypeReference  oTypeStkPop  ] |
| semantic.pt | 60-511 | Changed the semantic operations to those generated in semantic.def | { ===== Pasted contents of semantic.def, generated by S/SL from semantic.ssl -  defines Semantic Operations, Input/Output Tokens, Error Codes and Semantic Types }  { Semantic Operations }  oSymbolTblEnter = 14;  oSymbolTblUpdate = 15;  oSymbolTblPushScope = 16;  oSymbolTblPopScope = 17;  oSymbolTblLookupExternal = 18; { >>14 }  oSymbolTblStripScope = 19;  oSymbolTblMergeScope = 20;  oSymbolStkPush = 21; { (17) }  oSymbolStkPushIdentifier = 22;  oSymbolStkPushLocalIdentifier = 23;  oSymbolStkPushStandardVariable = 24; { (19) }  oSymbolStkPushFormalParameter = 25;  oSymbolStkSetKind = 26; { (17) }  oSymbolStkLinkToStandardType = 27; { (16) }  oSymbolStkEnterValue = 28;  oSymbolStkEnterStringValue = 29;  oSymbolStkEnterCodeAddress = 30;  oSymbolStkEnterDataAddress = 31;  oSymbolStkEnterTypeReference = 32;  oSymbolStkChooseKind = 33; { >>17 }  oSymbolStkChooseStandardFile = 34; { >>19 }  oSymbolStkChooseStandardRoutine = 35; { >>18 }  oSymbolStkPop = 36;  oSymbolStkPrint = 37;  oTypeTblEnter = 38;  oTypeTblUpdate = 39;  oTypeStkPush = 40; { (20) }  oTypeStkPushSymbol = 41;  oTypeStkPushComponent = 42;  oTypeStkSetKind = 43; { (20) }  oTypeStkSetRecursionFlag = 44; { (14) }  oTypeStkChooseRecursionFlag = 45; { >>14 }  oTypeStkSetPackedComponentFlag = 46; { (14) }  oTypeStkChoosePackedComponentFlag = 47; { >>14 }  oTypeStkLinkToStandardComponentType = 48; { (16) }  oTypeStkEnterBounds = 49;  oTypeStkVerifyBounds = 50; { >>14 }  oTypeStkEnterParameterCount = 51;  oTypeStkCompareParameterCount = 52; { >>14 }  oTypeStkEnterComponentReference = 53;  oTypeStkChooseKind = 54; { >>20 }  oTypeStkChooseTypeReference = 55; { >>14 }  oTypeStkCompareNames = 56; { >>14 }  oTypeStkSwap = 57;  oTypeStkPop = 58;  oTypeStkDuplicate = 59;  oTypeStkPrint = 60;  oEmitNullAddress = 61;  oEmitValue = 62;  oEmitString = 63;  oEmitDataAddress = 64;  oEmitTrapKind = 65; { (21) }  oEmitCaseBranchTable = 66;  oAllocateAlignOnWord = 67;  oAllocateVariable = 68;  oAllocateVarParameter = 69;  oAllocateDescriptor = 70;  oValuePush = 71; { (13) }  oValuePushInteger = 72;  oValuePushStringLength = 73;  oValuePushSymbol = 74;  oValuePushCount = 75;  oValuePushTypeStkLowerBound = 76;  oValuePushTypeStkUpperBound = 77;  oValueNegate = 78;  oValueChoose = 79; { >>13 }  oValuePop = 80;  oCasePushDisplay = 81;  oCasePopDisplay = 82;  oCaseLookupLabel = 83; { >>14 }  oCasePushLabel = 84;  oCountPush = 85; { (13) }  oCountPushValue = 86;  oCountPop = 87;  oCountIncrement = 88;  oCountDecrement = 89;  oCountChoose = 90; { >>13 }  oFixPushForwardBranch = 91;  oFixPopForwardBranch = 92;  oFixPushTargetAddress = 93;  oFixPopTargetAddress = 94;  oFixPopAndEnterValue = 95;  oFixSwap = 96;  { Input Tokens }  sIdentifier = 0;  firstInputToken = 0;  sInteger = 1;  sLiteral = 2;  sProgram = 3;  sParmBegin = 4;  sParmEnd = 5;  sConst = 6;  sType = 7;  sVar = 8;  sProcedure = 9;  sPublic = 10;  sBegin = 11;  sEnd = 12;  sNegate = 13;  sArray = 14;  sPacked = 15;  sFile = 16;  sRange = 17;  sCaseStmt = 18;  sDefault = 19;  sCaseEnd = 20;  sLabelEnd = 21;  sExpnEnd = 22;  sNullStmt = 23;  sAssignmentStmt = 24;  sSubscript = 25;  sCallStmt = 26;  sFieldWidth = 27;  sIfStmt = 28;  sThen = 29;  sElse = 30;  sClass = 31;  sLoopStmt = 32;  sLoopBreakWhen = 33;  sLoopEnd = 34;  sReturn = 35;  sEq = 36;  sNE = 37;  sLT = 38;  sLE = 39;  sGT = 40;  sGE = 41;  sAdd = 42;  sSubtract = 43;  sMultiply = 44;  sDivide = 45;  sModulus = 46;  sInfixOr = 47;  sOr = 48;  sInfixAnd = 49;  sAnd = 50;  sNot = 51;  sSubstring = 52;  sLength = 53;  sNewLine = 54;  sEndOfFile = 55;  lastInputToken = 55;  { Output Tokens }  tMultiply = 0;  firstOutputToken = 0;  tDivide = 1;  tModulus = 2;  tAdd = 3;  tSubtract = 4;  tEQ = 5;  tNE = 6;  tGT = 7;  tGE = 8;  tLT = 9;  tLE = 10;  tAnd = 11;  tInfixAnd = 12;  tOr = 13;  tInfixOr = 14;  tNegate = 15;  tNot = 16;  tChr = 17;  tOrd = 18;  tEoln = 19;  tEOF = 20;  tVarParm = 21;  tFetchAddress = 22;  tFetchInteger = 23;  tFetchString = 24;  tFetchBoolean = 25;  tAssignBegin = 26;  tAssignAddress = 27;  tAssignInteger = 28;  tAssignString = 29;  tAssignBoolean = 30;  tStoreParmAddress = 31;  tStoreParmInteger = 32;  tStoreParmString = 33;  tStoreParmBoolean = 34;  tSubscriptBegin = 35;  tSubscriptAddress = 36;  tSubscriptInteger = 37;  tSubscriptString = 38;  tSubscriptBoolean = 39;  tArrayDescriptor = 40;  tFileDescriptor = 41;  tIfBegin = 42;  tIfEnd = 43;  tConcatenate = 44;  tSubstring = 45;  tLength = 46;  tStringEqual = 47;  tCaseBegin = 48;  tCaseElse = 49;  tLoopBegin = 50;  tLoopBreakWhen = 51;  tLoopTest = 52;  tLoopEnd = 53;  tCallBegin = 54;  tParmEnd = 55;  tProcedureEnd = 56;  tWriteBegin = 57;  tReadBegin = 58;  tTrapBegin = 59;  tWriteEnd = 60;  tReadEnd = 61;  tLiteralAddress = 62;  firstCompoundOutputToken = 62;  tLiteralInteger = 63;  tLiteralBoolean = 64;  tLiteralString = 65;  tIfThen = 66;  tIfMerge = 67;  tCaseSelect = 68;  tCaseMerge = 69;  tCaseEnd = 70;  tSkipProc = 71;  tCallEnd = 72;  tLineNumber = 73;  tTrap = 74;  lastCompoundOutputToken = 74;  tEndOfFile = 75;  lastOutputToken = 75;    { Input/Output Tokens }  { Error Codes }  eDuplicateName = 10;  firstErrorCode = 10;  eMultiplyDefined = 11;  eUndefinedIdentifier = 12;  eConstantReqd = 13;  eNullString = 14;  eSubrangeReqd = 15;  eScalarReqd = 16;  eIntegerConstReqd = 17;  eBounds = 18;  eSimpleTypeReqd = 19;  eExternalUndeclared = 20;  eExternalDeclare = 21;  eFileVarReqd = 22;  eFileNameReqd = 23;  eOperandOperatorTypeMismatch = 24;  eNonScalarValParm = 25;  eParameterTypeMismatch = 26;  ePackedComponentByVar = 27;  eInsufficientActuals = 28;  eExtraActuals = 29;  eRecursiveCall = 30;  eProcedureReqd = 31;  eTypeIdentifierReqd = 32;  eIntegerExpnReqd = 33;  eBooleanExpnReqd = 34;  eCharExpnReqd = 35;  eArrayVarReqd = 36;  eVariableReqd = 37;  eDuplicateLabel = 38;  eExpnOperandReqd = 39;  eTypeMismatch = 40;  eInvalidExpn = 41;  eInputNotVisible = 42;  eOutputNotVisible = 43;  eFieldWidth = 44;  eCharFileVarReqd = 45;  eWriteExpn = 46;  eReadVar = 47;  lastSslErrorCode = 47;  { Type Values }  { Type 13 }  zero = 0;  one = 1;  two = 2;  three = 3;  ten = 10;  byteSize = 1;  wordSize = 4;  stringSize = 1024;  { Type 14 }  no = 0;  yes = 1;  { Type 15 }  firstPredeclaredId = 1;  firstPredeclaredType = 1;  pidInteger = 1;  pidString = 2;  pidBoolean = 3;  pidText = 4;  lastPredeclaredType = 4;  pidTrue = 5;  pidFalse = 6;  pidReset = 