

MEBI 531: Life and Death Computing

Syllabus as of September 5, 2008

1 Instructor information

Instructor: Ira J. Kalet, Ph.D.

Office: NN146A University of Washington Medical Center

Email: ikalet@u.washington.edu

Phone: 206 598-4107

2 Course description

MEBI 531 addresses the complex software design issues that come up in biomedical and health informatics, programming for safety critical applications in medicine and health care, as well as in the application of computing to unlock the secrets of life. Examples from biology, medicine and health motivate software engineering topics such as: use of abstraction layers, design of tightly coupled but modular and extensible software, formal models and safety in real time control of medical equipment, design of network application protocols, integration of diverse biomedical data sources.

Many of the examples will use the Common Lisp programming language, but prior knowledge of Lisp is not assumed. The course introduces programming in Common Lisp as well as concepts and methods useable in most programming languages and environments. If time allows, the Prolog programming language will also be introduced and used.

The course objectives, prerequisites, schedule and other current information may also be found on the course web site, <http://www.radonc.washington.edu/medinfo/mebi531/>.

2.1 Learning objectives

At the completion of the course, students will be able to identify, analyze and solve problems in a range of areas of biomedical computing, including:

- distinguish between modular and non-modular designs with respect to biomedical software
- identify properties of software that may enhance or detract from effective use
- identify issues that should be addressed in designing network based medical applications and network protocols for medical data
- distinguish between software correctness, safety and quality in a biomedical context
- explain why biomedical software safety must be considered in context with the hardware and user environment
- explain how not understanding the algorithm(s) a medical program uses can lead to serious consequences
- analyze a biomedical programming problem and give reasons for using or not using particular programming languages, software environments or methodologies

- identify potential areas in a biomedical computer program where speed and/or memory usage could be an important issue, and describe how you would determine it.

In addition, students will have gained strong basic skills with programming in Common Lisp, including the Common Lisp Object System and the Meta-Object Protocol.

2.2 Prerequisites

The following skills and knowledge are needed for success in the course:

Biomedical knowledge: the student should be familiar with basic concepts of modern biology (genes, proteins, DNA, basic functions of living organisms), and have some experience with disease, treatment and the health care system.

Computer skills and knowledge: More important than skill in any particular programming language is general familiarity with the programming environment, including:

- know how to type commands in a command window, including correction of typing errors (when can you use backspace, delete etc.), know the basic commands - run a program by name, list the files in a directory, print files, change the working directory, rename, copy, and delete files,
- be able to create and edit files of plain text using a text editor (NOT a word processing program),
- be able to create a "terminal emulation" window and login to a remote (time sharing) computer that is running a Unix type operating system, using a secure connection (ssh),
- be able to copy files from one computer to another using secure copy programs,
- be able to use Internet domain names and IP addresses in the above where appropriate, and
- be able to navigate web sites where programs are available for download, and to be able to download and install programs like compilers, interpreters and text editors for various programming environments.

Mathematical skills and knowledge: Competence at basic algebra and trigonometry, and basic understanding of functions, variables and graphs.

2.3 Topics covered

The topics will depend on student interest but typically include the following:

- Symbolic computing in biology, medicine and health,
- Design for safety in computer controlled medical devices,
- Principles of networking, medical data interchange protocols,
- System and data security in medical centers, the role of the FDA and other issues,
- Federated access to data,
- Building and managing large biomedical application systems.

Other topics are covered on request depending on student interest.

3 Course format

The course meets for two lecture/classroom sessions per week, each one hour and 20 minutes. Approximately 6 hours of outside time are expected to devote to homework, reading and study. There will be several homework assignments, and a course project, on which the student is expected to report both orally and in writing.

The schedule of lectures each year is posted on the course web site.

4 Textbooks

1. Some of the course material is covered in “Principles of Biomedical Informatics”, by Ira Kalet (Academic Press, October 2008).
2. “ANSI Common Lisp”, by Paul Graham, is also assigned, in order to provide a text for learning Common Lisp.
3. Other course material will be covered in assigned readings and course handouts.

5 Other resources

The Informatics computing lab of the Biomedical and Health Informatics Graduate Program provides a Linux system with a Common Lisp programming environment. To obtain an account and learn about access, after registering for the course, contact the Informatics Lab Manager, in room T-277, Health Sciences Building.

6 Assignments

Each student is expected to complete each assignment by the indicated due date. Homework assignments should be the individual work of each student, although you are allowed to discuss the assignments with each other.

A Course Project is required. The project is a programming project that involves solving a biomedical problem in one of the areas studied in the course. A written report and an oral presentation of the project will be required. Projects may be done individually or in groups. If a group does a project, the report must delineate which parts of the work were done by each participant.

All students are expected to abide by the University of Washington’s Statement on Academic Honesty, <http://depts.washington.edu/grading/issue1/honesty.htm>

7 Grading

The course is graded; the grade is based on the homework assignments and a final exam or final project. The homework assignments collectively will comprise 50% of your grade, and the exam or project will be 30%. The remaining 20% will be based on class participation.

8 Class participation

This class is small, with a lot of opportunity for participation. You are expected to observe common courtesy, listen carefully to whomever is speaking, avoid potentially hurtful or insulting comments, show respect to everyone in the room. Although computing may seem to be a dry topic on which little can be discussed, it is my experience that there is a lot to talk about, and I am open to a wide range of questions and ideas.