CMPUT325 Introduction

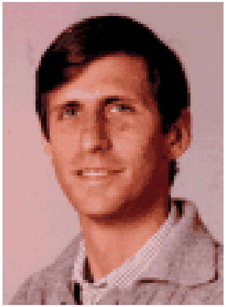
Dr. B. Price & Dr. R. Greiner

8th September 2004

CMPUT325 Non-Procedural Languages

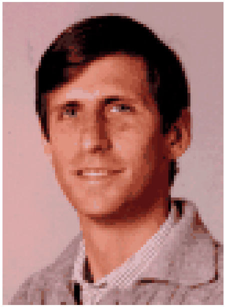
Instructor	Dr. B. Price	Dr. R. Greiner
	Functional Programming	Declarative Programming
Phone	492-0365	492-5461
Office	CSC 3-55	ATH 3-59
Office Hours	Immediately after class, or by arrangement	
Course Page	http://www.cs.ualberta.ca/~greiner/C-325	
News Group	*news:ualberta.courses.cmput.325*	

Pop Quiz



- Which of us is Dr G and which of us is Dr P?

Pop Quiz



Dr Greiner

Dr Price

- Which of us is Dr G and which of us is Dr P?

Lectures

- Section: A1
- Time: Tuesdays & Thursdays 9:30 - 10:50
- Place: V 102 (We have moved from the Tory Building)
- TAs: Kevin Andrusky, Tommy Chu, David Silver

The University on Course Outlines

Every course outline should contain the following statement:

The University on Course Outlines

Every course outline should contain the following statement:

- Policy about course outlines can be found in section 23.4(2) of the University Calendar

Tentative Term Calendar - Part I

Week		Topic
1	Sep 9	Introduction
2	Sep 14,16	Functional Programming in LISP
3	Sep 21,23	
4	Sep 28,30	**Ass #1 Due (Tue 28)**
5	Oct 5,7	Oct 5 withdrawal deadline
6	Oct 12,14	
7	Oct 19,21	**Ass #2 Due (Tue 19)**
8	Oct 26,28	**Midterm (Tue 26) in class** Declarative Programming (Thu 28)

Tentative Term Calendar - Part II

Week		Topic
9	Nov 2,4	
10	Nov 9,11	
11	Nov 16,18	
12	Nov 23,25	**Ass #3 Due (Tue 23)**
13	Nov 30, Dec 2	Other Programming Paradigms
14	Dec 7	**Ass#4 Due (Tue 7) Last lecture**
**	TBA	Final Exam
**	Jan 7, 2:00 pm	Deferred Exam Date

Textbooks

- There is NO REQUIRED TEXT for this course
- Course notes are available on the course page
- Additional notes will be made available during the term

Web Resources

- Professor Price & Greiner's Notes:
  http://www.cs.ualberta.ca/~greiner/C-325
- TA Tom Chu's On-line lab notes:
  http://www.cs.ualberta.ca/~tommy/
- Professor Mueller's Notes :
  http://www.cs.ualberta.ca/~mmueller/Courses/325Fall2003/index.
- Professor You's Notes:
  http://www.cs.ualberta.ca/~you/courses/325/w04/index.html

Optional References I

- Guy L. Steele Jr. *Common LISP: The Language*, 2nd Edition, Digital Press, 1990 Available online. {http://www.supelec.fr/docs/cltl/cltl2.html}
- P. Kogge,*The Architecture of Symbolic Computers*, McGraw-Hill, 1991. {SECD machine}

Optional References II

- L. Sterling and E. Shapiro, *The Art of Prolog*, Second Edition, MIT Press, 1994.
- L. C. Paulson, *ML for the Working Programmer*, Second Edition, Cambridge University Press, 1996.
- W. F. Clocksin and C. S. Mellish, *Programming in Prolog,* Springer-Verlag, 1994 (4th ed).
- Ivan Bratko, *Prolog: Programming for Artificial Intelligence*, Addison-Wesley, 2001 (3rd ed).

The University on Academic Integrity

Every course outline should contain the following statement:

- The University of Alberta is committed to the highest standards of academic integrity and honesty. Students are expected to be familiar with these standards regarding academic honesty and to uphold the policies of the University in this respect.
- Students are particularly urged to familiarize themselves with the provisions of the Code of Student Behavior (online at www.ualberta.ca/secretariat/appeals.htm) and avoid any behavior which could potentially result in suspicions of cheating, plagiarism, misrepresentation of facts and/or participation in an offence. Academic dishonesty is a serious offence and can result in suspension or expulsion from the University."

CMPUT 325 Policies on Integrity

Do not cheat on assigments:

- Discuss only *general approaches* to problem
- Do not take written notes on other's work

CMPUT 325 Policies on Integrity

Do not cheat on assigments:

- Discuss only *general approaches* to problem
- Do not take written notes on other’s work

Respect the lab environment. Do not:

- Interfere with the operation of the computing system
- Interfere with other’s files
- Change another’s password
- Copy another’s program
- etc.