7;  pidRewrite = 8;  pidWrite = 9;  pidWriteln = 10;  pidRead = 11;  pidReadln = 12;  pidChr = 13;  pidOrd = 14;  pidEoln = 15;  pidEof = 16;  pidAssign = 17;  lastPredeclaredId = 17;  pidInput = 18;  firstSpecialId = 18;  pidOutput = 19;  lastSpecialId = 19;  { Type 16 }  stdInteger = 0;  stdString = 1;  stdBoolean = 2;  stdText = 3;  { Type 17 }  syConstant = 0;  syType = 1;  syVariable = 2;  syVarParameter = 3;  syProcedure = 4;  syFunction = 5;  syExternal = 6;  syExpression = 7;  syUndefined = 8;  syClass = 9;  syPublicProcedure = 10;  { Type 18 }  rtReset = 1;  rtRewrite = 2;  rtWrite = 3;  rtWriteln = 4;  rtRead = 5;  rtReadln = 6;  rtOrd = 7;  rtChr = 8;  rtEoln = 9;  rtEof = 10;  rtAssign = 11;  rtNull = 12;  { Type 19 }  stdInput = 0;  stdOutput = 1;  stdNull = 2;  { Type 20 }  tpInteger = 0;  tpString = 1;  tpBoolean = 2;  tpSubrange = 3;  tpArray = 4;  tpPackedArray = 5;  tpFile = 6;  tpNull = 7;  { Type 21 }  trHalt = 0;  trReset = 1;  trRewrite = 2;  trRead = 3;  trReadln = 4;  trWrite = 5;  trWriteln = 6;  trWriteInteger = 8;  trWriteString = 108;  trReadInteger = 10;  trReadString = 107;  trAssign = 12;  { S/SL Rule Table Addresses } {  Program = 0;  ProgramParameter = 43;  Block = 103;  AllocateVar = 906;  ProcedureDefinition = 1001;  ConstantDefinitions = 139;  TypeDefinitions = 302;  VariableDeclarations = 715;  BeginStmt = 1388;  ConstantValue = 149;  SymbolStkPushIntegerConstant = 4292;  SymbolStkPushCharConstant = 4304;  TypeBody = 371;  IndexType = 448;  ComponentType = 494;  SimpleType = 543;  ValuePushValuePlusOne = 4283;  SymbolStkPushIntegerType = 4316;  SubrangeUpperBound = 653;  VariableAttributes = 736;  ClassDefinition = 984;  ProcedureHeading = 1092;  ProcedureParameterType = 1138;  ProcedurePrologue = 1209;  CountCopy = 4288;  Statement = 1278;  AssignmentStmt = 1326;  CallStmt = 1402;  IfStmt = 1665;  LoopStmt = 1696;  CaseStmt = 1719;  Variable = 2488;  Expression = 1883;  CompareAndSwapTypes = 2269;  OpenProcedure = 2896;  AssignProcedure = 2965;  WriteProcedure = 3129;  WritelnProcedure = 3667;  ReadProcedure = 3779;  ReadlnProcedure = 4141;  ActualParameters = 1535;  FlushActuals = 4356;  VariableActual = 1603;  BooleanControlExpression = 4268;  CaseSelectorExpression = 4251;  CaseAlternative = 1779;  Operand = 1913;  UnaryOperator = 2047;  BinaryOperator = 2091;  TernaryOperator = 2254;  FlushExpn = 4335;  ConstantOperand = 2622;  VariableOperand = 2646;  FunctionOperand = 2673;  SymbolStkPushIntegerVariable = 4323;  FileOperand = 2871;  StringLengthCompareTypes = 2033;  CompareOperandAndResultTypes = 2427;  CompareRelationalOperandTypes = 2438;  CompareTernaryOperandAndResultTypes = 2417;  CompareAndSwapTypesTernaryWithInt = 2373;  CompareAndSwapTypesTernary = 2394;  VariableExtension = 2530;  StandardFunctionActual = 2776;  WriteChar = 3459;  WriteNonChar = 3583;  WriteStandardExpnList = 3392;  ReadCharParameterList = 3909;  ReadNonCharParameterList = 3975;  ReadChar = 4041;  ReadNonChar = 4070;  }  { S/SL Table Parameters }  sslTblSize = 4375;  minSslTableValue = -32767;  maxSslTableValue = 32767;  { ===== End of contents of semantic.def } |
| semantic.pt | EVERY WHERE | Changed char to string everywhere except for where char was being used by the compiler |  |