Filed on: {QV}<LISPCOURSE>TOC-EXTEN-V2

LispCourse #23: What is Programming; Basic Lisp Revisited

What is Programming? (pp. 1-5)

Procedures and Data: The Semantics of Programming (pp. 1-3)

Procedure versus Process: Running a Program (pp. 3-4)

Correctness, Efficiency, Maintainability, & Adaptability Quickly and with Minimal Effort: The Pragmatics of Programming (pp. 4-5)

Correct

Efficient

Maintainable

Adaptable

Reviewing the Basics of Lisp Programming (p. 5-19)

The READ-EVAL-PRINT Loop (p. 5)

Basic Syntactic Building Blocks: Atoms and Lists (pp. 5-7)

Forms and the Lisp Interpreter (p. 7)

Special Forms & QUOTE (p. 8)

QUOTE (p. 8)

QUOTE is used to prevent evaluation where it is not required. (p. 8)

EVAL (pp. 8-9)

Lisp Control Structures (pp. 9-13)

COND (pp. 9-10)

Iterative control structures: The FOR Loop and its cousins (p. 10)

The FOR Loop (pp. 11-12)

(FOR *variable* IN *list* DO *operation*)

Alternative FOR loops (pp. 12-13)

DO versus COLLECT

"IN *list*" versus "FROM *n* TO *m* BY *k*"

WHILE and UNTIL loops: Alternatives to FOR (p. 13)

(WHILE *predicate* {DO, COLLECT} *operations*)

UNTIL

Defining and Applying Functions (pp. 13-16)

Defining functions (pp. 14-15)

(DEFINEQ *Defn1 Defn2* ...)

(*FunctionName FunctionDefn*).

(LAMBDA *ParameterList FunctionBody*)

Applying functions (pp. 15-16)

APPLY

Lisp Interpreter: A combination of EVAL and APPLY (p. 16)

Lists: the primary Lisp data strucuture (p. 17)

List Maniupulation Functions (pp. 17-18)

List Formats (p. 18-19)

ASSOC list

ASSOC

Prop list

LISTGET

LISTPUT

Exercise: Learning Data Abstraction (pp. 19-20)
References (p. 20)

LispCourse #24: Data Abstraction

Programs as Representations of the Real World (p. 1)

Data, Compound Data & Data Structures (pp. 1-2)

Data Abstraction: Dealing with Compound Data (pp. 2-8)

The Need for Data Abstraction (pp. 2-4)

Data Abstraction (p. 4)

Abstraction Barriers (pp. 5-6)

Constructing Access Functions: Constructors, Selectors & Mutators (pp. 6-7)

Constructors, Selectors and Mutators (pp. 7-8)

Example: Problem from LispCourse #23 (pp. 8-9)

References (p. 9)

Exercises (p. 9)

LispCourse #25: The Record Package; The Inspector

Records: Data Abstraction in Interlisp (pp. 1-16)

Records: Basic Stuff (pp. 1-2)

Record Declarations (pp. 2-3)

(RECORD *RecordName Fields ExtraStuff1 ExtraStuff2* ...)

Constructing Record Instances: The CREATE statement (pp. 3-4)

(CREATE *RecordName Assignment1 Assignment2* ...)

FieldName _ Form

Selecting Fields in a Record: The fetch statement (pp. 5-6)

fetch

(fetch (*RecordName FieldName*) of *Form*)

Mutating Fields in a Record: The replace statement (pp. 6-7)

(replace (*RecordName FieldName*) of *Form1* with *Form2*)

Records as Data Abstraction (pp. 7-8)

Hierarchical Data Structures (pp. 8-10)

Records constructed from different list structures (pp. 12-13)

TYPERECORD (p. 12)

ASSOCRECORD (p. 12)

PROPRECORD (p. 13)

Determining the record type of an object: The TYPE? statement (pp. 13-16)

(TYPE? *Form*) (p. 14)

(TYPE? *RecordName Form*) (pp. 14-16)

DATATYPES: Defining New Objects in the Interlisp System (pp. 16-19)

(DATATYPE Name (First Middle Last))