CMPUT 325 Policies on Integrity

Do not cheat on assigments:

- Discuss only *general approaches* to problem
- Do not take written notes on other's work

Respect the lab environment. Do not:

- Interfere with the operation of the computing system
- Interfere with other's files
- Change another's password
- Copy another's program
- etc.
- Cheating is reported to university whereupon it is out of our hands

Consquences of Academic Offenses

Consequences include:

- A mark of 0 for the course
- A permanent note on student record
- Suspension from the university
- Expulsion from university

Evaluation

4 assignments totalling	40
Midterm exam	25
Final exam	35
Final grade	100

- NOTE: The final exam cannot be rewritten.
  (See the course schedule for date of deferred exam)

Assignment Guidelines

- Programming assignments should be
  - Neat & well documented
  - Include convincing examples and tests

Assignment Guidelines

- Programming assignments should be
  - Neat & well documented
  - Include convincing examples and tests
- The onus is on **you** to convince us it works

Assignment Guidelines

- Programming assignments should be
  - Neat & well documented
  - Include convincing examples and tests
- The onus is on **you** to convince us it works
- You may work anywhere you like, but your code must run on our lab computers

Assignment Guidelines

- Programming assignments should be
  - Neat & well documented
  - Include convincing examples and tests
- The onus is on **you** to convince us it works
- You may work anywhere you like, but your code must run on our lab computers
- Submit assignments using the "`try`" command (TA's will cover this)

Assignment Guidelines

- Programming assignments should be
  - Neat & well documented
  - Include convincing examples and tests
- The onus is on **you** to convince us it works
- You may work anywhere you like, but your code must run on our lab computers
- Submit assignments using the "`try`" command (TA's will cover this)
- You have 4 excused late days (max 2 per assignment)

Assignment Guidelines

- Programming assignments should be
  - Neat & well documented
  - Include convincing examples and tests
- The onus is on **you** to convince us it works
- You may work anywhere you like, but your code must run on our lab computers
- Submit assignments using the "`try`" command (TA's will cover this)
- You have 4 excused late days (max 2 per assignment)
- Assignments are due at 11:59 p.m. of the day they are due

GradeBook

- Current marks, late days, and class statistics will be available through "GradeBook"

  http://www.cs.ualberta.ca/~zaiane/courses
  /Tools/GradeBook/cgi-bin/GB_ShowBook.cgi?ci=C325-04

Labs

- Required software available in programming concepts lab 1-21 CSC
- Tutorials will be available in the labs in the CSC to help you get started
  - Starting programming environments,
  - Operating debuggers
  - Using language constructs,etc.
- TA's guaranteed to be available for the FIRST HOUR
- If a group requests a lab on a topic, we will schedule an additional lab

Tentative Scheduled Tutorials

Tutorial Dates	Topic
Sep 13-17	Introduction to Lisp
Sep 22,23,24 and 27,28	Assignment 1 Help
Oct 13,14,15 and 18,19	Assignment 2 Help
Nov 1-5	Introduction to Prolog
Nov 17,18,19 and 22,23	Assignment 3 Help
Dec 1,2,3 and 6,7	Assignment 4 Help

Goals of the Course I

- Computer scientists are interested in solving problems in many domains
  - Mathematical & numeric computations
  - Database transaction oriented computations
  - Event driven and interactive computations
  - Diagnostic, inferential and adaptive computations

Goals of the Course I

- Computer scientists are interested in solving problems in many domains
  - Mathematical & numeric computations
  - Database transaction oriented computations
  - Event driven and interactive computations
  - Diagnostic, inferential and adaptive computations
- Each domain emphasizes different styles of computation

Goals of the Course I

- Computer scientists are interested in solving problems in many domains
  - Mathematical & numeric computations
  - Database transaction oriented computations
  - Event driven and interactive computations
  - Diagnostic, inferential and adaptive computations
- Each domain emphasizes different styles of computation
- Languages allow us to formally describe computations (e.g. $\mathrm{average}(X) \equiv \frac{1}{|X|} \sum_i X_i$)

Goals of the Course II

- There are many, many different languages - too many to consider

Goals of the Course II

- There are many, many different languages - too many to consider
- Languages fall into a small number of paradigms

Goals of the Course II

- There are many, many different languages - too many to consider
- Languages fall into a small number of paradigms
- For any specific computation, a paradigm will vary in
  - Ease of expression of your problem
  - Efficiency of implementation
  - Verifiability & Maintainability

Goals of the Course II

- There are many, many different languages - too many to consider
- Languages fall into a small number of paradigms
- For any specific computation, a paradigm will vary in
  - Ease of expression of your problem
  - Efficiency of implementation
  - Verifiability & Maintainability
- Introductory CS courses emphasize the **procedural** paradigm

Goals of the Course II

- There are many, many different languages - too many to consider
- Languages fall into a small number of paradigms
- For any specific computation, a paradigm will vary in
    - Ease of expression of your problem
    - Efficiency of implementation
    - Verifiability & Maintainability
- Introductory CS courses emphasize the **procedural** paradigm
- Other important paradigms are more suitable for many applications

Goals of the Course III

- There are three main goals for the course:

Goals of the Course III

- There are three main goals for the course:
    1. Introduce additional paradigms and languages implementing them

Goals of the Course III

- There are three main goals for the course:
  1. Introduce additional paradigms and languages implementing them
  2. Give you practice working with paradigms so you'll be able to make informed choices for future projects

Goals of the Course III

- There are three main goals for the course:
  1. Introduce additional paradigms and languages implementing them
  2. Give you practice working with paradigms so you'll be able to make informed choices for future projects
  3. Acquaint you with underlying abstractions of these paradigms so that you will be able to
     - see relationships between families of languages
     - implement your own languages
     - reason abstractly about properties of languages (deciability, soundness, etc.)