File created: 21-Apr-86 13:43:26 **{POGO:PARC:XEROX}<LOOPS>TESTER>LTKER.;2**

changes to: (CLASSES LOOPSTestLispFunc)

previous date: 25-Mar-85 16:14:18 {POGO:PARC:XEROX}<LOOPS>TESTER>LTKER.;1

Read Table: OLD-INTERLISP-FILE

Package: INTERLISP

Format: XCCS


```
(RPAQQ LTKERCOMS
       ((E (ResetLTKERVARS))
        (CLASSES LOOPSCoreTest LOOPSTestBasic LOOPSTestBraidObject LOOPSTestEnvironment LOOPSTestKernel
               LOOPSTestLispFunc LOOPSTestMethod LOOPSTestObject LOOPSTestPrimitive LOOPSTestSyntax)
        (METHODS LOOPSTestEnvironment.TEST LOOPSTestEnvironment.TestSelf LOOPSTestKernel.TEST
               LOOPSTestObject.DefineTEST LOOPSTestObject.EditTEST LOOPSTestObject.EditTestInTTYProcess
               LOOPSTestObject.IgnoreTest LOOPSTestObject.ReTest LOOPSTestObject.ReTestDep
               LOOPSTestObject.Reset! LOOPSTestObject.ResetAll LOOPSTestObject.ResetDep LOOPSTestObject.TEST
               LOOPSTestObject.TEST! LOOPSTestObject.TESTDep LOOPSTestObject.TESTall LOOPSTestObject.TestSelf
               LOOPSTestPrimitive.XTEST)
        (FNS AddAltTest AllowRemove AskPreTest AskSubTest AskSyntaxTest AskTestCases BuildPreTest
             BuildPrimClassTest CheckClassTest CheckPreTest CreateLTKBS1 DescribePreviousTry DescribeTestLink
             DisplayTestBrowser DoLoopsTest EQACTVAL EditOtherLinksMenu EditPreTest EditTestOtherCmds
             ExecTestFields FindObjForLink GIVGetFn GetFromActVal LinkEditOtherMenu MakeBackLink MakeSet
             ObjectName PreTestsSatisfied? PushClassValueNew PutInActVal ReadLinkMenu RemoveValue
             ResetLTKERCLASSES ResetLTKERVARS ResetPutLocalStateVars SetupEditTestObjMenu TickleBrowserNodes)
        (INSTANCES * LTKERINSTANCES)
        (VARS * LTKERVARS)
        (P (ResetLTKERCLASSES))))

(DEFCLASSES LOOPSCoreTest LOOPSTestBasic LOOPSTestBraidObject LOOPSTestEnvironment LOOPSTestKernel
       LOOPSTestLispFunc LOOPSTestMethod LOOPSTestObject LOOPSTestPrimitive LOOPSTestSyntax)

(DEFCLASS LOOPSCoreTest (MetaClass LOOPSTestMeta Edited:          (* sm: "18-Mar-85 16:30"))
       (Supers LOOPSTestObject))

(DEFCLASS LOOPSTestBasic (MetaClass LOOPSTestMeta Edited:         (* sm: "18-Mar-85 16:31")
                                 doc                              (* class for Test objects which are by themselves and not share
                                 PreTest dependencies)
                                 )
       (Supers LOOPSCoreTest)
       (ClassVariables (InstanceComsVar LTCORETESTSINSTANCES)
              (InstancePrefix LT)))

(DEFCLASS LOOPSTestBraidObject (MetaClass LOOPSTestMeta Edited:   (* sm: "18-Mar-85 16:31")
                                      doc                         (* instances are objects for testing LOOPS's built-in classes)
                                      )
       (Supers LOOPSCoreTest)
       (ClassVariables (InstanceComsVar LTCORETESTSINSTANCES)
              (InstancePrefix LTB)))

(DEFCLASS LOOPSTestEnvironment (MetaClass LOOPSTestMeta Edited:   (* sm: "18-Mar-85 16:32")
                                      doc                         (* used for testing KB and environments)
                                      )
       (Supers LOOPSTestObject)
       (ClassVariables (InstancePrefix LTE)
              (InstanceComsVar LTKBTESTINSTANCES)
              (ClassPreTest #. ($A (LTMethod)
                                   NIL AllowRemove)
                     Failed ?)
              (ClassTested? U)
              (UnnamedInstanceCount 0))
       (InstanceVariables (AfterTest NIL)))

(DEFCLASS LOOPSTestKernel (MetaClass LOOPSTestMeta Edited:        (* sm: "18-Mar-85 16:32")
                                 doc                              (* instances are for testing Kernel features.
                                                                  currently only LTKernel is needed)
                                 )
       (Supers LOOPSCoreTest)
       (ClassVariables (InstanceComsVar LTCORETESTSINSTANCES)))

[DEFCLASS LOOPSTestLispFunc (MetaClass LOOPSTestMeta Edited:      (* sm: "18-Mar-85 16:32")
                                   doc                            (* instances are Test objects for LOOPS's Lisp functions)
                                   )
       (Supers LOOPSCoreTest)
       (ClassVariables (InstanceComsVar LTLispFunTestsInstances)
              (InstancePrefix LTF))
       (InstanceVariables (LispName #.NotSetValue doc             (* Lisp function name corresp to this object)]

(DEFCLASS LOOPSTestMethod (MetaClass LOOPSTestMeta Edited:        (* sm: "18-Mar-85 16:32")
                                 doc                              (* instances are for testing LOOPS's built-in methods)
                                 )
       (Supers LOOPSCoreTest)
```

```
       (ClassVariables (ClassPreTest #. ($A (LTMethod)
                                            NIL AllowRemove))
              (ClassTested? U)
              (InstanceComsVar LTMethodInstances)
              (InstancePrefix LTM)
              (UnnamedInstanceCount 0)))

[DEFCLASS LOOPSTestObject (MetaClass LOOPSTestMeta Edited:        (* sm: "18-Mar-85 16:34")
                                 doc                              (* all testobjects which contain code for some test have this on
                                                                  their supers chain)
                                 )
       (Supers LOOPSTestAbstract)
       (ClassVariables (ClassPreTest #. ($A NIL NIL AllowRemove)
                               Failed NIL doc

          (* list of objects that need to be tested for a class of test objects and are tried before an objects own PreTest)
)
              (InstancePrefix LTO)
              (UnnamedInstanceCount 0)
              (InstanceComsVar LTKERINSTANCES)
              (ClassTested? U doc                                 (* set to T/NIL depending on ClassPreTest results))
              (ResetList ((IV Tested? CompletelyTested? (PreTest Tested?)
                              (Tested? DoneOnce)
                              (Tested? Ignored)
                              (SetUp Tested?)
                              (SyntaxTest Tested?)
                              (SubTest Tested?)
                              (AltTest Tested?)
                              (SetUp FailedExp)
                              (TestExpr FailedExpr)
                              (TestExpr HowFailed)
                              (PreTest Failed)
                              (SyntaxTest Failed)
                              (SubTest Failed)
                              (AltTest Failed))
                          (CV ClassTested? (ClassPreTest Failed)))
                     doc

          (* list of iv/cvs or their props which must be set to ? to begin a new test sequence)

                     ))
       (InstanceVariables (CasesUsed #. ($A NIL NIL MakeBackLink)
                                 doc                              (* list of TestCases objects used)
                                 BackLink UsedBy)
              (PreTest #. ($A NIL NIL MakeBackLink)
                     BackLink PreTestOf doc                       (* list of objects that need to be tested before this one)
                     Tested? U Failed NIL)
              (PreTestOf #. ($A NIL NIL MakeBackLink)
                     BackLink PreTest DontSave (Value)
                     doc                                          (* list of TestObjects which use this as PreTest)
                     )
              (SyntaxTest #. ($A NIL NIL MakeBackLink)
                     BackLink SyntaxTestOf doc

          (* list of objects of type LOOPSTestSyntax for testing any special syntactic form associated with this object)

                     Tested? U Failed NIL)
              (AltTestOf #. ($A NIL NIL MakeBackLink)
                     BackLink AltTest doc                         (* list of TestObj which use this for alternate ways of testing their
                     concept))
              (UsesObj #. ($A NIL NIL AllowRemove)
                     doc

          (* list of objects used by this; for ease of editing only -
          no back links are kept)

                     )
              (CompletelyTested? U Result NIL doc                 (* result of all tests))
              (StandBy #. ($A NIL NIL AllowRemove)
                     doc                                          (* list of objects which were once used but may not be in use
                                                                  anymore)
                     )
              (SubTest #. ($A NIL NIL MakeBackLink)
                     BackLink SubTestOf Tested? U Failed NIL doc
                                                                  (* further things to Test if this tests OK))
              (AltTest #. ($A NIL NIL MakeBackLink)
                     BackLink AltTestOf Tested? U Failed NIL doc
                                                                  (* list of TestObj which specify alternate ways of testing this
                                                                  concept)]

[DEFCLASS LOOPSTestPrimitive (MetaClass LOOPSTestMeta Edited:     (* sm: "18-Mar-85 16:37")
                                    doc

          (* instances are for testing some primitive aspect of LOOPS not covered by other classes and pointed to USUALLY by
          SubTest IV in other Test objects)

                                    )
```

```
       (Supers LOOPSTestObject)
       (ClassVariables (InstancePrefix LTP)
              (UnnamedInstanceCount 4)
              (ClassPreTest #. ($A NIL BuildPrimClassTest AllowRemove)
                     comm
```

(* this active values always adds the IV SubTestOf to this list, so that the supers are always considered as PreTests for a Primitive TestObj)

```
                     Failed ?))
       (InstanceVariables (PreTest #. ($A NIL BuildPreTest MakeBackLink)
                                  BackLink PreTestOf)
              (SubTestOf #. ($A NIL NIL MakeBackLink)
                     BackLink SubTest doc                          (* back link to TestObj which uses this for a subtest)]

[DEFCLASS LOOPSTestSyntax (MetaClass LOOPSTestMeta Edited:         (* sm: "18-Mar-85 16:34")
                                  doc
```

(* instances are for testing syntantic short forms in LOOPS. these instances are pointed to from SyntaxTest IV in particular test objects)

```
                                  )
       (Supers LOOPSTestObject)
       (ClassVariables (InstancePrefix LTS)
              (UnnamedInstanceCount 0))
       (InstanceVariables (SyntaxTestOf #. ($A NIL NIL MakeBackLink)
                                  BackLink SyntaxTest doc          (* list of TestObjs which use this for syntax test)]

(METH LOOPSTestEnvironment TEST (TestedLst)                        (* does not test if KBTestsFlg is NIL.)
      )

(METH LOOPSTestEnvironment TestSelf (TestedLst)                    (* does not test if KBTestsFlg is NIL)
      )

(METH LOOPSTestKernel TEST NIL                                     (* TESTS THE KERNEL FEATURES OF LOOPS))

(METH LOOPSTestObject DefineTEST NIL                               (* used to define Test description for a TestObj.))

(METH LOOPSTestObject EditTEST NIL                                 (* edits a TestObj using menus))

(METH LOOPSTestObject EditTestInTTYProcess NIL                     (* calls EditTEST in TTY Process))

(METH LOOPSTestObject IgnoreTest (TestedLst)
      NIL)

(METH LOOPSTestObject ReTest (TestedLst)                           (* sends a ResetSelf if needed followed by TEST)
      )

(METH LOOPSTestObject ReTestDep NIL                                (* first sends ResetDep and then TESTDep to self))

(METH LOOPSTestObject Reset! (ResetLst)
```