System DATATYPEs (p. 18-19)

Editing record declarations and record instance: EditRec and The Inspector (p. 19)

References (p. 19)

Exercises (p. 19)

LispCourse #26: Reprise; Continuation of Record Package from #25

Reprise: Sample Programming Exercise (pp 1-9)

The Problem (pp. 1-2)

Step 1: Describe the database (pp. 2-3)

Step 2: Write the constructors, selectors, and mutators for the data structures (pp. 3-6)

Step 3: Write the functions that do the work as required. (pp. 6-9)

Step 4: Save your work on a file and list it. (p. 9)

Step 5: Test, Debug, and redo the MAKEFILE. (p. 9)

Continuation from #25: Editing Record/Datatype Declarations and Instances (pp. 10-16)

Examining and Editing Record/Datatype Declarations: RECLOOK and EDITREC (pp. 10-11)

(RECLOOK *RecordName*).

(EDITREC *RecordName*)

Examining and Editing Record/Datatype Instances: The Inspector (pp. 11-15)

(INSPECT *Instance*)

Using the Inspector (pp. 15-16)

References (p. 16)

Exercise (p. 16)

(FOR *Variable* IN *List* WHEN *Predicate* DO *S-Exprs*)

LispCourse #27: Homework from #26; Completion of #25 & #26

LispCourse #28: Type Checking; Strings; Arrays

Type Checking in Interlisp (pp. 1-8)

The Concept of Type Checking (pp 1-3)

Adding type checking to your Interlisp functions (p. 4)

Type Checking Technique (pp. 4-6)

Type checking predicates (p. 7)

LITATOM, **NUMBERP**, and **LISTP**

TYPE?

Overview of the Types of Data in Interlisp (pp. 8-9)

Print Names (pp. 9-10)

Arrays (pp. 10-11)

Array manipulation functions (pp. 11-13)

Creating arrays (p. 11)

(ARRAY *Size*)

Array predicate (pp. 11-12)

(ARRAYP *Arg*)

Accessing the entries of an array (pp. 12-13)

(SETA *Array N Value*)

(ELT *Array N*)

Finding out the size of an array (p. 13)

(ARRAYSIZE *Array*)

Using arrays (pp. 13-14)

Strings (pp. 15-21)

String manipulation functions (pp. 15-19)

String predicate (pp. 15-16)

(STRINGP *Arg*)

Creating strings (pp. 16-17)

(MKSTRING *Arg*)

(ALLOCSTRING *N Character*

Comparing strings (p. 17)

(STREQUAL *Str1 Str2*)

Concatenating strings (pp. 17-18)

(CONCAT *Str1 Str2* ...)

Decomposing strings (p. 18)

(SUBSTRING *Str Start End*)

Searching strings (p. 18-19)

(STRPOS *Pattern String Start SkipChar*)

Using Strings (pp. 19-21)

References (p. 21)

LispCourse #29: Solutions for Homework #28

LispCourse #30: Conceptual Models for Atoms, Lists *et al.*

Introduction (pp. 1-2)

Everything in Lisp is a *Pointer* (p. 2)

Representing Atoms (pp. 3-4)

Representing Lists (pp. 4-20)

- **The CONS cell (pp. 4-5)**
- **Lists are built from CONS cells (pp. 5-8)**
- **CAR and CDR revisited (p. 9)**
  - **CAR**
  - **CDR**
- **CONSing onto a list and LIST revisited (pp. 9-11)**
- **APPEND revisited (pp. 11-12)**
- **The COPY function (pp. 12-13)**
- **RPLACA, RPLACD, and NCONC -- The destructive functions (pp. 14-17)**
  - **(RPLACA *X Y*) (pp. 14-15)**
  - **(RPLACD *X Y*) (pp. 15-16)**
  - **(NCONC *X1 X2 X3* ...) (p. 17)**
- **Some dangers in using the destructive functions (pp. 18-20)**

Representing Datatypes and Arrays (pp. 21 -22)

Representing Strings (pp. 22-25)

- ***string pointer*** and a ***character vector***.
  - **GNC (pp. 23-24)**

GLC (pp. 24-25)

