CS 234319: Programming Languages
Joseph (Yossi) Gil*
April 1, 2015

1 Preliminaries

Contents [35 frames]
1.1 Administration [11 frames] .................. 1
1.2 Motivation [13 frames] ...................... 2
1.3 Hello, World! [11 frames] ................. 4
1.4 Thirteen weeks schedule .................. 5
List of equations ................................ 7
List of tables ................................... 7

1.1 Administration

11 Frames: □ Instructors □ Teaching assistants □ Course material
□ Resources managed by students □ Text books □ Bibliography
for tutorials □ Regulations □ Grade components □ Final grade
□ Syllabus □ At the end of the course you’ll know

1. Prof. Yossi Gil:
   - Office hours:
     http://yogi.youcanbook.me
   - Mail:
     mailto:yogi@cs.technion.ac.il

2. Dr. Sara Porat:
   - Office hours:
     Wednesday 14:30–15:30; Taub 717
   - Mail:
     mailto:porat@cs.technion.ac.il

2. Teaching assistants

3. Mr. Matan Peled:
   - Mail:
     mailto:mip@cs.technion.ac.il

4. Mr. Jenya Moroshko:
   - Mail:
     mailto:mjenya@cs.technion.ac.il

1. Instructors
2. Teaching assistants
3. Course material
4. Resources managed by students
5. Text books

Official web site
http://webcourse.cs.technion.ac.il/234319

where you will find:
- Printouts of slides in a variety of formats
- A couple of Hebrew monographs
- Some lecture notes taken by students
- Past exams
And the usual: grades, current assignment information & FAQs, periodical announcements,

Facebook group
https://www.facebook.com/groups/programming.languages/

Indexed Q&A site
https://safot.cs.technion.ac.il/

Main text book

But also English Wikipedia¹ and Google search. For further, in depth, reading:
“Programming Languages: Concepts and Constructs” (2nd Ed), Ravi Sethi. Addison-Wesley, 1996.

¹Hebrew Wikipedia is essentially rubbish
6. Bibliography for tutorials

Programming languages taught: PASCAL, ML, PROLOG

- “PROLOG Programming for Artificial Intelligence”, by Ivan Bratko. Addison-Wesley.

7. Regulations

Highlights

Midterm exam none
Prerequisites enforced
Co-requisites enforced
Homework grades crucial
Old homework grades cannot be transferred

Appeals
- considered favorably
- treated seriously
- must be in writing
- must be signed by student
- no “grades negotiation” discussing appeals or grades with staff voids all appeal rights

8. Grade components

Assignments (grade denoted by A, 0 ≤ A ≤ 100)
- Every 2–3 weeks
- Mandatory
- Typically includes both programming and mini-research problems
- Teams of two students each (strict!)
- Matching services provided by teaching assistants

Exam (grade denoted by E, 0 ≤ E ≤ 100)
- Guaranteed to include at least one homework assignment
- Guaranteed to include at least one past exam question
- Typically includes 8–12 questions

9. Final grade

Denoted by F, 0 ≤ F ≤ 100:

\[ F = \begin{cases} \text{round}(E) & \text{if } E - 1.5 \leq A \leq E \\ \text{round}(E) & \text{if } E \leq 50 \\ \text{round}(E + \sqrt{A - E}) & \text{if } 50 < E \leq A \\ \text{round}(E - \sqrt{E - A}) & \text{otherwise} \end{cases} \]  

10. Syllabus

- Concepts
  - values, types and expressions
  - typing systems
  - storage
  - commands
  - sequencers
  - function abstractions

- Programming paradigms
  - Imperative C, C++, PASCAL, JAVA, AWK, GO,
  - Functional ML, HASKELL, LISP
  - Declarative PROLOG

11. At the end of the course you’ll know

- What distinguishes different PLs from one another
- A variety of mechanisms in familiar and less familiar PLs
- Programming in the functional language ML
- Some basic concepts from PASCAL, PROLOG and other PLs

- Main skills:
  - Quickly learn a new PL
  - Evaluate PLs
  - Use any PL more cleverly
  - Search in Google

1.2 Motivation

13 Frames: Concrete reasons to study PLs? Discovering you speak prose New modes of thought But also many practical benefits Or, at least, understand the terminology/jargon of the trade Not for the faint of heart Possible approaches Problem... Who needs PLs? What is a PL? Language processors Relations to other fields in computer science Closely related topics
12. Concrete reasons to study PLs?

Fun
- ML is neat
- PROLOG is elegant
- There is more to it than C, C++, JAVA
- Enhance thinking flexibility

Professional skills
- Over 2,000 different languages out there
- Common concepts and shared paradigms
- Framework for comparative study of PLs

Useful for other courses:
- OOP
- Compilation
- Software Engineering
- Semantics of PLs
- Memory Management

13. Discovering you speak prose

«Par ma foi ! il y a plus de quarante ans que je dis de la prose sans que j’en susse rien, et je vous suis le plus obligé du monde de m’avoir appris cela.»