(* resets itself by sending Reset to self and also Reset! its SyntaxTest, AltTest and SubTest.
Also does a Reset on PreTest and ClassPreTest)

```
      )

(METH LOOPSTestObject ResetAll (ResetLst)
```

(* completely resets itself, all tests on which it depends, and all which depend on it)

```
      )

(METH LOOPSTestObject ResetDep (ResetLst)                          (* resets itself and all tests which depend on it)
      )

(METH LOOPSTestObject TEST (TestedLst)                             (* performs the basic TEST for a TestObject)
      )

(METH LOOPSTestObject TEST! (ContFl)                               (* first TESTs itself, and if successful, then does other tests-
                                                                   SubTest, SyntaxTest, AltTest.)
      )

(METH LOOPSTestObject TESTDep NIL
```

(* generates TEST call to self, and TESTDep to PreTestOf list, and SubTest list only if it succeeds)

```
)

(METH LOOPSTestObject TESTall NIL                                  (* generates TEST call to self, PreTestOf list, and SubTest list))

(METH LOOPSTestObject TestSelf (TestedLst)                         (* performs the basic TEST for a TestObject but continues even
      if PreTests fail)
      )

(METH LOOPSTestPrimitive XTEST NIL
```

(* performs the basic TEST for a Primitive TestObject ONLYIF its super test Tested T and sets own Tested? IV)

)

(DEFINEQ

(LOOPSTestEnvironment.TEST

```
  (Method ((LOOPSTestEnvironment TEST)
           self TestedLst)                                          (* sm: " 5-Jun-84 12:59")
                                                                    (* does not test if KBTestsFlg is NIL.)
                        (COND
                           (KBTestsFlg (_Super
                                       self TEST TestedLst))
                           (T (_ self IgnoreTest TestedLst)))))
```

(LOOPSTestEnvironment.TestSelf

```
  (Method ((LOOPSTestEnvironment TestSelf)
           self TestedLst)                                          (* sm: " 5-Jun-84 11:02")
                                                                    (* does not test if KBTestsFlg is NIL)
                        (COND
                           (KBTestsFlg (_Super
                                       self TestSelf TestedLst))
                           (T (_ self IgnoreTest TestedLst)))))
```

(LOOPSTestKernel.TEST

```
  (Method ((LOOPSTestKernel TEST)
           self)                                                    (* sm: "20-SEP-83 16:31")
                                                                    (* TESTS THE KERNEL FEATURES OF LOOPS)
                 (PROG (LTKInst LTKClass Temp)                      (* TEST: Create a new class)
                                                                    (* Also test that Class is set properly)
                       (COND
                          ((NOT (EQ (@ Tested?)
                                    NotSetValue))
                           (printout TTY "Kernel Test was " (COND
                                                              ((NULL (@ Tested?))
                                                               "Unsuccessful")
                                                              (T "Successful"))
                                   T)
                           (RETURN (@ Tested?)))
                          (T (printout TTY "Begin test of Kernel features of LOOPS..." T)))
                       (DC (QUOTE LTClass)
                           (QUOTE (Object)))
                       (COND
                          ((EQ (SETQ LTKClass (GetClass ($ LTClass)))
                               ($ Class)))
                          (T (printout TTY "Bug: Class of a newly created class not set properly" LTKClass T)))
                                                                    (* TEST: Create an instance of this)
                                                                    (* Also test that Class is set properly)
                                                                    (* Also tests message passing and Method Inheritance)
                       (SETQ LTKInst (_ ($ LTClass)
                                        New))
                       (COND
                          ((EQ (SETQ LTKClass (Class LTKInst))
                               ($ LTClass)))
                          (T (printout TTY "Bug: Class of an instance of LTClass not set properly" LTKClass T)))
                                                                    (* TEST: Add CV and IVs to the class and Get from Instance)
                                                                    (* Tests inheritance of IV and CVs)
                       (_ ($ LTClass)
                          Add
                          (QUOTE CV)
                          (QUOTE CVTest1)
                          (QUOTE CVal1))
                       (_ ($ LTClass)
                          Add
                          (QUOTE IV)
                          (QUOTE IVTest1)
                          (QUOTE IVal1))
                       (_ ($ LTClass)
                          Add
                          (QUOTE IV)
                          (QUOTE IVTest2)
                          (QUOTE IVal2))
                       (COND
                          [(EQUAL (GetClassValue ($ LTClass)
                                                 (QUOTE CVTest1))
                                  (GetClassValue LTKInst (QUOTE CVTest1]
                          (T (_@
                              Tested? NIL)
                             (printout TTY " TestMsg: CVs not being inherited properly" T)))
                       (COND
                          [(EQUAL (GetValue ($ LTClass)
                                            (QUOTE IVTest1))
                                  (GetValue LTKInst (QUOTE IVTest1]
                          (T (_@
                              Tested? NIL)
                             (printout TTY "TestMsg: IVs not being inherited properly" T)))
                       (PutValue LTKInst (QUOTE IVTest2)
```

```
                        (QUOTE IVal3))
                  (COND
                     ((EQUAL (GetValue ($ LTClass)
                                    (QUOTE IVTest2))
                             (GetValue LTKInst (QUOTE IVTest2)))
                      (_@
                       Tested? NIL)
                      (printout TTY "TestMsg: Inherited Values are overriding local values" T)))
                                                          (* TEST: Destroy the instance and its class)
                  (_ LTKInst Destroy)
                  (_ ($ LTClass)
                     Destroy)
                  [SETQ Temp (ERSETQ (GetClass ($ LTClass]
                  (COND
                     [(OR (NULL Temp)
                          (NULL (CAR Temp]
                     (T (printout TTY "Class Object not destroyed properly: GetClass returns non-NIL" T)))
                  (COND
                     ((NULL (@ Tested?)))
                     (T (_@
                         Tested? T)))
                  (printout TTY "Kernel features test  " -2 (COND
                                                               ((NULL (@ Tested?))
                                                                "failed.. Send bug report to
                                                                LOOPSCORE^.pa")
                                                               (T "Successfully!!"))
                        T)
                  (RETURN (@ Tested?)))))
```

(LOOPSTestObject.DefineTEST

```
(Method ((LOOPSTestObject DefineTEST)
         self)                                            (* sm: "20-SEP-83 16:27")
                                                          (* used to define Test description for a TestObj.)
                                                          (* This general proc may be used in conjunction with more specialized ones)
        (PROG (Name PreTest Cmd)
              (SETQ Name (@ name))
              (printout TTY " Please do not change the dependency lists in the editor" T)
              (SEND self Edit)
              (COND
                 ((NULL (GetValueOnly self (QUOTE PreTest)))
                  (AskPreTest self)))
              (COND
                 ((NULL (@ CasesUsed))
                  (AskTestCases self)))
              (COND
                 ((NULL (@ SyntaxTest))
                  (AskSyntaxTest self)))
              (COND
                 ((NULL (@ SubTest))
                  (AskSubTest self)))
              (RETURN self))))
```

(LOOPSTestObject.EditTEST

```
(Method ((LOOPSTestObject EditTEST)
         self)                                            (* sm: "20-SEP-83 16:28")
                                                          (* edits a TestObj using menus)
        (PROG ((EditTestObj self)
               (Stack (CONS self))
               (Redo T)
               INP2 INP)
              [COND
                 (EditTestWindows (for x in EditTestWindows do (OPENW x)))
                 (T (SETQ EditTestWindows (SetupEditTestObjMenu]
              (printout TTY "Editing Test Object:" -4 (@ name)
                     T)
          LOOP
              (COND
                 (Redo (PRIN1 "EditCmd:"))
                 (T (RETURN self)))
              [ERSETQ (SELECTQ (SETQ INP (READ))
                           ((Quit QUIT Q q)
                               [MAPC EditTestWindows (FUNCTION (LAMBDA (X)
                                                                  (CLOSEW X]
                               (SETQ Redo NIL))
                           (DEDIT (printout TTY "Please do not edit dependency link fields" T)
                                  (SEND EditTestObj Edit (QUOTE (de))))
                           ((EE VEDIT)
                               (printout TTY "Please do not edit dependency link fields" T)
                               (SEND EditTestObj Edit (QUOTE (ee))))
                           (EDIT (SEND EditTestObj Edit))
                           ((PP PP!)
                               (ERSETQ (DoMethod EditTestObj INP)))
                           (DESCRIBE (ERSETQ (SEND EditTestObj Describe)))
                           (WHO (printout TTY "Editing:" (ObjectName EditTestObj)
```

```
                                    T))
                              (TOP (SETQ Stack (LAST Stack))
                                   (SETQ EditTestObj (CAR Stack))
                                   (printout TTY "Editing:" -3 EditTestObj T))
                              (SELECT (PRIN1 "ObjName:")
                                      (SETQ INP2 (READ))
                                      (COND
                                         ((GetObjectRec INP2)
                                          (SETQ Stack (CONS EditTestObj Stack))
                                          (SETQ EditTestObj (GetObjectRec INP2)))
                                         (T (printout TTY INP2 -3 "Not an object. Redo command" T))))
                              (REMEMBER (SETQ Stack (CONS EditTestObj Stack)))
                              (UNREMEMBER (SETQ EditTestObj (GetObjectRec (CAR Stack)))
                                          (printout TTY "Editing:" (ObjectName EditTestObj)
                                                    T)
                                          [SETQ Stack (COND
                                                         ((EQ (LENGTH Stack)
                                                              1)
                                                          Stack)
                                                         (T (CDR Stack])
                              ((PT CU SUB ST AT UO PTO SUBO ATO STO CPT UB)
                                  [SETQ INP2 (EditPreTest EditTestObj (SELECTQ INP
                                                                         (PT (QUOTE PreTest))
                                                                         (SUB (QUOTE SubTest))
                                                                         (PTO (QUOTE PreTestOf))
                                                                         (CU (QUOTE CasesUsed))
                                                                         (ST (QUOTE SyntaxTest))
                                                                         (AT (QUOTE AltTest))
                                                                         (UO (QUOTE UsesObj))
                                                                         (SUBO (QUOTE SubTestOf))
                                                                         (ATO (QUOTE AltTestOf))
                                                                         (STO (QUOTE SyntaxTestOf))
                                                                         (printout TTY "Cannot edit along
                                                                                this link YET!" T]
                                      (COND
                                         (INP2 (SETQ Stack (APPEND (CDR INP2)
                                                                   (CONS EditTestObj Stack)))
                                               (SETQ EditTestObj (GetObjectRec (CAR INP2)))
                                               (printout TTY "Editing:" -4 (ObjectName EditTestObj)
                                                         T))))
                              (EVAL (UE))
                              (EXEC (ExecTestFields EditTestObj))
                              (TEST (ERSETQ (SEND EditTestObj TEST)))
                              (RETEST (ERSETQ (SEND EditTestObj ReTest)))
                              (TESTSUB (ERSETQ (SEND EditTestObj TEST!)))
                              (RESET (ERSETQ (SEND EditTestObj ResetSelf)))
                              (RESETPRE (ERSETQ (SEND EditTestObj Reset)))
                              (RESETSUB (ERSETQ (SEND EditTestObj Reset!)))
                              (? (printout TTY "Use the menu" T))
                              (COND
                                 ((NULL INP))
                                 (T (printout TTY "Illegal command. " -3 INP T]
                        (GO LOOP))))
```

(LOOPSTestObject.EditTestInTTYProcess

```
  (Method ((LOOPSTestObject EditTestInTTYProcess)
           self)                                                (* sm: "31-MAY-83 16:17")
                                                                (* calls EditTEST in TTY Process)
         (EVAL.IN.TTY.PROCESS (LIST (QUOTE _)
                                    self
                                    (QUOTE EditTEST)))))
```