(RPLSTRING *String1 N String2*) (pp. 25-26)

Sameness and Equality in Lisp: EQ versus EQUAL (pp. 27-30)

EQ (p. 28)

EQUAL (p. 29)

References (p. 30)

Exercises (attachment)

LispCourse #31: Solutions for Homework #30, Problems 2 and 3

LispCourse #32: Homework on circular queues

Exercise

Overall problem (p. 1)

Defintions (p. 1)

Subproblems (p. 2-3)

LispCourse #33: Solutions for Homework #32

LispCourse #34: Variable Binding and the Interlisp Stack

Variable Reference Inside Functions: Bound and Free Variables (pp. 1-4)

Review: Evaluating S-expressions using EVAL and APPLY (pp. 4-8)

EVAL (p. 5)

APPLY (p. 6)

EXAMPLE (from LispCourse #3, page 6) (pp. 7-8)

Missing Pieces (p. 8)

The Lisp Stack, Variable Binding, Variable Lookup, and Related Topics (pp.

The Lisp Stack (pp. 8-10)

Variable Binding (pp. 10-11)

Looking up the Value of a Variable (pp. 11-13)

Setting the Value of a Variable (pp. 13-16)

Function Definition Lookup (pp. 16-17)

Variable Reference in Practice: LET and PROG are used to bind variables

Avoid Free Variables! (pp. 17-19)

The LET Special Form (pp. 19-23)

The PROG Special Form (pp. 23-25)

FOR and WHILE are implemented using PROG (pp. 25-27)

Global Variables: Free variables used as system or package parameters (pp. 28-29)

References (pp. 29-30)

Exercise (pp. 30-31)

LispCourse #35: Solutions to Homework #34

Apologies (pp. 1-2)

Solutions (attachment)

Sample Runs (attachment)

LispCourse #36: More on Variable Binding; Control Structures in Lisp

NLAMBDA Functions (pp. 1-3)

Spread and NoSpread Functions (pp. 3-8)

NLAMBDA nospread functions (p. 4)

LAMBDA nospread functions (p. 5-6)

What are nospread functions good for? (pp. 6-7)

Summary of 4 function types (pp. 7-8)

Functions as Arguments in Function Calls (pp. 8-10)

FUNCTION (p. 10)

A more realistic example (p. 10)

Control Structures in Lisp (pp. 11-

Procedural Abstraction (pp. 11-12)

Interlisp Control Structures (pp. 12-

Review: Control Structures Already Covered (pp. 13-

Sequential evaluation (p. 13)

Embedding (pp. 13-14)

COND (pp. 14-15)

AND, OR (pp. 16-18)

Iteration: FOR, WHILE, and UNTIL (pp. 18-19)

LET (p. 20)

PROG with RETURN (pp. 20-21)

PROG with GO (p. 22)

Other Control Structures

SELECTQ (p. 22-24)

EFS (p. 25)

PROG1, PROG2, PROGN (pp.25-27)

Simple Repetition: RPT and RPTQ (27-28)

Mapping functions: MAP, MAPC, MAPLIST, MAPCAR (pp. 28-30)

(MAPC *List WorkFn NextFn*) (p. 29)

(MAPCAR *List WorkFn NextFn*) (pp. 29-30)

Recursion -- The ultimate control structure in Lisp (p. 30)

References (pp. 30-31)

LispCourse #37: Recursion; Organizing Large Programs

Rercursion ÿ The ultimate Lisp control structure (pp. 1-3)

Types of Recursion (pp. 4-7)

Single Recursion (and tail recursion) (pp. 4-6)

Double (or more) Recursion (pp. 6-7)

Iteration versus Recursion (pp. 7-10)

Organizing Large Programs by Object and Operation (pp. 7-18)

Organization by Operation (pp. 11-12)

Organization by Type of Object (pp. 12-14)

Organization using the table: Data-directed programming (pp. 14-15)

Organization by Object Instance: Object-oriented programming (pp. 15-18)

Choosing from among these organizations (p. 18)

References (p. 18)

LispCourse #38: Files; Streams; Input/Output

Files (pp. 1-3)