- The pleasure to discover that you have been speaking prose all your life without knowing it;
- or, more generally, learning, something new about old things.
- So, yes, the course will be telling you new stuff about old stuff...
- and we will practice some new modes of thought

14. New modes of thought

New
- Programming languages mechanisms much beyond the if and while
- Programming techniques
- Paradigms of thought
- Directions for your minds

And also,
- Get ready to Object Oriented Programming and other advanced courses.²
- Hone web-search skills.³

²the instructor of the Object Oriented Programming course paid me to mention this
³Google paid me to include this in the topic of our course

15. But also many practical benefits

Main objective
- learn, understand, and evaluate any new programming language

16. Or, at least, understand the terminology/jargon of the trade:

What kind of a beast is JavaScript?
- Imperative,
- With prototypes (object-based, but not object-oriented),
- Functions are first-class entities,
- Has lambda functions,
- With closures,
- Is weakly typed,
- Has dynamic typing,
- Has static scoping,
- … and a must-know for any modern website developer!

By the end of these course, many of these terms will be covered in depth.

17. Not for the faint of heart

- No mathematical interest. This is not yet another technical course:
  - Many “soft” definitions
  - Much reliance on common sense
  - No theorems, proofs, lemmas, or integration by parts
  - No easy grades for mathematical geniuses
- No “computational” interest. The expressive power of all programming mechanisms and computational devices is basically the same
- No “algorithmic” interest. You don’t discover new fascinating algorithms using better programming languages.

18. Possible approaches

- Define and compare paradigms of PLs
- Present formal approaches to syntax and semantics
- Present ways of implementing and analyzing programs in various PLs
- Show the concepts that must be dealt with by any PL, and the possible variety in treatment
19. Problem...

• To teach you PL theory, we need to draw examples from different PLs.
• Right now, most of you know 2.5 languages (C, C++, Unix shell scripts).
• Examples in these slides come from (alphabetically): ADA, ALGOL, AWK, C, C++, C#, EIFFEL, FORTRAN, HASKELL, JAVA, ML, Lazy-ML, LISP, PASCAL, PROLOG, PYTHON, SQL, and probably a few more I forgot.
• Can you please learn all these for next week?
• Recitations are here to help.

20. Who needs PLs?

• Computers’ native tongue is machine language
• Programmers need higher level languages, because:
  – They can’t write machine language correctly
  – They can’t read machine language fluently
  – They can’t express their ideas in machine language efficiently
  – Life is too short to program in machine language.
• A formal language is not only a man-machine interface, but also a person-to-person language!

Conclusion: PLs are a compromise between the needs of humans and the needs of machines

21. What is a PL?

• A linguistic tool with formal syntax and semantics
• A consciously designed artifact that can be implemented on computers
• “A conceptual universe for thinking about programming” (Alan Perlis, 1969)

22. Language processors

• A system for processing a language:
  – Compiler
  – Interpreter
  – Syntax directed editor
  – Program checker
  – Program verifier
• Studied in other courses:
  – Compilation
  – Program verification
  – Software engineering

To know the semantics of a language (the function a program encodes) one can ignore its implementation
/* HELLO WORD PROGRAM
TO OUTPUT HELLO WORLD */
HELLO: PROCEDURE OPTIONS (MAIN);
   PUT SKIP DATA('HELLO,\'WORLD');
END HELLO;

with i_o;
use i_o;
procedure hello is
   begin
      put ('\"Hello,\"World\"');
   end hello;

hello :-
   printstring('Hello,\"World\"'),
   printstring([]),
   printstring([H|T]) :-
      put(H),
      printstring(T).

OUTPUT = 'Hello,\"World'
END

Lots of food for one person
Hello World Souffle.
   Ingredients. 72 g haricot beans 101 eggs 108 g lard 111 cups oil
32 zucchinis 119 ml water 114 g red salmon 100 g dijon mustard 33
potatoes
   Method. Put potatoes into the mixing bowl. Put dijon mustard into
the mixing bowl. Put lard into the mixing bowl. Put oil into the
mixing bowl. Put water into the mixing bowl. Put red salmon into
the mixing bowl. Put zucchinis into the mixing bowl. Put oil into the
mixing bowl. Put lard into the mixing bowl. Put zucchinis into the
mixing bowl. Put oil into the mixing bowl. Put lard into the mixing
bowl. Put eggs into the mixing bowl. Put haricot beans into the
mixing bowl. Liquefy contents of the mixing bowl. Pour contents of
the mixing bowl into the baking dish.
   Serves 1.