(LOOPSTestObject.IgnoreTest

```
  (Method ((LOOPSTestObject IgnoreTest)
           self TestedLst)                                      (* sm: " 5-Jun-84 13:04")
                                                                (* marks test as failed because ignored)
                          (PutValue self (QUOTE Tested?)
                                 T
                                 (QUOTE DoneOnce))
                          (PutValue self (QUOTE Tested?)
                                 T
                                 (QUOTE Ignored))
                          (PutValue self (QUOTE Tested?)
                                 NIL)
                          (PrintIfLev LTMsgLev 8 (printout TTY "Test for " -4 (@ TestDesc)
                                                           -4 "was IGNORED" -5 T))
                          (COND
                             ([AND TestedLst (NOT (FMEMB self (CAR TestedLst]
                              (TCONC TestedLst self)))
                          (@ Tested?)))
```

(LOOPSTestObject.ReTest

```
  (Method ((LOOPSTestObject ReTest)
           self TestedLst)                                      (* sm: "20-SEP-83 16:32")
```

```
                                                        (* sends a ResetSelf if needed followed by TEST)
                        (PROG (Res)
                              (COND
                                 ((@ self Tested? DoneOnce)
                                  (_ self ResetSelf)))
                              (SETQ Res (_ self TEST TestedLst))
                              (COND
                                 ((EQ Res T)
                                  (printout TTY "Test was successful!!" T)))
                              (RETURN Res))))
```

(LOOPSTestObject.ReTestDep

```
(Method ((LOOPSTestObject ReTestDep)
         self)                                          (* sm: "30-NOV-82 12:55")
                                                        (* first sends ResetDep and then TESTDep to self)
              (_ self ResetDep)
              (_ self TESTDep)))
```

(LOOPSTestObject.Reset!

```
(Method ((LOOPSTestObject Reset!)
         self ResetLst)                                 (* sm: "25-NOV-82 16:01")
```

(* resets itself by sending Reset to self and also Reset! its SyntaxTest, AltTest and SubTest.
Also does a Reset on PreTest and ClassPreTest)

(* ResetLst is a list to which obj names are added as they are reset.
It is initially NIL)

```
                      (PROG (HELPFLAG)                  (* If ResetLst is NIL, create a pointer for it)
                            [COND
                               ((NULL ResetLst)
                                (SETQ ResetLst (CONS]
                            [COND
                               ((NOT (FMEMB (@ name)
                                            (CAR ResetLst)))
                                (TCONC ResetLst (@ name))
                                (for x in (@ AltTest) do (_ (GetObjectRec x)
                                                            Reset! ResetLst))
                                (for x in (@ SyntaxTest) do (_ (GetObjectRec x)
                                                               Reset! ResetLst))
                                (for x in (@ SubTest) do (_ (GetObjectRec x)
                                                            Reset! ResetLst))
                                (_ self ResetSelf ResetLst)
                                (for x in (@ PreTest) do (_ (GetObjectRec x)
                                                            Reset ResetLst))
                                (for x in (@@ ClassPreTest) do (_ (GetObjectRec x)
                                                                  Reset ResetLst]
                            (RETURN ResetLst))))
```

(LOOPSTestObject.ResetAll

```
(Method ((LOOPSTestObject ResetAll)
         self ResetLst)                                 (* sm: "29-NOV-82 15:16")
```

(* completely resets itself, all tests on which it depends, and all which depend on it)

(* ResetLst is a list to which obj names are added as they are reset.
It is initially NIL)

```
                      (PROG (HELPFLAG)                  (* If ResetLst is NIL, create a pointer for it)
                            [COND
                               ((NULL ResetLst)
                                (SETQ ResetLst (CONS]
                            [COND
                               ((NOT (FMEMB (@ name)
                                            (CAR ResetLst)))
                                (TCONC ResetLst (@ name))
                                (for x in (@ AltTest) do (_ (GetObjectRec x)
                                                            ResetAll ResetLst))
                                (for x in (@ SyntaxTest) do (_ (GetObjectRec x)
                                                               ResetAll ResetLst))
                                (for x in (@ SubTest) do (_ (GetObjectRec x)
                                                            ResetAll ResetLst))
                                (for x in (@ PreTestOf) do (_ (GetObjectRec x)
                                                              ResetAll ResetLst))
                                (_ self ResetSelf ResetLst)
                                (for x in (@ PreTest) do (_ (GetObjectRec x)
                                                            ResetAll ResetLst))
                                (for x in (@@ ClassPreTest) do (_ (GetObjectRec x)
                                                                  ResetAll ResetLst]
                            (RETURN ResetLst))))
```

(LOOPSTestObject.ResetDep

```
(Method ((LOOPSTestObject ResetDep)
```

```
          self ResetLst)                                        (* sm: "29-NOV-82 15:08")
                                                                (* resets itself and all tests which depend on it)

          (* resets itself by sending Reset to self and also Reset! its SyntaxTest, AltTest and SubTest.
          Also does a Reset on PreTestOf list)

          (* ResetLst is a list to which obj names are added as they are reset.
          It is initially NIL)

                        (PROG (HELPFLAG)                        (* If ResetLst is NIL, create a pointer for it)
                              [COND
                                 ((NULL ResetLst)
                                  (SETQ ResetLst (CONS]
                              (COND
                                 ((NOT (FMEMB (@ name)
                                              (CAR ResetLst)))
                                  (TCONC ResetLst (@ name))
                                  (for x in (@ AltTest) do (_ (GetObjectRec x)
                                                              ResetDep ResetLst))
                                  (for x in (@ SyntaxTest) do (_ (GetObjectRec x)
                                                                 ResetDep ResetLst))
                                  (for x in (@ SubTest) do (_ (GetObjectRec x)
                                                              ResetDep ResetLst))
                                  (for x in (@ PreTestOf) do (_ (GetObjectRec x)
                                                                ResetDep ResetLst))
                                  (_ self ResetSelf ResetLst)))
                              (RETURN ResetLst))))
```

(LOOPSTestObject.TEST

```
  (Method ((LOOPSTestObject TEST)
           self TestedLst)                                      (* sm: "20-SEP-83 16:28")
                                                                (* performs the basic TEST for a TestObject)
                          (PROG (FMSG (TestingObject self)
                                      PrevRes HELPFLAG (HeaderFlag T))
                                                                (* HeaderFlag is used by: PreTestsSatisfied?, DoTestSelf)
                                (CloseCurrentEnvironment)
                                (AND CurrentEnvironment (SEND CurrentEnvironment MakeNotCurrent))
                                (PutValue self (QUOTE Tested?)
                                       T
                                       (QUOTE DoneOnce))
                                (COND
                                   ([AND TestedLst (NOT (FMEMB self (CAR TestedLst]
                                    (TCONC TestedLst self)))
                                (COND
                                   ((ValueExists? (SETQ PrevRes (ExaminePreviousTry self TestedLst)))
                                    (PrintIfLev LTMsgLev 7 (printout TTY "Test for " -4 (@ TestDesc)
                                                                     -4 "was" -5 (COND
                                                                                    (PrevRes "Successful")
                                                                                    (T "Unsuccessful"))
                                                                     T))
                                    (RETURN PrevRes)))          (* check pretests)
                                (AND LTBROWSER (_ LTBROWSER BoxNode self))
                                (COND
                                   ((EQ (SETQ Res (PreTestsSatisfied? self TestedLst))
                                        T))
                                   (T                           (* pretests failed. Return)
                                      (AND LTBROWSER (_ LTBROWSER BoxNode self))
                                      (RETURN Res)))
                                (RETURN (DoTestSelf self TestedLst)))))
```

(LOOPSTestObject.TEST!

```
  (Method ((LOOPSTestObject TEST!)
           self ContFl)                                         (* sm: "20-SEP-83 16:29")
                                                                (* first TESTs itself, and if successful, then does other tests-
                                                                SubTest, SyntaxTest, AltTest.)
                                                                (* Sets IV CompletelyTested? if SubTest and SyntaxTest are
                                                                successful)

          (* ContFl if T, then continues, even if basic test failed. this may be useful in some cases)

                        (PROG (TestRes)                         (* Check if already CompletelyTested?)
                              [COND
                                 ((ValueExists? (@ CompletelyTested?))
                                  (printout TTY "Overall Test for" -4 (@ TestDesc)
                                            -4 "was" -5 (COND
                                                           ((@ CompletelyTested?)
                                                            "Successful")
                                                           (T "Unsuccessful"))
                                            T)
                                  (RETURN (@ CompletelyTested?]
                              (SEND self TEST)
                              [COND
                                 ([AND (NOT (EQ ContFl T))
                                       (NOT (ValueNonNIL? (GetValue self (QUOTE Tested?]
                                  (printout TTY "Not proceeding with complete test of.." -4 (@ TestDesc)
```

```
                                    T)
                                (_@
                                 CompletelyTested? NotSetValue)
                                (RETURN (@ CompletelyTested?]
                            (COND
                               ([NOT (ValueNonNIL? (GetValue self (QUOTE Tested?]
                                (printout TTY "Proceeding with complete test, even though this feature failed.
                                      Interpret results CAREFULLY" T)))
                            (DoLoopsTest self (QUOTE SubTest)
                                   (QUOTE TEST!)
                                   ContFl)
                            (DoLoopsTest self (QUOTE SyntaxTest)
                                   (QUOTE TEST!)
                                   ContFl)
                            (DoLoopsTest self (QUOTE AltTest)
                                   (QUOTE TEST!)
                                   ContFl)
                            [PutValue self (QUOTE CompletelyTested?)
                                   (PROGN [SETQ TestRes (LIST (@ Tested?)
                                                              (GetValue self (QUOTE SubTest)
                                                                     (QUOTE Tested?))
                                                              (GetValue self (QUOTE SyntaxTest)
                                                                     (QUOTE Tested?]
                                          (COND
                                             ((FMEMB NotSetValue TestRes)
                                              NotSetValue)
                                             ((FMEMB NIL TestRes)
                                              NIL)
                                             (T T]
                            (printout TTY "Overall test for" -4 (@ TestDesc)
                                   -4 "was" -5 (SELECTQ (@ CompletelyTested?)
                                                        (NIL "Unsuccessful")
                                                        (T "Successful")
                                                        (QUOTE "Indeterminate"))
                                   -2 TestRes T)
                            (RETURN (@ CompletelyTested?)))))
```

(LOOPSTestObject.TESTDep

```
  (Method ((LOOPSTestObject TESTDep)
           self)                                                    (* sm: "13-DEC-82 13:15")

          (* generates TEST call to self, and TESTDep to PreTestOf list, and SubTest list only if it succeeds)
                                                                    (* Returns list of objects called)
                 (PROG (Called)
                       (SETQ Called (LIST self))
                       (SEND self TEST)
                       [COND
                          ((ValueNonNIL? (@ Tested?))
                           (NCONC Called (for x in (GetValue self (QUOTE SubTest))
                                              join (_ (GetObjectRec x)
                                                      TESTDep))
                                  (for x in (GetValue self (QUOTE PreTestOf)) join (_ (GetObjectRec x)
                                                                                      TESTDep]
                       (RETURN Called))))
```

(LOOPSTestObject.TESTall

```
  (Method ((LOOPSTestObject TESTall)
           self)                                                    (* sm: "26-OCT-82 11:35")
                                                                    (* generates TEST call to self, PreTestOf list, and SubTest list)
                                                                    (* Returns list of objects called)
                 (PROG (Called)
                       (SETQ Called (LIST self))
                       (SEND self TEST)
                       (NCONC Called (for x in (GetValue self (QUOTE SubTest)) join (SEND (GetObjectRec x)
                                                                                          TESTall))
                              (for x in (GetValue self (QUOTE PreTestOf)) join (SEND (GetObjectRec x)
                                                                                     TESTall)))
                       (RETURN Called))))
```

