Essential Computing for Bio-Informatics

Description: This course provides a broad yet intense overview of the discipline of Computer Science at a level suitable for a mature graduate student audience. It first discusses the most important mathematical computing models and uses them to illustrate the fundamental limits of computation. The course then discusses basic concepts of computer architecture and proceeds to introduce software development in a high level language.

Objectives: The central objective of this course is to provide graduate students in Bio-informatics whose core academic backgrounds lye outside computer science with a set of fundamental concepts in the discipline necessary to take advanced courses in Bio-Informatics.

Specific Objectives:

At the end of this course the student will have attained:

1. Knowledge of the fundamental mathematical models of computing
2. Understanding of the fundamental limits of computation
3. Knowledge of how information is represented inside the computer
4. Basic understanding of the inner working of a computer system
5. Ability to design and analyze fundamental computer algorithms
6. Ability to design computer programs in a modern high level language
7. Experience with commonly used software development environments and operating systems.

Units: Three credits

Pre-Requisites: none

Grading: Computing laboratories and homework exercises will account for about 50% of the final grade. Partial exams account for 25% and a final exam accounts for the remaining 25%. Some of the programming assignments will involve teams of students who will orally present their results.

Instructional Strategies: The course material will be mainly presented in lectures. Students will be assigned computing laboratories and homework exercises to complement the lectures and provide an active learning experience.

Topics to be covered in the course:

- **Mathematical Computing:** computing models, fundamental limits of computation, computability and intractability.

- **Physical computing:** essential computer architecture, Von Neumann architecture, binary information encoding: integer numbers, floating-point numbers, symbols; from logic gates to processors, machine instructions and assembly language.
- **Algorithms and Data Structures:** Big O notation, algorithm design, recursion, divide and conquer, dynamic programming.
- **Programming languages and paradigms:** imperative, functional, object oriented, scripting.
- **Software engineering concepts:** software development cycle, design, API's, SDK's, debugging, testing, performance tuning.

References:

- **The Practice of Programming**, B. Kernigham & R. Pike.
- **Introduction to Computing Systems: Form Bits and Gates to C and Beyond**. Y. Patt.