Files From the Programmer's Point of View (p. 1)

File Names (pp. 1-3)

(FILENAMEFIELD *FileName FieldName*) (p. 2)

(UNPACKFILENAME *FileName*) (p. 2)

(PACKFILENAME *FieldName1 FieldContents1 ... FieldNameN FieldContentsN*) (p. 2)

Streams (pp. 3-4)

Low-level Access to Files Using Streams (p. 4-10)

Opening and Closing (pp. 4-7)

(OPENSTREAM *FileName Access Recognition*) (p. 4)

(OPENP *FileNameOrStream Access*) (pp. 5-6-

(CLOSEF? *Stream*) (pp. 6-7)

(CLOSEALL) (p. 7)

File Pointers (pp. 7-9)

(GETFILEPTR *Stream*) (p. 7)

(GETEOFPTR *Stream*) (p. 8)

(RANDACCESSP Stream) (p. 8)

(SETFILEPTR *Stream Position*) (p. 8)

(FILEPOS *Pattern Stream Start End Skip*) (p. 8)

Reading and Writing (pp. 9-10)

(BIN *Stream*) (p. 9)

(BOUT *Stream Byte*) (p. 9)

Example (pp. 9-10)

Higher-level File Input and Output (pp. 10-18)

Higher-level Input Functions (p. 10)

(READ *Stream*) (pp. 10-11)

(SKREAD *Stream*) (p. 11)

(READC *Stream*) (p. 11)

(RATOM *Stream*) (pp. 11-12)

(RSTRING *Stream*) (pp. 12)

(READFILE *StreamOrFileName*) (p. 12)

Higher-level Output Functions (pp. 11-18)

General Printing

(PRIN1 *SExpression Stream*) (pp. 13-14)

(PRIN2 *SExpression Stream*) (p. 14)

(TERPRI *Stream*) (p. 14)

(SPACES *N Stream*) (p. 14)

(PRINT *SExpression Stream*) (p. 14)

Printing Numbers (pp. 14-16)

(PRINTNUM *Format Number Stream*) (pp. 14-16)

(FIX *Width Base Pad0Flg LeftFlushFlg*) (pp. 14-15)

(FLOAT *Width DecWidth ExpWidth Pad0Flg*) (pp. 15-16)

PRINTOUT (pp. 16-18)

(PRINTOUT *Stream Command1 Command2* ...) (pp. 16-18)

Miscellaneous Considerations (pp. 18-20)

The File T (pp. 18-19)

Line Buffering When Reading Keyboard Input (p. 19)

(CONTROL T) (p. 19)

(CONTROL NIL) (p. 19)

Default Input/Output File (p. 20)

(INPUT *Stream*) (p. 20)

(OUTPUT *Stream*) (p. 20)

References (p. 20)

LispCourse #39: Solutions to Homework #38

Solution functions are attached. (16 pages)

Notes on Problem C (pp. 1-2)

LispCourse #40: Using the Display: Bitmaps, DisplayStreams, & Windows

Introduction (pp. 1-3)

***bitmap* (p. 1)**

Display streams **(p. 1)**

Windows **(p. 1)**

Coordinate Systems, Positions, & Regions (pp. 3-5)

Coordinate Systems for Specifying Locations in Bitmaps, et al. (pp. 3-4)

Positions and Regions (pp. 4-5)

POSITION **(create POSITION XCOORD _ *X* YCOORD _ *Y*) (p. 4)**

REGION **(CREATEREGION *Left Bottom Width Height*) (p. 4)**

(INSIDEP *Region Position*) (p. 4)

(INTERSECTREGIONS *Region1 Region2 ... RegionN*) (p. 5)

(UNIONREGIONS *Region1 Region2 ... RegionN*) (p. 5)

BITMAPs (pp. 5-12)

Introduction (p. 5)

Creating Bitmaps (p. 6)

(BITMAPCREATE *Width Height*)

BITBLT (pp. 6-10)

(BITBLT *SourceBitMap SourceLeft SourceBottom DestBitMap DestLeft DestBottom Width Height SourceType Operation Texture*) (pp. 6-7)