(LOOPSTestObject.TestSelf

```
  (Method ((LOOPSTestObject TestSelf)
           self TestedLst)                                          (* sm: "20-SEP-83 16:29")
                                                                    (* performs the basic TEST for a TestObject but continues even
                        if PreTests fail)
                        (PROG (FMSG (TestingObject self)
                                    PrevRes HELPFLAG (HeaderFlag T))
                                                                    (* HeaderFlag is used by: PreTestsSatisfied?, DoTestSelf)
                              (CloseCurrentEnvironment)
                              (AND CurrentEnvironment (SEND CurrentEnvironment MakeNotCurrent))
                              (PutValue self (QUOTE Tested?)
                                     T
                                     (QUOTE DoneOnce))
                              (COND
                                 ([AND TestedLst (NOT (FMEMB self (CAR TestedLst]
```

```
                                    (TCONC TestedLst self)))
                                (COND
                                   ((ValueExists? (SETQ PrevRes (ExaminePreviousTry self TestedLst)))
                                    (PrintIfLev LTMsgLev 7 (printout TTY "Test for " -4 (@ TestDesc)
                                                                  -4 "was" -5 (COND
                                                                                 (PrevRes "Successful")
                                                                                 (T "Unsuccessful"))
                                                                  T))
                                    (RETURN PrevRes)))          (* check pretests)
                                (AND LTBROWSER (_ LTBROWSER BoxNode self))
                                (COND
                                   ((EQ (SETQ Res (PreTestsSatisfied? self TestedLst))
                                        T))
                                   (T                           (* pretests failed. Return)
                                      (printout TTY "PreTests failed for" -4 (@ TestDesc)
                                             -4 "Nonethless, continuing with test - interpret results
                                             accordingly" T)))
                                (RETURN (DoTestSelf self TestedLst)))))
```

(**LOOPSTestPrimitive.XTEST**

```
  (Method ((LOOPSTestPrimitive XTEST)
           self)                                                (* sm: "20-SEP-83 16:30")
```

(* performs the basic TEST for a Primitive TestObject ONLYIF its super test Tested T and sets own Tested? IV)

```
          [PROG (TestExp Resp Sval Tval)
                [COND
                   ((ValueExists? (@ Tested?))
                    (printout TTY "Basic Test for " -4 (@ TestDesc)
                           -4 "was" -5 (COND
                                          ((@ Tested?)
                                           "Successful")
                                          (T "Unsuccessful"))
                           T)
                    (RETURN (@ Tested?]
                (printout TTY "TESTING.." -4 (@ TestDesc)
                     T)
                (COND
                   ((NOT (EQ (CheckClassTest self)
                             T))
                    (printout TTY 5 "Basic test for this class failed. Stopping.." T)
                    (RETURN NotSetValue)))              (* The class PreTest were successful so continue...)
                (COND
                   ((NOT (EQ (CheckPreTest self)
                             T))
                    (printout TTY 5 "Basic pretests failed. Cannot test further.." T)
                    (RETURN NotSetValue)))              (* Now test the object)
                (SETQ TestExp (@ TestExpr))
                [COND
                   ((NULL TestExp)
                    (printout TTY 5 "No test is currently available for.." -3 (@ TestDesc)
                           T 5 "Indicate if this feature works OK?")
                    (SETQ Resp (READ))
                    (RETURN (COND
                               ((FMEMB Resp (QUOTE (y t Y Yes T True YES TRUE)))
                                (_@
                                 Tested? T))
                               ((FMEMB Resp (QUOTE (n f N No F FALSE NO False nil NIL)))
                                (_@
                                 Tested? NIL))
                               (T NotSetValue]
                (SETQ Sval (ERRORSET (@ SetUp)
                                     T))
                (COND
                   ((NULL Sval)
                    (printout TTY 5 "Error in setting up the test environment. Cannot test further.." T)
                    (RETURN NotSetValue)))
                (SETQ Tval (ERRORSET (@ TestExpr)
                                     T))
                (COND
                   ((OR (NULL Tval)
                        (NULL (CAR Tval)))
                    (printout TTY "Test failed for.." (@ TestDesc)
                           T 10 "Send bug report to LOOPSCORE^.pa" T)
                    (_@
                     Tested? NIL))
                   (T (printout TTY "Test successful!! for.." (@ TestDesc T)
                              (_@
                               Tested? T)))
                   (RETURN (@ Tested?])))

)

(DEFINEQ
```

(**AddAltTest**

```
  [LAMBDA (self)                                                    (* sm: "29-OCT-82 15:29")
                                                                    (* offers to help add AltTest. Returns T -
                                                                    if any added. NIL -
                                                                    otherwise)
                                                                    (* currently a no op)

    (PROG NIL
          (RETURN NIL])
```

(AllowRemove

```
  [LAMBDA (self varName newValue propName activeVal type)           (* sm: "11-OCT-82 15:42")

          (* This is a putFn for allowing values to be removed by using the format
          (-
          v)%. Any other kind will be added as such)

    (COND
       ((ATOM newValue)
        (PutLocalState activeVal (CONS newValue (GetLocalState activeVal self varName propName))
               self varName propName type))
       ((EQ (CAR newValue)
            (QUOTE -))
        (PutLocalState activeVal (DREMOVE (CADR newValue)
                                          (GetLocalState activeVal self varName propName))
               self varName propName type))
       (T (PutLocalState activeVal newValue self varName propName type])
```

(AskPreTest

```
  [LAMBDA (self)                                                    (* sm: "20-SEP-83 16:22")
                                                                    (* ask for changes to PreTests)

    (PROG (PreTest)
          (SETQ PreTest (ASKUSER NIL NIL (LIST "Current PreTests are:" (@ PreTest)
                                               "Do you want to add/delete any")
                                 (QUOTE ((A "dd
                                            " EXPLAINSTRING "Add - enter the ADDITIONS to PreTest list" KEYLST
                                            ((Ÿ NIL RETURN ANSWER CONFIRMFLG T)))
                                         (M "odify
                                            " EXPLAINSTRING "Modify - enter the NEW PreTest list" KEYLST
                                            (((READ)
                                              NIL RETURN ANSWER CONFIRMFLG T)))
                                         (D "elete
                                            " EXPLAINSTRING "Delete - enter the list of PreTests to be deleted"
                                            KEYLST (((READ)
                                                     NIL RETURN ANSWER CONFIRMFLG T)))
                                         (N "o
                                            " EXPLAINSTRING "No - no change" RETURN NIL)))
                                 NIL NIL (QUOTE (CONFIRMFLG NIL))
                                 NIL))
          [COND
             (PreTest (SELECTQ (CAR PreTest)
                               ((A Add)
                                    (for x in (MKLIST (CADR PreTest)) do (PutValue self (QUOTE PreTest)
                                                                                   x)))
                               ((M Modify)
                                    (for x in (@ PreTest) do (RemoveValue self (QUOTE PreTest)
                                                                          x))
                                    (for y in (MKLIST (CADR PreTest)) do (PutValue self (QUOTE PreTest)
                                                                                   x)))
                               ((D Delete)
                                    (for x in (MKLIST (CADR PreTest)) do (RemoveValue self (QUOTE PreTest)
                                                                                   x)))
                               (printout TTY "Illegal format PreTest list. Ignoring.." PreTest T]
          (printout TTY "New PreTests are:" T (@ PreTest)
                 T])
```

(AskSubTest

```
  [LAMBDA (self)                                                    (* sm: "20-SEP-83 16:22")
                                                                    (* ask for changes to SubTest list)

    (PROG (PreTest)
          (SETQ PreTest (ASKUSER NIL NIL (LIST "Current SubTest objects are:" (@ SubTest)
                                               "Do you want to add/delete any")
                                 (QUOTE ((A "dd
                                            " EXPLAINSTRING "Add - enter the ADDITIONS to SubTest list" KEYLST
                                            (((READ)
                                              "a number of new Primitive or a list of existing SubTest objects"
                                              RETURN ANSWER CONFIRMFLG T)))
                                         (M "odify
                                            " EXPLAINSTRING "Modify - enter the NEW SubTest list" KEYLST
                                            (((READ)
                                              NIL RETURN ANSWER CONFIRMFLG T)))
                                         (D "elete
                                            " EXPLAINSTRING "Delete - enter the list of SubTests to be deleted"
                                            KEYLST (((READ)
                                                     NIL RETURN ANSWER CONFIRMFLG T)))
                                         (N "o
```

```
                                    " EXPLAINSTRING "No - no change" RETURN NIL)))
                            NIL NIL (QUOTE (CONFIRMFLG NIL))
                            NIL))
          [COND
             (PreTest (SELECTQ (CAR PreTest)
                         ((A Add)
                              [COND
                                 [(NUMBERP (CADR PreTest))
                                  (RPTQ (CADR PreTest)
                                        (PutValue self (QUOTE SubTest)
                                               (GetValue (SEND ($ LOOPSTestPrimitive)
                                                               New)
                                                         (QUOTE name]
                                 (T (for x in (MKLIST (CADR PreTest)) do (PutValue self (QUOTE SubTest)
                                                                                   x])
                         ((M Modify)
                              (for x in (@ SubTest) do (RemoveValue self (QUOTE SubTest)
                                                                    x))
                              (for y in (MKLIST (CADR PreTest)) do (PutValue self (QUOTE SubTest)
                                                                              x)))
                         ((D Delete)
                              (for x in (MKLIST (CADR PreTest)) do (RemoveValue self (QUOTE SubTest)
                                                                              x)))
                         (printout TTY "Illegal format SubTest list. Ignoring.." PreTest T]
          (printout TTY "New SubTests are:" T (@ SubTest)
                 T])
```

(AskSyntaxTest

```
  [LAMBDA (self)                                                    (* edited: "25-Mar-85 16:13")
                                                                    (* ask for changes to Syntax test list)
    (PROG (PreTest)
          (SETQ PreTest (ASKUSER NIL NIL (LIST "Current SyntaxTest objects are:" (@ SyntaxTest)
                                               "Do you want to add/delete any")
                                (QUOTE ((A "dd
                                           " EXPLAINSTRING "Add - enter the ADDITIONS to SyntaxTest list" KEYLST
                                           (((READ)
                                             "a number of new or a list of existing SyntaxTest objects" RETURN
                                             ANSWER CONFIRMFLG T)))
                                       (M "odify
                                          " EXPLAINSTRING "Modify - enter the NEW SyntaxTest list" KEYLST
                                          (((READ)
                                            NIL RETURN ANSWER CONFIRMFLG T)))
                                       (D "elete
                                          " EXPLAINSTRING "Delete - enter the list of SyntaxTests to be
                                          deleted" KEYLST (((READ)
                                                            NIL RETURN ANSWER CONFIRMFLG T)))
                                       (N "o
                                          " EXPLAINSTRING "No - no change" RETURN NIL)))
                                NIL NIL (QUOTE (CONFIRMFLG NIL))
                                NIL))
          [COND
             (PreTest (SELECTQ (CAR PreTest)
                         ((A Add)
                              [COND
                                 [(NUMBERP (CADR PreTest))
                                  (RPTQ (CADR PreTest)
                                        (PutValue self (QUOTE SyntaxTest)
                                               (GetValue (SEND ($ LOOPSTestSyntax)
                                                               New)
                                                         (QUOTE name]
                                 (T (for x in (MKLIST (CADR PreTest)) do (PutValue self (QUOTE SyntaxTest)
                                                                                   x])
                         ((M Modify)
                              (for x in (@ SyntaxTest) do (RemoveValue self (QUOTE SyntaxTest)
                                                                       x))
                              (for x in (MKLIST (CADR PreTest)) do (PutValue self (QUOTE SyntaxTest)
                                                                              x)))
                         ((D Delete)
                              (for x in (MKLIST (CADR PreTest)) do (RemoveValue self (QUOTE SyntaxTest)
                                                                              x)))
                         (printout TTY "Illegal format SyntaxTest list. Ignoring.." PreTest T]
          (printout TTY "New SyntaxTests are:" T (@ SyntaxTest)
                 T])
```

