CS 300: Programming Languages

Course Syllabus, Spring 2013

Contents

1 Course description

This course begins with an overview of programming languages syntax and semantics, and programming paradigms. You will learn the following languages and gain a beginning-level of programming expertise: C, C++, R and ggplot2, Python, Lisp (or Haskell), and Ruby.

2 Course Catalog description

Comparative study of the concepts found in contemporary programming languages. Emphasis is on design and evaluation of a language in terms of its features and their implementation. (PR: CS 210)

3 Instructor information and office hours

- Dr. V.N. Gudivada, Gullickson Hall Room 207, Phone: 304-696-5452, Email: gudivada@marshall.edu. Please use this email only if you cannot access the course muOnline email.
- Office hours: TBD; Other times by appointment.

4 Prerequisites

- CS 210 (Algorithm Analysis and Design)

5 Course topics

- Programming languages syntax and semantics
- Data and control abstractions
- Subprograms, modules, exceptions, and polymorphism
- C
- C++
- R and ggplot2
- Python

- ❍ Lisp (or Haskell)
- ❍ Ruby

6 BSCS degree program goals

a. an ability to apply knowledge of computing and mathematics appropriate to the discipline, including the ability to analyze and evaluate performance tradeoffs of algorithms, data structures, and hardware solutions;

b. an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;

c. an ability to design, implement, and evaluate a computer-based system, process, component, or program, including software systems of varying complexity, to meet desired needs;

d. an ability to function effectively on teams to accomplish a common goal;

e. an understanding of professional, ethical, legal, security, and social issues and responsibilities;

f. an ability to communicate effectively, both written and oral, with a range of audiences;

g. an ability to analyze the local and global impact of computing on individuals, organizations, and society;

h. a recognition of the need for and an ability to engage in continuing professional development;

i. an ability to use current techniques, skills, and tools necessary for computing practice, including the ability of expressing algorithms in at least two of the most important computer languages currently in use in academia and industry.

7 Course goals and relationship to BSCS program goals

After successful completion of this course, students should be able to:

① Understand the principles that underlie all programming languages (contributes to degree program goal i).

② Understand the basic structure of programming languages including syntax, semantics, data and control abstractions, pointers, subprograms, modules, exceptions, and polymorphism (contributes to degree program goals c and i).

③ Study different language paradigms: C (imperative), C++ (object-oriented), R (functional and object-oriented), Python, Ruby, and Lisp (multiple paradigms) (contributes to degree program goal c).

④ Understand, analyze, and document features of a contemporary programming language through self-study (contributes to degree program goals f and h).

8 Course timeline

❍ Week 1

- ◆ Principles that underlie all programming languages
- ◆ Syntax and semantics

❍ Week 2

- ◆ Data and control abstractions
- ◆ Subprograms, modules, exceptions, and polymorphism

❍ Week 3

- ◆ C
- ◆ C

❍ Week 4

- ◆ C
- ◆ C

❍ Week 5

- ◆ C++
- ◆ C++

❍ Week 6

- ◆ C++

- Midterm exam 1

- Week 7
  - R
  - R

- Week 8
  - R
  - ggplot2

- Week 9
  - ggplot2
  - Python

- Week 10
  - Python
  - Python

- Week 11
  - Python
  - Midterm exam 2

- Week 12
  - Lisp (or Haskell)
  - Lisp (or Haskell)

- Week 13
  - Lisp (or Haskell)
  - Ruby

- Week 14
  - Ruby
  - Ruby

- Week 15
  - Final exam

9 Instructional materials

In this course you will be learning 6 different programming languages. Therefore, it is not practical to require students to buy a textbook for each language. Lecture materials will be drawn from various resources listed in Bibliography in section 17. Lecture slides, handouts, URLs for additional Web resources will be made available on muOnline.

Web resources

- TIOBE Programming Community Index. http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
- Practical Common Lisp (free online textbook). http://www.gigamonkeys.com/book/
- Understanding Programming Languages (free online textbook). http://www.freetechbooks.com/understanding-programming-languages-t657.html

10 Writing assignments

There will be two writing assignments, one corresponding each of the following topics:

- Explain basic concepts of programming languages and categorize languages using various facets.
- Analyze and summarize a programming language using various facets.

11 Programming assignments

There will be a total of six programming assignments. You will develop a program in each of the following languages:

- C
- C++
- R and ggplot2
- Python

- Lisp
- Ruby

12 Course assessment

The course assessment components include: writing assignments (10%), programming projects (30%), and two midterm exams (@20% each), and final exam (20%). Course grade is awarded based on the following scheme:

Score	Letter Grade
>= 90	A
>= 80 & < 90	B
>= 70 & < 80	C
>= 60 & < 70	D
< 60	F

13 Attendance policy

Students are required to attend all class meetings. University approved excuse is required for any missed classes.

14 Classroom etiquette

- Students are expected to show up for class on time and remain in the class for the entire duration of the class.
- Students are not allowed to use personal laptops during the lecture part of the class.
- All types of phones and personal digital assistants must be turned off or put in silent mode during lectures.
- While taking tests, all types of electronic gadgets including cell phones, iPhones, iPod touch, blackberries, laptops must be turned off. No internet browsing is allowed during test taking.