Examples (pp. 8-10)

Miscellaneous Bitmap Functions (p. 10)

(BITMAPBIT *BitMap X Y NewValue*)

(BITMAPCOPY *BitMap*)

(BITMAPHEIGHT *BitMap*)

(BITMAPWIDTH *BitMap*)

(SCREENBITMAP)

(CURSORBITMAP)

Hand Editing Bitmaps (pp. 11-12)

(EDITBM *BitMap*) (pp. 11-12)

Display Streams (pp. 12-18)

Display Stream Properties (& creating a display stream) (pp. 12-14)

Destination (p. 12)

XOffset, YOffset (p. 12)

ClippingRegion (p. 13)

XPosition, YPosition (p. 13)

Texture (p. 13)

Font (p. 13)

Operation (p. 13)

(DSPCREATE *Destination*) (pp. 13-14)

Moving the Current Position in the Display Stream (pp. 14-15)

(MOVETO *X Y DisplayStream*)

(RELMOVETO *DeltaX DeltaY DisplayStream*)

Printing and Drawing on Display Streams (pp. 15-18)

(DRAWTO *X Y Width Operation DisplayStream*) (p. 15)

(RELDRAWTO *DeltaX DeltaY Width Operation DisplayStream*) (p. 15)

(DRAWBETWEEN *Position1 Position2 Width Operation DisplayStream*) (p. 15-16)

(DRAWCURVE *Knots ClosedFlg Brush Dashing DisplayStream*) (pp. 16-17)

(DRAWCIRCLE *X Y Radius Brush Dashing DisplayStream*) (p. 18)

(DRAWELLISPSE *X Y MinorRadius MajorRadius Orientation Brush Dashing DisplayStream*) (p. 18)

Windows & the Window Package (pp. 18-36)

Space Management: The Window Stack (pp. 18-20)

Windows (pp. 20-21)

Open and Closed Windows (p. 20)

(OPENW *Window*)

(CLOSEW *Window*)

(OPENWP *Window*)

Creating Windows (pp. 20-21)

(CREATEW *Region Title Border NoOpenFlg*) (pp. 20-21)

(WINDOWP *Window*) (p. 21)

Windows and Display Streams (pp. 21-22)

(WINDOWPROP *Window* ’DSP) (pp. 21-22)

Printing and Drawing in Windows (p. 22)

Doing Things to Windows (pp. 22-23)

(MOVEW *Window Position*) (p. 22)

(SHAPEW Window NewRegion) (p. 22)

(TOTOPW *Window*) (p. 23)

(BURYW *Window*) (p. 23)

(CLEARW *Window*) (p. 23)

(REDISPLAYW *Window*) (p. 23)

(SHRINKW *Window*) (p. 23)

(EXPANDW *Window*) (p. 23)

Tailoring Windows to Special Applications: Window Properties (pp. 24-28)

Looks Properties (pp. 24-25)

TITLE (p. 24)

BORDER (p. 24)

WINDOWTITLESHADE (p. 24-25)

Read-only Properties (p. 25)

DSP (p. 25)

HEIGHT, WIDTH (p. 25)

REGION (p. 25)

Processes Property (p. 25)

PROCESS

Functions to Be Invoked by Operations on the Window (pp. 25-26)

CLOSEFN (p. 25)

OPENFN (p. 26)

TOTOPFN (p. 26)

MOVEFN (p. 26)

AFTERMOVEFN (p. 26)

REPAINTFN (p. 26)

Functions to Be Invoked by Mouse Events in the Window (pp. 27-28)

BUTTONEVENTFN (p. 27)

RIGHTBUTTONFN (p. 27)

WINDOWENTRYFN (p. 27-28)

CURSORINFN, CURSOROUTFN, CURSORMOVEDFN (p. 28)

Properties that Support Icons and Shrinking (p. 28)

SHRINKFN

ICONFN

EXPANDFN

Manipulating a Window's Properties: WINDOWPROP, et al. (pp. 28-30)

(WINDOWPROP *Window Property NewValue*) (p. 29)

(WINDOWADDPROP *Window Property ValueToBeAdded*) (p. 29)