(AskTestCases

```
  [LAMBDA (self)                                                    (* sm: "20-SEP-83 16:23")
                                                                    (* ask for changes to TestCases list)
    (PROG (PreTest)
      TC  (SETQ PreTest (ASKUSER NIL NIL (LIST "Current TestCase objects are:" (@ CasesUsed)
                                               "Do you want to add/delete any")
                                (QUOTE ((A "dd
                                           " EXPLAINSTRING "Add - enter the number of new TestCase objects to be
                                           created" KEYLST (((READ)
                                                             NIL RETURN ANSWER CONFIRMFLG T)))
                                       (D "elete
```

```
                                        " EXPLAINSTRING "Delete - enter the list of TestCases to be deleted"
                                        KEYLST (((READ)
                                                 NIL RETURN ANSWER CONFIRMFLG T)))
                                     (N "o
                                        " EXPLAINSTRING "No - no change" RETURN NIL)))
                               NIL NIL (QUOTE (CONFIRMFLG NIL))
                               NIL))
          [COND
             (PreTest (SELECTQ (CAR PreTest)
                               ((A Add)
                                (COND
                                   ((NOT (NUMBERP (CADR PreTest)))
                                    (printout TTY "Specify a number. " T)
                                    (GO TC)))
                                [RPTQ (CADR PreTest)
                                      (PutValue self (QUOTE CasesUsed)
                                             (GetValue (SEND ($ LOOPSTestCases)
                                                             New)
                                                    (QUOTE name])
                               ((D Delete)
                                (for x in (MKLIST (CADR PreTest)) do (RemoveValue self (QUOTE CasesUsed)
                                                                                  x)))
                               (printout tTTY "Illegal format CasesUsed list. Ignoring.." PreTest T]
          (printout TTY "New TestCases are:" T (@ CasesUsed)
                 T])
```

(BuildPreTest

```
  [LAMBDA (self varName localSt propName activeVal type)        (* sm: " 8-OCT-82 11:25")
                                                                (* This is a getFn for PreTest in LOOPSTestPrimitive)
```

(* it appends SubTestOf to the front of the local state and returns that as the actual PreTest list)

(* ASSUMPTION: This now forces all SubTestOf objects to be considered as PreTests.
If this assumption is later violated, this scheme of building PreTest list will have to be undone)

```
    (APPEND (@ SubTestOf)
           (GetLocalState activeVal self varName propName])
```

(BuildPrimClassTest

```
  [LAMBDA (self varName localSt propName activeVal type)        (* sm: " 8-OCT-82 10:56")
                                                                (* This is a getFn for ClassPreTest CV in LOOPSTestPrimitive)
```

(* it adds the value of IV SubTestOf to any values already in the CV ClassPreTest to compute the actual ClassPreTest list.
Normally, no separate values are planned for ClassPreTest. Currently, the main purpose of this active value is to return the
SubTestOf list as the ClassPreTest)

```
    (APPEND (@ SubTestOf)
           (GetLocalState activeVal self varName propName])
```

(CheckClassTest

```
  [LAMBDA (self TestedLst)                                      (* sm: "20-SEP-83 16:24")
                                                                (* Checks the Class Pretests of a TestObj)
    (PROG (TestRes Mat)                                         (* check if class already tested)
          [COND
             ((ValueExists? (@@ ClassTested?))
              (RETURN (@@ ClassTested?]                         (* if class has no pretest, return T)
          (COND
             ((NULL (@@ ClassPreTest))
              (PrintIfLev LTMsgLev 5 (printout TTY 10 "SysNote: No class pretests for" -4 (@ name)
                                             T))
              (RETURN T)))
          [SETQ TestRes (MAPCAR (@@ ClassPreTest)
                                (QUOTE (LAMBDA (x)
                                               (CONS (SEND (GetObjectRec x)
                                                           TEST TestedLst)
                                                     x]
          (COND
             ((SETQ Mat (FASSOC NIL TestRes))
              (PutClassValue self (QUOTE ClassPreTest)
                     (CDR Mat)
                     (QUOTE Failed))
              (_@@
               ClassTested? NIL))
             ((SETQ Mat (FASSOC NotSetValue TestRes))
              (PutClassValue self (QUOTE ClassPreTest)
                     (CDR Mat)
                     (QUOTE Failed))
              (_@@
               ClassTested? NotSetValue))
             (T (_@@
                 ClassTested? T)))
          (RETURN (@@ ClassTested?])
```

(CheckPreTest

```
[LAMBDA (self)                                                    (* sm: "20-SEP-83 16:24")
                                                                  (* checks the PreTest of a TestObj)
   (PROG (TestRes)                                                (* see if already tested)
         [COND
            ((ValueExists? (GetValue self (QUOTE PreTest)
                                   (QUOTE Tested?)))
             (RETURN (GetValue self (QUOTE PreTest)
                           (QUOTE Tested?]
         (COND
            ((NULL (@ PreTest))
             (printout TTY 10 "SysNote: No PreTests for  " -4 (@ name)
                    T)
             (RETURN T)))
         [SETQ TestRes (MAPCAR (@ PreTest)
                               (QUOTE (LAMBDA (X)
                                              (SEND (GetObjectRec X)
                                                    TEST]
         [COND
            ((FMEMB NIL TestRes)
             (PutValue self (QUOTE PreTest)
                    NIL
                    (QUOTE Tested?)))
            ((FMEMB NotSetValue TestRes)
             (PutValue self (QUOTE PreTest)
                    NotSetValue
                    (QUOTE Tested?)))
            (T (PutValue self (QUOTE PreTest)
                      T
                      (QUOTE Tested?]
         (RETURN (GetValue self (QUOTE PreTest)
                       (QUOTE Tested?])
```

(CreateLTKBS1

```
[LAMBDA NIL                                                       (* sm: "14-Dec-84 10:09")
                                                                  (* creates the knowledge base LTKBS1 used by LTESummarize)
   (PROG NIL
         (_ ($ KB)
            New
            (QUOTE LTKBS1)
            (QUOTE X1)
            T)
         (_ ($ X1)
            Open)
         (_ ($ LOOPSCasesMeta)
            New
            (QUOTE LOOPSEnvSumm)
            (QUOTE (LOOPSTestCases)))
         (_ ($ LOOPSEnvSumm)
            Add
            (QUOTE CV)
            (QUOTE Instances)
            NIL)
         (_ ($ LOOPSEnvSumm)
            Add
            (QUOTE CV)
            (QUOTE InstancePrefix)
            (QUOTE LESA))
         (_ ($ LOOPSEnvSumm)
            Add
            (QUOTE CV)
            (QUOTE UnnamedInstanceCount)
            0)
         (_ ($ LOOPSEnvSumm)
            Add
            (QUOTE CV)
            (QUOTE InstanceComsVar)
            (QUOTE DUMMYINSTANCES))
         (_ ($ LOOPSEnvSumm)
            Add
            (QUOTE IV)
            (QUOTE Etest)
            NIL)
         (_ ($ LOOPSEnvSumm)
            Add
            (QUOTE IV)
            (QUOTE Link)
            NIL)
         (_ ($ LOOPSEnvSumm)
            New
            (QUOTE LESAA))
         (_ ($ LOOPSEnvSumm)
            New)
         (PutValue ($ LESAA)
                (QUOTE Etest)
                (QUOTE Old))
```

```
          (_ ($ Class)
             New
             (QUOTE LOOPSEnvTest))
          (_ ($ LOOPSEnvTest)
             Add
             (QUOTE IV)
             (QUOTE Key)
             NIL)
          (PutValue ($ LESAA)
                 (QUOTE Link)
                 (_ ($ LOOPSEnvTest)
                    New))
          (PutValue (GetValue ($ LESAA)
                           (QUOTE Link))
                 (QUOTE Key)
                 (QUOTE LESAA))
          [SETQ LTV11 (UID (GetValue ($ LESAA)
                                   (QUOTE Link]
          (_ ($ X1)
             Cleanup)
          (_ ($ X1)
             Close)
          (_ ($ KB)
             Old
             (QUOTE LTKBS1)
             (QUOTE X2))
          (_ ($ X2)
             Open)
          (PutValue ($ LESAA)
                 (QUOTE Etest)
                 (QUOTE New))
          (_ ($ LOOPSEnvSumm)
             New)
          (PutValue ($ LESA2)
                 (QUOTE Etest)
                 (QUOTE LESA2))
          (_ ($ X2)
             Close)
          (_ ($ KB)
             Old
             (QUOTE LTKBS1)
             (QUOTE X3))
          (_ ($ X3)
             Open)
          (_ ($ LESA1)
             Destroy)
          (PutValue ($ LESA2)
                 (QUOTE Link)
                 (_ ($ LOOPSEnvTest)
                    New))
          (PutValue (GetValue ($ LESA2)
                           (QUOTE Link))
                 (QUOTE Key)
                 (QUOTE LESA2))
          [SETQ LTV12 (UID (GetValue ($ LESA2)
                                   (QUOTE Link]
          (PutValue ($ LESAA)
                 (QUOTE Link)
                NIL)
          (_ ($ X3)
             Close)
          (RETURN (LIST (QUOTE (LOOPSEnvSumm LOOPSEnvTest))
                        (QUOTE (LESAA LESA2))
                        (QUOTE (LESA1))
                        (LIST (QUOTE LTV11)
                              LTV11)
                        (LIST (QUOTE LTV12)
                              LTV12])
```

(**DescribePreviousTry**

```
  [LAMBDA (self)                                          (* sm: " 5-Jun-84 10:51")
                                                          (* describes result of previous test of this obj)

    (PROG NIL
          (RETURN (COND
                     ((ValueNonNIL? (GetValue self (QUOTE Tested?)
                                              (QUOTE Ignored)))
                      (printout TTY "Test was IGNORED" T)
                      NIL)
                     ((ValueNonNIL? (ExaminePreviousTry self))
                      (printout TTY "Test was successful!!" T)
                      T)
                     ((AND (@@ ClassPreTest)
                           (NOT (EQ (@@ ClassTested?)
                                    T)))
                      (printout TTY "Following PreTest for this Class of tests failed:" T (@@ self ClassPreTest
                                                                                             Failed)
```

```
                        T)
                  NIL)
                 ((AND (@ PreTest)
                       (NOT (EQ (@ self PreTest Tested?)
                                T)))
                  (printout TTY "Following PreTests failed:" (@ self PreTest Failed)
                        T)
                  NIL)
                 ((AND (@ SetUp)
                       (NOT (EQ (@ self SetUp Tested?)
                                T)))
                  (printout TTY "Following SetUp expression(s) caused error:" T)
                  (PrintFailedExp self (@ self SetUp FailedExp))
                  NIL)
                 ((NULL (@ TestExpr))
                  (printout TTY "No test is available yet." T)
                  T)
                 (T (printout TTY "Following TestExpression(s) failed or caused error:" T)
                    (PrintFailedExp self (@ self TestExpr FailedExp))
                    NIL])
```