( DEFUN HELLO-WORLD ( )
   (PRINT (LIST 'HELLO 'WORLD ) )
)

Transcript show: 'Hello,\"World'; cr

%!PS
1.00000 0.99083 scale
/Courier findfont 12 scalefont setfont
0 0 translate
/row 769 def
85 {/col 18 def 6 {col row moveto (Hello, World)
   show /col col 90 add def)
repeat /row row 9 sub def) repeat
showpage save restore

1.4 Thirteen weeks schedule
<table>
<thead>
<tr>
<th>Week</th>
<th>Topic</th>
<th>Lecture</th>
<th>Recitation</th>
<th>Tutorial</th>
<th>Self reading</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>(i) [11 frames] 1.1 Administration</td>
<td>(i) Regular expressions</td>
<td>PASCAL introduction</td>
<td>(i) Monograph First steps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ii) [11 frames] 1.2 Motivation</td>
<td>(ii) PASCAL’s EBNF</td>
<td>(ii) [11 frames] 1.3 Hello, World!</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(iii) [9 frames] 2.1 PL design</td>
<td>(iii) Specifying an EBNF with itself</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(iv) [10 frames] 2.2 Programming paradigms</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Introduction</td>
<td>(i) [25 frames] 2.4 Syntax specification</td>
<td>(i) EBNF example</td>
<td>ML introduction</td>
<td>Monograph Summary of first two lectures</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ii) [31 frames] 2.5 Tokens: the atoms of syntax</td>
<td>(ii) Regex examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Types</td>
<td>(i) [34 frames] 3.1 Value systems</td>
<td>(i) [9 frames] 3.5.1 Taxonomy of atomic types</td>
<td>ML curried functions</td>
<td>(i) [3 frames] 3.5.3 Integral primitive types</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ii) [11 frames] 3.2 Introduction to types</td>
<td>(ii) [5 frames] 3.5.2 Set of primitive types as PL i.d.</td>
<td></td>
<td>(ii) [6 frames] 3.5.5 Real numbers</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(iii) [2 frames] 3.5.4 More on language design</td>
<td></td>
<td>(iii) [2 frames] 3.5.7 Strings as atomic types</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(iv) [6 frames] 3.5.6 The character” primitive type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Types</td>
<td>[40 frames] 3.3 The type constructors of Mock</td>
<td>[27 frames] 3.4 Type constructors in actual PLs</td>
<td>ML lists</td>
<td>[6 frames] 3.6 Representation of types in memory</td>
</tr>
<tr>
<td>5</td>
<td>Types</td>
<td>[40 frames] 4.1 Classification of type systems</td>
<td>[27 frames] 4.4 Polymorphism in practice</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>[69 frames] 4.3 Theoretical polymorphism</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Storage</td>
<td>[21 frames] 5.1 Storage models</td>
<td>[24 frames] 5.2 Arrays</td>
<td>ML lists</td>
<td>TBD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[55 frames] 5.3 Variables’ life time</td>
<td>[16 frames] 5.5 Automatic memory management</td>
<td>ML datatypes</td>
<td>TBD</td>
</tr>
<tr>
<td>7</td>
<td>Storage</td>
<td>[27 frames] 5.4 Value vs. reference semantics</td>
<td>[16 frames] 5.6 Run time type information</td>
<td>ML exceptions</td>
<td>TBD</td>
</tr>
<tr>
<td>8</td>
<td>Storage</td>
<td>[27 frames] 5.4 Value vs. reference semantics</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Commands</td>
<td>(i) [10 frames] 6.1 Expressions vs. commands</td>
<td>[11 frames] 6.2 Recursive definitions</td>
<td>ML sequences</td>
<td>TBD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ii) [4 frames] 6.3 Atomic commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(iii) [6 frames] 6.4 Block commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Commands</td>
<td>(i) [11 frames] 6.7 Structured programming</td>
<td>(i) [13 frames] 6.5 Conditional commands</td>
<td>ProLOG introduction</td>
<td>TBD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ii) [11 frames] 6.8 Sequencers</td>
<td>(ii) [11 frames] 6.6 Iterative commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Advanced constructs</td>
<td>[19 frames] 7.1 Expressions’ evaluation order c</td>
<td>[13 frames] 7.3 Function objects</td>
<td>ProLOG lists</td>
<td>TBD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[19 frames] 7.1 Expressions’ evaluation order</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Advanced constructs</td>
<td>[41 frames] 7.2 Closures</td>
<td>[17 frames] 7.5 Iterators in JAVA</td>
<td>ProLOG controlling backtraceing</td>
<td>[8 frames] 7.6 Examples of JAVA iterators</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[17 frames] 7.2 Iterators in JAVA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Advanced constructs</td>
<td>(i) [11 frames] 7.4 Generators</td>
<td>Overflow of previous tutorials; if time is left, then “examples in modern languages”</td>
<td>ProLOG database examples</td>
<td>TBD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ii) [17 frames] 7.7 Coroutines</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*may be a bit stressed
*this may a bit too much
*there is room here for some more material

Table 1.4.1: Course schedule (semester of 13 weeks)
List of equations

(1.1.1) Course grade ($F$) computed from exam’s grade ($E$) and assignments’ grade ($A$) .................. 2

$$F = \begin{cases} 
\text{round}(E) & \text{if } E - 1.5 \leq A \leq E \\
\text{round}(E) & \text{if } E \leq 50 \\
\text{round}(E + \sqrt{A - E}) & \text{if } 50 < E \leq A \\
\text{round}(E - \sqrt{E - A}) & \text{otherwise}
\end{cases}$$

List of tables

1.4.1 Course schedule (semester of 13 weeks) . . . 6