(WINDOWDELPROP *Window Property ValueToBeRemoved*) (p. 30)

Making Scrollable Windows (pp. 30-36)

The SCROLLFN Property (pp. 30-31)

SCROLLBYREPAINTFN (pp. 31-36)

Menus (pp. 36-45)

Creating the menu data structure (pp. 36-39)

MENU

Fields that describe the menu's behavior (pp. 36-38)

ITEMS (pp. 36-37)

WHENSELECTEDFN (p. 37)

WHENHELDFN (p. 37)

WHENUNHELDFN (p. 38)

Fields that describe the menu's looks (p. 38)

TITLE (p. 38)

MENUFONT (p. 38)

MENUROWS *or* MENUCOLUMNS (p. 38)

ITEMHEIGHT, ITEMWIDTH (p. 38)

CENTERFLG (p. 38)

MENUBORDERSIZE (p. 38)

MENUOUTLINESIZE (p. 39)

Fields that describe the menu's positioning (p. 39)

MENUPOSITION

MENUOFFSET

Bringing up a menu on the screen: Pop-up and Fixed Menus (pp. 39-42)

(MENU *Menu Position*) (pp. 40-41)

(ADDMENU *Menu Window Position*) (pp. 41-42)

(DELETEMENU *Menu CloseFlg Window*) (p. 42)

Hierarchical Menus (pp. 42-44)

Menu Example (pp. 44-45)

Using the Mouse (pp. 45-50)

Writing BUTTONEVENTFNs, CURSORxxxFNs, and WHENSELECTEDFNs (pp. 45-47)

Getting Information about the Mouse and the Cursor (pp. 47-50)

Position of the Cursor (pp. 47-48)

(CURSORPOSITION *NewPosition Window*) (p. 47)

(LASTMOUSEX *Window*), (LASTMOUSEY *Window*) (p. 48)

LASTMOUSEX, LASTMOUSEY (p. 48)

(GETMOUSESTATE) (p. 48)

State of the Mouse Buttons (pp. 48-49)

(MOUSESTATE *ButtonSpec*) (pp. 48-49

(LASTMOUSESTATE *ButtonSpec*) (p. 49)

(UNTILMOUSESTATE *ButtonSpec Interval*) (p. 49)

Example of Using the Mouse (pp. 49-50)

References (p. 50)

LispCourse #41: The Compiler and MASTERSCOPE

The Interlisp Compiler

What is the Interlisp Compiler? (pp. 1-3)

An example of what the compiler does (pp. 3-5)

Using the compiler (pp. 5-8)

Compiler Questions (pp. 5-8)

Listing? (p. 5)

REDFINE? (p. 6)

SAVE EXPRS? (p. 6)

OUTPUT FILE? (p. 7)

Functions that invoke the compiler (p. 8-10)

(COMPILE *Functions*) (p. 8)

(TCOMPL *Files*) (p. 8)

(RECOMPILE *File*) (pp. 8-9)

Example (pp. 9-10)

Special Considerations when Writing Code to be Compiled (p. 10)

GLOBALVARS (pp. 11-12)

Macros (pp. 12-16)

Using Macros (pp. 13-14)

Defining macros and macro expansion (pp. 14-16)

(*List SExpression*) (p. 14)

(*LitAtom SExpression*) (p. 14)

(LAMBDA *ParamList FunctionDefinition*) (pp. 15-16)

The bottom line on Interlisp macros (p. 16)

MASTERSCOPE (pp. 16-19)

(. ANALYZE ALL ON *File*) (pp. 16-17)

(. SHOW PATHS FROM *Function*) (p. 17)

(. SHOW PATHS TO *Function*) (p. 17)

(. WHO CALLS *Function*) (p. 17)

(. WHO USES *Variable*) (p. 18)

(. WHO USES AS A RECORD *RecordName*) (p. 18)

(. EDIT WHERE ANY CALLS *Function*) or **(. EDIT WHERE ANY USES *Variable*)** or **(. EDIT WHERE ANY USES AS A RECORD *RecordName* (p. 18)**

References (p. 19)