(DescribeTestLink

```
  [LAMBDA (self link Desc Selected)                                    (* sm: "20-SEP-83 16:25")
                                                                       (* describes the values of link, using a menu of
                                                                       description-choices)
    (PROG ((vals (GetValue self link))
           obj)                                                        (* Desc -
                                                                       if NIL, user is asked for a value)
                                                                       (* Selected -
                                                                       if NIL user is asked.)
          (COND
             ((NULL vals)
              (RETURN NIL)))
          [COND
             (Desc)
             (T (SETQ Desc (MENU (create MENU
                                         ITEMS _ (QUOTE (TestDesc TestDescAll TestDescAsk TestCode PP))
                                         TITLE _ "Desc What"]
          (COND
             ((NULL Desc)
              (RETURN NIL)))
          [COND
             (Selected)
             (T (SETQ Selected (ReadLinkMenu self link T T]
          [for x in Selected do (PROGN (SETQ obj (GetObjectRec x))
                                       (SELECTQ Desc
                                           (PP (SEND obj PP))
                                           (TestCode (PrintTestCode obj))
                                           (TestDesc (printout TTY (ObjectName obj)
                                                             -5
                                                             (TestObjectDesc obj)
                                                             T))
                                           (TestDescAll (printout TTY (ObjectName obj)
                                                                 -5
                                                                 (TestObjectDesc obj)
                                                                 T)
                                                        (DescribeTestLink obj link (QUOTE TestDescAll)
                                                                 (GetValue obj link)))
                                           (TestDescAsk (printout TTY (ObjectName obj)
                                                                 -5
                                                                 (TestObjectDesc obj)
                                                                 T)
                                                        (DescribeTestLink obj link (QUOTE TestDescAsk)
                                                                 NIL))
                                           (printout TTY "Error!! Should not reach here in func:
                                                  DescribeTestLink" T]
          (RETURN Selected])
```

(DisplayTestBrowser

```
  [LAMBDA (Objs BrowserClass Link Region Title FlipFlg)                (* sm: " 3-DEC-82 09:55")

          (* displays a TestBrowser with "Objs" , using "Link" , in a window bounded by "Region" and "Title" %.
          All except the first arg can be defaulted to whats in the class ($ TestBrowser))

          (* FlipFlg -
          if T then all objects which have Tested? IV set to T will be flipped)
                                                                       (* binds global LTBROWSER to the browser created)
    (PROG NIL
          (COND
             ((NULL Objs)
              (RETURN NIL)))
          [COND
             ((NULL BrowserClass)
              (SETQ BrowserClass (QUOTE TestBrowser]
          (SETQ LTBROWSER (_ (GetObjectRec BrowserClass)
```

```
                          New))
          [COND
             (Link (PutValue LTBROWSER (QUOTE subLinks)
                         (MKLIST Link]
          (COND
             (Title (PutValue LTBROWSER (QUOTE title)
                          Title)))
          [_ LTBROWSER Show [SETQ Objs (MAPCAR Objs (QUOTE (LAMBDA (X)
                                                             (GetObjectRec X]
             (CREATEW [COND
                        (Region)
                        (T (GetValue LTBROWSER (QUOTE window)
                                   (QUOTE DefRegion]
                    (GetValue LTBROWSER (QUOTE title]
          (COND
             (FlipFlg (TickleBrowserNodes Objs LTBROWSER)))
          (RETURN LTBROWSER])
```

(DoLoopsTest

```
  [LAMBDA (self testVar testMsg contFl TestedLst)              (* sm: "20-SEP-83 16:25")
```

(* tests the objs indicated by IV testVar, using msg testMsg, and force test flag contFl)

```
    (PROG (TestRes Mat)                                        (* set up default values if not passed)
          (COND
             ((NULL testVar)
              (printout TTY "No Test IV given." -2 "Aborting.." T)
              (RETURN NIL)))
          [COND
             ((NULL testMsg)
              (SETQ testMsg (QUOTE TEST]                       (* see if already tested)
          [COND
             ((ValueExists? (GetValue self testVar (QUOTE Tested?)))
              (RETURN (GetValue self testVar (QUOTE Tested?]
          (COND
             ((NULL (GetValue self testVar))
              (PrintIfLev LTMsgLev 5 (printout TTY 10 "SysNote: No " testVar " for  " -4 (@ name)
                                          T))
              (PutValue self testVar T (QUOTE Tested?))
              (RETURN T)))
          [SETQ TestRes (MAPCAR (GetValue self testVar)
                                (QUOTE (LAMBDA (X)
                                             (CONS (DoMethod (GetObjectRec X)
                                                             testMsg NIL TestedLst contFl)
                                                   X]
          [COND
             ((SETQ Mat (FASSOC NIL TestRes))
              (PutValue self testVar (CDR Mat)
                     (QUOTE Failed))
              (PutValue self testVar NIL (QUOTE Tested?)))
             ((SETQ Mat (FASSOC NotSetValue TestRes))
              (PutValue self testVar (CDR Mat)
                     (QUOTE Failed))
              (PutValue self testVar NotSetValue (QUOTE Tested?)))
             (T (PutValue self testVar T (QUOTE Tested?]
          (RETURN (GetValue self testVar (QUOTE Tested?])
```

(EQACTVAL

```
  [LAMBDA (ActVal Flist)                                       (* sm: "29-NOV-82 17:08")
                                                               (* checks if ActVal is an active value and has fields as given by
                                                               Flist)
                                                               (* currently works only for singly nested active value)
    (AND (EQ (CAR Flist)
             (fetch localState of ActVal))
         (EQ (CADR Flist)
             (fetch getFn of ActVal))
         (EQ (CADDR Flist)
             (fetch putFn of ActVal])
```

(EditOtherLinksMenu

```
  [LAMBDA NIL                                                  (* sm: "25-NOV-82 14:02")
                                                               (* gives a popup menu to select other Test Links)
    (MENU (create MENU
                  ITEMS _ [QUOTE ((AlternateTestOf (QUOTE ATO))
                                  (SubTestOf (QUOTE SUBO))
                                  (UsedBy (QUOTE UB))
                                  (ClassPreTest (QUOTE CPT]
                  TITLE _ "Other Test Links"])
```

(EditPreTest

```
  [LAMBDA (self link)                                          (* sm: "20-SEP-83 16:26")
                                                               (* allows user to edit a dependency list via a menu)
    (PROG (INP INP2 (Redo (QUOTE LOOP))
```

```
            Rval)
         (COND
            ((NULL link)
             (RETURN NIL)))
         (printout TTY "Editing" -2 link "::" -2 (GetValue self link)
               T)                                          (* if linkval is NIL, offer to add via Add menu)
         [COND
            ((NULL (GetValue self link))
             (SETQ INP (MENU (create MENU
                             ITEMS _ (QUOTE ((Add (QUOTE Add)
                                                  "object name or list of names to be added along this
                                                  link")
                                             (AddNum (QUOTE AddNum)
                                                    "enter a number for how many objects of this type
                                                    are to be added")
                                             (DefAdd (QUOTE DefAdd)
                                                    "ClassName or (ClsName InsName)  to CREATE and then
                                                    add an object")))
                             TITLE _ "Add Options")))
             (COND
                (INP (SETQ Redo (QUOTE LOOP2]
    LOOP
         (COND
            ((EQ Redo (QUOTE RETURN))
             (RETURN Rval))
            ((OR (NULL Redo)
                 (EQ Redo (QUOTE LOOP)))
             (PRIN1 "LinkEdCmd:")
             (SETQ INP (READ)))
            ((EQ Redo (QUOTE LOOP2))                       (* for looping without reading again)
             ))
         (SETQ Redo (QUOTE RETURN))
         [ERSETQ (SELECTQ INP
                      (Add (PRIN1 "ADD:")
                           (SETQ INP2 (READ))
                           (for x in (MKLIST INP2) do (PutValue self link x)))
                      (AddNum [COND
                                 ((FMEMB link (QUOTE (PreTest UsesObj)))
                                  (printout TTY "Not a valid way to add to" -2 link -2 "Use Add commd" T)
                                  (SETQ Redo (QUOTE LOOP)))
                                 (T (PRIN1 "Number:")
                                    (SETQ INP2 (READ))
                                    (COND
                                       ((NOT (NUMBERP INP2))
                                        (printout TTY "Enter a number!!" T)
                                        (SETQ Redo (QUOTE LOOP2)))
                                       (T [RPTQ INP2 (PutValue self link (GetValue (SEND (FindObjForLink link
                                                                                                      self)
                                                                                         New)
                                                                                   (QUOTE name]
                                                  (SETQ Redo (QUOTE LOOP])
                      (Delete (SETQ INP2 (ReadLinkMenu self link NIL T))
                              (for x in (MKLIST INP2) do (RemoveValue self link x))
                              (SETQ Redo (QUOTE LOOP)))
                      (Desc (DescribeTestLink self link)
                            (SETQ Redo (QUOTE LOOP)))
                      (DefAdd (PRIN1 "Class or (Class Inst)::")
                              (SETQ INP2 (MKLIST (READ)))
                              (COND
                                 [(type? class (GetObjectRec (CAR INP2)))
                                  (PutValue self link (GetValue (SEND (GetObjectRec (CAR INP2))
                                                                      New
                                                                      (COND
                                                                         ((CDR INP2)
                                                                          (CADR INP2))
                                                                         (T NIL)))
                                                                (QUOTE name]
                                 (T (printout TTY (CAR INP2)
                                              -2 "Not a class. Redo" T)))
                              (SETQ Redo (QUOTE LOOP)))
                      ((QUIT Quit Q))
                      (Modify (PRIN1 "NewList:")
                              (SETQ INP2 (READ))
                              (for x in (GetValue self link) do (RemoveValue self link x))
                              (for y in (MKLIST INP2) do (PutValue self link y))
                              (SETQ Redo (QUOTE LOOP)))
                      ((PP! PP)
                           (SETQ INP2 (ReadLinkMenu self link T T))
                           (for x in INP2 do (DoMethod (GetObjectRec x)
                                                       INP))
                           (SETQ Redo (QUOTE LOOP)))
                      ((Edit EDIT VEDIT Dedit EditIV EditCV)
                           (SETQ INP2 (ReadLinkMenu self link T T))
                           (for x in INP2 do (_ (GetObjectRec x)
                                                Edit
                                                (SELECTQ INP
                                                     (VEDIT (QUOTE (EE)))
```

```
                                                (Dedit (QUOTE (de)))
                                                ((Edit EDIT)
                                                    NIL)
                                               NIL))))
                        (List (printout TTY (GetValue self link)
                                      T)
                              (SETQ Redo (QUOTE LOOP)))
                        (Select (SETQ INP2 (ReadLinkMenu self link T T))
                                (SETQ Rval INP2))
                        (PROGN (COND
                                  ((NULL INP))
                                  (T (printout TTY "Illegal command" -3 INP -3 "Retry" T)))
                               (SETQ Redo (QUOTE LOOP]
          (GO LOOP])
```

(EditTestOtherCmds

```
  [LAMBDA NIL                                                    (* sm: "24-NOV-82 18:33")
                                                                 (* creates a popup menu for less often used commands for
                                                                 EditTEST cmds)
    (MENU (create MENU
                  ITEMS _ (QUOTE ((Edit (QUOTE EDIT)
                                        "edit using the lisp editor")
                                  (EditIVs (QUOTE EDITIV)
                                        "edit IVs only using TTYEdit")
                                  (EditCVs (QUOTE EDITCV)
                                        "edit CVs only using TTYEdit")
                                  (PP! (QUOTE PP!)
                                       "PP! of test object")
                                  (ResetPre (QUOTE RESETPRE)
                                        "reset test object and its pretests")
                                  (ResetSub (QUOTE RESETSUB)
                                        "reset test object and its subtests, syntax test etc")
                                  (TestSub (QUOTE TESTSUB)
                                        "test current object and all its subtests, syntax test")))
                  TITLE _ "Other Edit Cmds"])
```