15 muOnline

It is important to visit muOnline regularly for up-to-date information about the course. It hosts all the course materials including assignments, handouts, lecture notes, and reading materials.

16 Policy for students with disabilities

Marshall University is committed to equal opportunity in education for all students, including those with physical, learning and psychological disabilities. University policy states that it is the responsibility of students with disabilities to contact the Office of Disabled Student Services (DSS) in Prichard Hall 117, phone 304-696-2271, to provide documentation of their disability. Following this, the DSS Coordinator will send a letter to each of the student's instructors outlining the academic accommodation he/she will need to ensure equality in classroom experiences, outside assignment, testing and grading. The instructor and student will meet to discuss how the accommodation(s) requested will be provided. For more information, please visit http://www.marshall.edu/disabled or contact Disabled Student Services Office at Prichard Hall 117, phone 304-696-2271.

17 Bibliography

References

[1] Joseph Adler. *R in a Nutshell: A Desktop Quick Reference*. O'Reilly Media, Sabestopol, CA, 2010.

[2] David Ascher, Alex Martelli, and Anna Ravenscroft. *Python Cookbook*. O'Reilly, second edition, 2005.

[3] Conrad Barski. *Land of Lisp: Learn to Program in Lisp, One Game at a Time!* No Starch Press, 2010.

[4] M. Ben-Ari. *Understanding Programming Languages*. http://www.freetechbooks.com/understanding-programming-languages-t657.html, 2006.

[5] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, 2009.

[6] David A. Black. *The Well-Grounded Rubyist*. Manning Publications Co., 2009.

[7] Vernon L. Ceder. *The Quick Python Book*. Manning Publications Co., 2010.

[8] Winston Chang. *R Graphics Cookbook: Practical Recipes for Visualizing Data*. O'Reilly Media, November 2012.

[9] Huw Collingbourne. *The Book of Ruby: A Hands-On Guide for the Adventurous*. No Starch Press, 2011.

[10] Ingo Feinerer. Introduction to the tm package: Text mining in r, February 2011.

[11] Ingo Feinerer, Kurt Hornik, and David Meyer. Text mining infrastructure in r. *Journal of Statistical Software*, 25(5):1 – 54, March 2008.

[12] David Flanagan and Yukihiro Matsumoto. *The Ruby Programming Language*. O'Reilly Media, Inc., 2008.

[13] Louise Francis and Matt Flynn. Text mining handbook, 2011.

[14] Jeri R. Hanley and Elliot B. Koffman. *Problem Solving and Program Design in C*. Addison Wesley, sixth edition, 2009.

[15] Robert I. Kabacoff. *R in Action: Data Analysis and Graphics with R*. Manning Publications Co., 2011.

[16] Brian W. Kernighan and Dennis M. Ritchie. *C Programming Language*. Prentice Hall, second edition, 1988.

[17] K.N. King. *C Programming: A Modern Approach*. W.W. Norton, 1996.

[18] Kent D. Lee. *Programming Languages: An Active Learning Approach*. Springer, 2008.

[19] Jeremy Leipzig and Xiao-Yi Li. *Data Mashups in R: A Case Study in Real-World Data Analysis*. O'Reilly Media, 2011.

[20] Paul D. Lewis. *R for Medicine and Biology*. Jones and Bartlett, Boston, MA, 2010.

[21] Ray Lischner. *C++ In a Nutshell: A Desktop Quick Reference*. O'Reilly Media, Inc., 2003.

[22] Mark Lutz. *Learning Python*. O'Reilly, fourth edition, 2009.

[23] Mark Lutz. *Programming Python*. O'Reilly Media, Inc., fourth edition, 2010.

[24] Open Access. The R Journal. http://journal.r-project.org/index.html, 2012.

[25] Peter Prinz and Tony Crawford. *C in a Nutshell*. O'Reilly Media, Inc., 2005.

[26] R. A Free Software Environment for Statistical Computing and Graphics. www.r-project.org/, May 2012.

[27] R. International R User Conference. http://biostat.mc.vanderbilt.edu/wiki/Main/UseR-2012, 2012.

[28] Deepayan Sarkar. *Lattice: Multivariate Data Visualization with R*. Springer, New York, 2008.

[29] Peter Seibel. *Practical Common Lisp*. Apress, 2005.

[30] Bruce A. Tate. *Seven Languages in Seven Weeks: A Pragmatic Guide to Learning Programming Languages*. The Pragmatic Bookshelf, 2010.

[31] Paul Teetor. *R Cookbook*. O'Reilly Media, Sabestopol, CA, 2011.

[32] TIOBE Software. Tiobe programming community index. http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html, 2013.

[33] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer, 2009.

[34] Hadley Wickham. The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, 40(1):1 – 29, April 2011.

[35] Hadley Wickham. ggplot2. http://ggplot2.org/, 2013.