(ExecTestFields

```
  [LAMBDA (self)                                                 (* sm: "20-SEP-83 16:27")
                                                                 (* executes one of the Test IVs -
                                                                 selected from a menu)
    (PROG (INP RES)
          (SETQ INP (MENU (create MENU
                                  ITEMS _ (QUOTE (SetUp TestExpr ResetExp))
                                  TITLE _ "Test Fields")))
          (COND
             (INP (printout TTY "Executing.." INP -2 .PPF (GetValue self INP)
                          T
                          (PROGN (SETQ RES (ERRORSET (GetValue self INP)
                                                     T))
                                 (COND
                                    (RES (CAR RES))
                                    (T "Error..")))
                          T)))
          (RETURN self])
```

(FindObjForLink

```
  [LAMBDA (link self)                                            (* sm: "18-OCT-82 10:51")

          (* returns the object corresp to the class that best determines the values of a particular dependency link)

    [COND
       ((NULL self)
        (SETQ self ($ LOOPSTestObject]
    (SELECTQ link
         (PreTest (Class self))
         (CasesUsed ($ LOOPSTestCases))
         (SubTest ($ LOOPSTestPrimitive))
         (SyntaxTest ($ LOOPSTestSyntax))
         (UsesObj ($ LOOPSTestSuper))
         (AltTest (Class self])
```

(GIVGetFn

```
  [LAMBDA (self varName localSt propName activeVal type)         (* sm: "29-MAR-83 16:43")
                                                                 (* GetFn used in LTFGetInitialValue tests.
                                                                 Attached to IVs GIV1 and GIV2 in LOOPSTestClass1)

    (ADD1 localSt])
```

(GetFromActVal

```
  [LAMBDA (self varName localSt propName activeVal type)         (* sm: "25-OCT-82 15:22")
                                                                 (* This is a getFn for testing Active Values)
                                                                 (* It returns the local state, going down to embedded Act Val if
                                                                 necessary)
```

```
    (GetLocalState activeVal self varName propName])
```

(LinkEditOtherMenu

```
  [LAMBDA NIL                                                    (* sm: "29-MAR-83 16:08")
                                                                 (* popup menu for other commands for editing Test Link objects)
    (MENU (create MENU
                  ITEMS _ (QUOTE (PP! (Edit (QUOTE Edit)
                                            "edit using Lisp editor")
                                      (EditIVs (QUOTE EditIV)
                                            "edit IVs only using TTYEdit")
                                      (EditCVs (QUOTE EditCV)
                                            "edit CVs only using TTYEdit")
                                      (Modify (QUOTE Modify)
                                            "replace current values by new list")))
                  TITLE _ "Other Cmds"])
```

(MakeBackLink

```
  [LAMBDA (self varName newValue propName activeVal type)        (* sm: "20-SEP-83 16:32")
```

(* This is a putFn for maintaining bi-links between TestCases and TestObject instances but could be used generally too.
Uses BackLink prop to find name of back link. Should NOT be invoked with other than PutValue.)

(* SPL CASES: (a) if newValue is of the form (-
v1 ..vn), then removes v1 to vn)

(* (b) if newValue is of the form (v1 ..vn), then adds only those vi which are already not there)

```
    (PROG [(blink (GetValue self varName (QUOTE BackLink]        (* if newValue is atom, make it a list)
          [COND
             ((ATOM newValue)
              (SETQ newValue (CONS newValue]
          [COND
             [(EQ (CAR newValue)
                  (QUOTE -))
              (for x in (CDR newValue) do (COND
                                             [(FMEMB blink (SEND (GetObjectRec x)
                                                                 List!
                                                                 (QUOTE IVs)))
                                              (COND
                                                 ((FMEMB x (GetLocalState activeVal self varName propName))
                                                  (PutLocalState activeVal (DREMOVE x (GetLocalState activeVal
                                                                                                     self varName
                                                                                                     propName))
                                                         self varName propName type)
                                                  (PutValue (GetObjectRec x)
                                                         blink
                                                         (LIST (QUOTE -)
                                                               (@ name)))
                                                  (MARKASCHANGED (GetObjectName self)
                                                         (QUOTE INSTANCES))
                                                  (MARKASCHANGED x (QUOTE INSTANCES]
                                             (T (printout TTY blink -2 "Not valid link for " x -2 "Ignoring.." T]
             (T (for x in newValue do (COND
                                         [(FMEMB blink (SEND (GetObjectRec x)
                                                             List!
                                                             (QUOTE IVs)))
                                          (COND
                                             ((NOT (FMEMB x (GetLocalState activeVal self varName propName)))
                                              (PutLocalState activeVal (CONS x (GetLocalState activeVal self
                                                                                              varName propName))
                                                     self varName propName type)
                                              (PutValue (GetObjectRec x)
                                                     blink
                                                     (@ name))
                                              (MARKASCHANGED (GetObjectName self)
                                                     (QUOTE INSTANCES))
                                              (MARKASCHANGED x (QUOTE INSTANCES]
                                         (T (printout TTY blink -2 "Not valid link for " x -2 "Ignoring.." T]
          (RETURN (GetLocalState activeVal self varName propName])
```

(MakeSet

```
  [LAMBDA (lis)                                                  (* sm: "26-OCT-82 13:35")
                                                                 (* from a list removes duplicates from the back, and changes the
    list)                                                        (* RETURNs THE MODIFIED LIST)
    (PROG ((Ptr lis))
      LOOP
          (COND
             ((NULL Ptr)
              (RETURN lis)))
          [COND
             ((FMEMB (CAR Ptr)
                     (CDR Ptr))
              (RPLACD Ptr (DREMOVE (CAR Ptr)
                                   (CDR Ptr]
```

```
          (SETQ Ptr (CDR Ptr))
          (GO LOOP])
```

(ObjectName

```
  [LAMBDA (x)                                                    (* sm: "21-OCT-82 13:42")
                                                                 (* returns the name of x, where x may be an object or object
                                                                 name)

    (PROG (obj)
          (SETQ obj (GetObjectRec x))
          (RETURN (COND
                     ((type? instance obj)
                      (GetValue obj (QUOTE name)))
                     (T (ClassName obj])
```

(PreTestsSatisfied?

```
  [LAMBDA (self TestedLst)                                       (* sm: "20-SEP-83 16:33")
                                                                 (* checks the PreTests of a test and returns T, NIL or
                                                                 NotSetValue)

    (PROG NIL
          [COND
             ((NOT (EQ (CheckClassTest self TestedLst)
                       T))
              (PrintTestHeader self)
              (printout TTY 5 "Following PreTest for this class failed:" -4 (@@ self ClassPreTest Failed)
                        T)
              (RETURN (PerformAltTest self (QUOTE ClassPreTest)
                             NotSetValue TestedLst]              (* The class PreTest were successful so continue...)
          [COND
             ((NOT (ValueNonNIL? (DoLoopsTest self (QUOTE PreTest)
                                       (QUOTE TEST)
                                       NIL TestedLst)))
              (PrintTestHeader self)
              (printout TTY 5 "Following pretests failed:" -4 (@ self PreTest Failed)
                        T)
              (RETURN (PerformAltTest self (QUOTE PreTest)
                             NotSetValue TestedLst]              (* PreTests were successful)
          (RETURN T])
```

(PushClassValueNew

```
  [LAMBDA (self var val prop)                                    (* sm: " 5-OCT-82 12:29")
                                                                 (* does PushClassValue only if val is already not on the value
                                                                 list)

    (COND
       ((FMEMB val (GetClassValue self var prop)))
       (T (PushClassValue self var val prop)))
    val])
```

(PutInActVal

```
  [LAMBDA (self varName newValue propName activeVal type)        (* sm: "25-OCT-82 15:24")
                                                                 (* This is a putFn for testing active values)
                                                                 (* it just puts the value in)
    (PutLocalState activeVal newValue self varName propName type])
```

(ReadLinkMenu

```
  [LAMBDA (self link default AllOpt)                             (* sm: "21-OCT-82 14:21")

          (* asks the user to select one of the values of (self link)%. Returns the selected value or NIL.
          If default is T, then if only one value is there, returns that as the selection, WITHOUT asking asking the user)

          (* If AllOpt is T, then gives an ALL option, and returns a list of selected names, even if all was not chosen)

    (PROG ((vals (GetValue self link))
           Sel)
          (RETURN (COND
                     ((NULL vals)
                      NIL)
                     [(AND (EQ (LENGTH vals)
                               1)
                           default)
                      (COND
                         (AllOpt vals)
                         (T (CAR vals]
                     (T (SETQ Sel (MENU (create MENU
                                                ITEMS _ (COND
                                                           (AllOpt (CONS (QUOTE ALL)
                                                                         vals))
                                                           (T vals))
                                                TITLE _ "Current Values")))
                        (COND
                           ((EQ Sel (QUOTE ALL))
                            vals)
                           ((NULL Sel)
```

```
                          NIL)
                         (AllOpt (CONS Sel))
                         (T Sel])
```

(RemoveValue

```
[LAMBDA (self var val prop)                                    (* sm: "11-OCT-82 17:52")
```

(* removes a value from var. will work only with var having the active value putFns MakeBackLink or AllowRemove)

```
  (PutValue self var (CONS (QUOTE -)
                           (MKLIST val))
        prop])
```

(ResetLTKERCLASSES

```
[LAMBDA NIL                                                    (* sm: "30-MAR-83 11:18")
  (PutClassValue ($ LOOPSTestMethod)
         (QUOTE ClassTested?)
         (QUOTE U))
  (PutClassValue ($ LOOPSTestEnvironment)
         (QUOTE ClassTested?)
         (QUOTE U])
```

(ResetLTKERVARS

```
[LAMBDA NIL                                                    (* sm: " 5-Jun-84 11:46")
  [SETQ Failed (SETQ NotDone (SETQ HasTest (SETQ Tested NIL]
  (SETQ EditTestWindows NIL)
  (SETQ LTAskNoTestAvailable NIL)
  (SETQ LTMsgLev (SETQ LTELev 8])
```

(ResetPutLocalStateVars

```
[LAMBDA NIL                                                    (* sm: "29-NOV-82 18:07")
                                                               (* recreates the active values in LOOPSTestClass6 used by
                                                               LTFPutLocalState)
  (PutValueOnly ($ LOOPSTestClass6)
         (QUOTE PLS1)
         (create activeValue
                localState _ (QUOTE LTC6)
                getFn _ NIL
                putFn _ NIL))
  (PutValueOnly ($ LOOPSTestClass6)
         (QUOTE PLS3)
         (create activeValue
                localState _ (create activeValue
                                     localState _ (QUOTE LTC6)
                                     getFn _ (QUOTE GetFromActVal)
                                     putFn _ (QUOTE PutInActVal))
                getFn _ (QUOTE GetFromActVal)
                putFn _ (QUOTE PutInActVal])
```

(SetupEditTestObjMenu

```
[LAMBDA NIL                                                    (* sm: "29-MAR-83 16:10")
                                                               (* Sets up the menus for editing TestObjs)
                                                               (* returns the list of windows created)
  (LIST (TMenu (QUOTE ((PreTest (QUOTE PT)
                                "manipulate PreTest list" "
                                ")
                       (SubTest (QUOTE SUB)
                                "manipulate SubTest list" "
                                ")
                       (PreTestOf (QUOTE PTO)
                                "manipulate PreTestOf list" "
                                ")
                       (UsesObj (QUOTE UO)
                                "manipulate UsesObj list" "
                                ")
                       (CasesUsed (QUOTE CU)
                                "manipulate CasesUsed list" "
                                ")
                       (SyntaxTest (QUOTE ST)
                                "manipulate SyntaxTest list" "
                                ")
                       (AltTest (QUOTE AT)
                                "manipulate Alternate Test list" "
                                ")
                       (OtherLinks (EditOtherLinksMenu)
                                "allows selection of less often used links" "
                                )))
                     "Test Links"
                     (QUOTE (645 20 150 150)))
        (TMenu (QUOTE ((Dedit (QUOTE DEDIT)
                              "Edit using Dedit" "
                              ")
```

```
                        (PP (QUOTE PP)
                            "prettyprint this object" "
                            ")
                        (Quit (QUOTE QUIT)
                              "Quit this editing session" "
                              ")
                        (Eval (QUOTE EVAL)
                              "enter the USEREXEC" "
                              ")
                        (Test (QUOTE TEST)
                              "test the current object" "
                              ")
                        (ReTest (QUOTE RETEST)
                               "retest the object, resetting it if needed" "
                               ")
                        (WhoAmI (QUOTE WHO)
                               "tells you which object is being edited" "
                               ")
                        (Describe (QUOTE DESCRIBE)
                               "describes the test object and state of any tests that were run" "
                               ")
                        (Top (QUOTE TOP)
                             "go back to the very first object in this session" "
                             ")
                        (Unremember (QUOTE UNREMEMBER)
                               "select (and pop) the last remembered object and edit it next" "
                               ")
                        (Remember (QUOTE REMEMBER)
                               "save this object on the stack" "
                               ")
                        (Select (QUOTE SELECT)
                               "asks for the next object to be edited (remembering current)" "
                               ")
                        (Reset (QUOTE RESET)
                               "reset the test state of current object" "
                               ")
                        (Execute (QUOTE EXEC)
                               "evaluate one of the selected Test fields" "
                               ")
                        (TtyEdit (QUOTE VEDIT)
                               "Edit using TTYIN editor" "
                               ")
                        (Others (EditTestOtherCmds)
                               "allows selection of less common commands" "
                               )))
                "TestObj Edit Cmds"
                (QUOTE (470 20 170 150)))
        (TMenu (QUOTE ((Add NIL "enter the objects to be added" "
                              ")
                        (Delete NIL "select the object to be deleted" "
                              ")
                        (Select NIL "selected item will be edited next" "
                              ")
                        (Quit NIL "quit editing this link" "
                              ")
                        (PP NIL "prettyprints the selected item" "
                             ")
                        (Dedit NIL "edits selected object using DEDIT" "
                              ")
                        (Desc NIL "describe the link values" "
                              ")
                        (List NIL "displays the current list" "
                              ")
                        (AddNum NIL "enter a number for new objects to be created" "
                              ")
                        (DefAdd NIL "add a new object to be defined by name" "
                              ")
                        (TtyEdit (QUOTE VEDIT)
                               "edits selected object using screen editor" "
                               ")
                        (Others (LinkEditOtherMenu)
                               "other commands" "
                               )))
                "Link Edit Cmds"
                (QUOTE (800 20 120 150])
```

(TickleBrowserNodes

```
  [LAMBDA (Objs Browser)                                    (* sm: "15-NOV-82 10:26")
```

(* given Browser with initial objects "Objs" Flips or Flashes these and other objects reachable via the link (s) given in the browser)

```
    (PROG [(Done (CONS))
           (Nodes (CONS))
           Ptr Next (Links (MKLIST (GetValue Browser (QUOTE subLinks]
          (LCONC Nodes (APPEND Objs))
```

```
          (SETQ Ptr (CAR Nodes))
      LOOP
          (SETQ Next (GetObjectRec (CAR Ptr)))
          (COND
             ((MEMBER Next (CAR Done))
              (GO LOOP2)))
          [COND
             ((EQ (GetValue Next (QUOTE Tested?))
                  T)
              (ERSETQ (_ Browser FlipNode Next)))
             ((OR (EQ (GetValue Next (QUOTE Tested?))
                      NIL)
                  (EQ (GetValue Next (QUOTE SetUp)
                                (QUOTE Tested?))
                      NIL))
              (ERSETQ (_ Browser FlashNode Next 5]
          [for x in Links do (LCONC Nodes (APPEND (GetValue Next x]
          (TCONC Done Next)
      LOOP2
          (SETQ Ptr (CDR Ptr))
          [COND
             ((NULL Ptr)
              (RETURN (CAR Done]
          (GO LOOP])

)

(RPAQQ LTKERINSTANCES NIL)

(RPAQQ LTKERVARS (AllTest CMN DUP EditTestWindows Failed HasTest LTAskNoTestAvailable LTELev LTMsgLev NotDone
                          OMN Seed Tested))

(RPAQQ AllTest
       (#. ($& LOOPSTestEnvironment "NPR@@A ")
           #.
           ($& LOOPSTestEnvironment "NPR@@A ")
           #.
           ($& LOOPSTestEnvironment "NPR@@A ")
           #.
           ($& LOOPSTestEnvironment "NQR@@AI")
           #.
           ($& LOOPSTestEnvironment "NPR@@AS")
           #.
           ($& LOOPSTestMethod "ONR@@AÁÃ")
           #.
           ($& LOOPSTestMethod "ONR@@A ")
           #.
           ($& LOOPSTestMethod "ONR@@A:")
           #.
           ($& LOOPSTestMethod "ONR@@Az")
           #.
           ($& LOOPSTestMethod "ONR@@Ae")
           #.
           ($& LOOPSTestMethod "ONR@@Aq")
           #.
           ($& LOOPSTestMethod "ONR@@A@")
           #.
           ($& LOOPSTestMethod "ONR@@AG")
           #.
           ($& LOOPSTestMethod "ONR@@AE")
           #.
           ($& LOOPSTestMethod "ONR@@AK")
           #.
           ($& LOOPSTestMethod "ONR@@AZ")
           #.
           ($& LOOPSTestMethod "ONR@@AY")
           #.
           ($& LOOPSTestLispFunc "OKR@@E`j")
           #.
           ($& LOOPSTestLispFunc "OKR@@E`k")
           #.
           ($& LOOPSTestLispFunc "OKR@@E`h")
           #.
           ($& LOOPSTestLispFunc "OER@@E *")
           #.
           ($& LOOPSTestLispFunc "OER@@E +")
           #.
           ($& LOOPSTestLispFunc "OER@@E /")
           #.
           ($& LOOPSTestLispFunc "OER@@E 8")
           #.
           ($& LOOPSTestLispFunc "OER@@E >")
           #.
           ($& LOOPSTestLispFunc "OER@@E ?")
           #.
           ($& LOOPSTestLispFunc "OER@@E 2")
           #.
```

```
           ($& LOOPSTestLispFunc "OER@@E 0")
           #.
           ($& LOOPSTestLispFunc "OER@@E 6")
           #.
           ($& LOOPSTestLispFunc "OKR@@EÏ")
           #.
           ($& LOOPSTestLispFunc "OKR@@EÓ")
           #.
           ($& LOOPSTestLispFunc "OGR@@E ")
           #.
           ($& LOOPSTestLispFunc "OGR@@E ")
           #.
           ($& LOOPSTestLispFunc "M]SC@=MN")
           #.
           ($& LOOPSTestLispFunc "M]SC@=MG")
           #.
           ($& LOOPSTestLispFunc "O\R@@[`h")
           #.
           ($& LOOPSTestLispFunc "O\R@@[`i")
           #.
           ($& LOOPSTestLispFunc "O\R@@[!")
           #.
           ($& LOOPSTestLispFunc "DCR@@EÄ")
           #.
           ($& LOOPSTestLispFunc "DCR@@EÅ")
           #.
           ($& LOOPSTestBasic "NXR@@EŽ")
           #.
           ($& LOOPSTestBasic "OSR@@A-")
           #.
           ($& LOOPSTestBasic "OSR@@AÕ")))

(RPAQQ CMN (Add CommentMethods CopyMethod DefMethod DisplaySubClasses EditMethod FetchMethod HasCV Initialize
                MethodDoc MoveMethod New NewTemp NewWithValues OnFile PPM PPMethod ReplaceSupers SubClasses))

(RPAQQ DUP (Destroy Destroy! DumpFacts Edit HasIV InspectFetch InspectPropCommand InspectProperties
                InspectStore InspectTitle InspectTitleMenu InspectValueCommand List List! PP PP! PPV! Put
                Rename SetName TitleCommand))

(RPAQQ EditTestWindows NIL)

(RPAQQ Failed NIL)

(RPAQQ HasTest NIL)

(RPAQQ LTAskNoTestAvailable NIL)

(RPAQQ LTELev 8)

(RPAQQ LTMsgLev 8)

(RPAQQ NotDone NIL)

(RPAQQ OMN (AddIV AssocKB At BreakIt Class ClassName CopyDeep CopyShallow DeleteIV DoMethod IVMissing Inspect
                InstOf InstOf! Instantiate MessageNotUnderstood NoObjectForMsg PrintOn PutIVProp
                ReturnDefaultValue TraceIt UnSetName Understands WhereIs))

(RPAQQ Seed
       (#. ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")
           #.
           ($& LOOPSTestLispFunc "#(())")))
```

```
(RPAQQ Tested NIL)

(ResetLTKERCLASSES)

(PUTPROPS LTKER COPYRIGHT ("Xerox Corporation" 1984 1985 1986))
```

FUNCTION INDEX

AddAltTest 10
AllowRemove 11
AskPreTest 11
AskSubTest 11
AskSyntaxTest 12
AskTestCases 12
BuildPreTest 13
BuildPrimClassTest 13
CheckClassTest 13
CheckPreTest 14
CreateLTKBS1 14
DescribePreviousTry 15
DescribeTestLink 16
DisplayTestBrowser 16
DoLoopsTest 17
EditOtherLinksMenu 17
EditPreTest 17
EditTestOtherCmds 19
EQACTVAL 17
ExecTestFields 19
FindObjForLink 19
GetFromActVal 19
GIVGetFn 19
LinkEditOtherMenu 20
LOOPSTestEnvironment.TEST 4
LOOPSTestEnvironment.TestSelf 4
LOOPSTestKernel.TEST 4
LOOPSTestObject.DefineTEST 5
LOOPSTestObject.EditTEST 5
LOOPSTestObject.EditTestInTTYProcess 6
LOOPSTestObject.IgnoreTest 6
LOOPSTestObject.Reset! 7
LOOPSTestObject.ResetAll 7
LOOPSTestObject.ResetDep 7
LOOPSTestObject.ReTest 6
LOOPSTestObject.ReTestDep 7
LOOPSTestObject.TEST 8
LOOPSTestObject.TEST! 8
LOOPSTestObject.TESTall 9
LOOPSTestObject.TESTDep 9
LOOPSTestObject.TestSelf 9
LOOPSTestPrimitive.XTEST 10
MakeBackLink 20
MakeSet 20
ObjectName 21
PreTestsSatisfied? 21
PushClassValueNew 21
PutInActVal 21
ReadLinkMenu 21
RemoveValue 22
ResetLTKERCLASSES 22
ResetLTKERVARS 22
ResetPutLocalStateVars 22
SetupEditTestObjMenu 22
TickleBrowserNodes 23

VARIABLE INDEX

AllTest 24
CMN 25
DUP 25
EditTestWindows 25
Failed 25
HasTest 25
LTAskNoTestAvailable ...25
LTELev 25
LTKERINSTANCES 24
LTKERVARS 24
LTMsgLev 25
NotDone 25
OMN 25
Seed 25
Tested 26