Comprehensive Course Syllabus

Course Title
Computational Science

Course Description:
For inclusion in the Learning Opportunities document for student registration, provide a brief overview of the course. Suggested length: 50 to 100 words.
Computational Science offers an introduction to using computer programming to solve science problems. Students will learn to apply programs they have written to real problems in physics, chemistry, biology, and other sciences. The course will discuss Monte Carlo methodology, minimization, finite element analysis, machine learning, and simulations. Assignments apply object orientation, polymorphism, and data structures to problems such as projectile motion, thermodynamics, reaction rates, natural selection, gravitational interactions, and population dynamics.

I. INSTRUCTOR(S):

- *Name(s): Peter Dong
- Office Number(s) (When and where you are available for help.): B117
- Telephone number(s): (630) 907-5476
- Email address(es): pdong@imsa.edu

Meeting Days, Time and Room(s)
in2, A-C days, mods 5-6
in2, B-D days, mods 7-8

Text(s) / Materials:
No textbook will be used. Students will use their own computers, and additional material is available online.

Essential Content:
Identify the disciplinary learning standards that will be addressed in this course. Specify by subset wherever possible. That is, each academic/programmatic area has between 7 and 16 specific learning standards, each with further specific aspects. Include both the broad learning standard and the specific aspects which the course will address. See IMSA Core Curriculum Template for models.

The relevant Science Team standards are:
A.2 designing and planning investigations and constructing questions which further understanding, forge connections, and deepen meaning. [IL-11.A.5b][NSES-A]
A.3 carrying out investigations that develop skills, concepts, and processes that support and enable complex thought. [IL-11.A.5c][NSES-A]
A.4 using appropriate technologies to collect, analyze and present information. [IL-11.A.5c][NSES-A]
A.6 supporting judgments and constructing models based on evidence. [IL-11.A.5e][NSES-A]
A.8 examining current issues in science and technology. [IL-][NSES-G]

The primary standard is, of course, A.4, since the entire course is predicated on using technology to analyze information. However, this requires students to develop and plan out their investigations (A.2 and A.3). In fact, the planning of the investigation (in this case, the program the students are writing) is more important than the details of the program – the students will learn how to use computer programs to answer scientific questions and make scientific predictions. This leads directly to standard A.6, since they will be constructing computer models, testing them against experiment, and drawing appropriate conclusions.

Some students, based on their prior knowledge, may also gain substantial understanding of a specific content area in chemistry, biology, or physics; for example, those who have not taken EBE will learn a lot about ecology and genetics, and students may get substantial insight into statistical mechanics and other areas such as astronomy or geology.

SSLs and Outcomes:
Identify the targeted SSL(s) and outcomes that will be addressed in this course. Specify by subset. Again, see IMSA core Curriculum Template for models.
The primary SSL for this course is, of course, III-A: Use appropriate technologies as extensions of the mind. Supporting this will be I-A: Develop automaticity in skills, concepts, and processes that support and enable complex thought. In this case, I-A refers to programming skills, used to allow students to achieve III-A, which allows them to improve their skills in several other SSLs, including:

I.B Construct questions which further understanding, forge connections, and deepen meaning.
I.D Evaluate the soundness and relevance of information and reasoning.
II.B Recognize, pursue, and explain substantive connections within and among areas of knowledge.
III.C Recreate the beautiful conceptions that give coherence to structures of thought.

SSL I-A will be met by a semester's worth of programming projects, which will help students develop automaticity in basic programming. This will allow them to use computers usefully, as extensions of the mind, to aid scientific research (SSL III-A). Lessons, homework, and projects, will all be focused on the ability to transfer scientific concepts into programming constructs that can be used to answer scientific questions. This is the primary outcome of the class. However, students will also practice coming up with specific questions formulated in a way that computer programs can answer (I-B) when planning projects; they evaluate the accuracy and reliability (or, more often, lack thereof) of computer simulations (I-D) when we test their computer predictions against reality; they use similarity of program structure to explore connections between different branches of science (II-B) when we use the same libraries in different applications; and, overall, they recreate in a computer the conceptions that give coherence to scientific thought (III-C).

***Instructional Design and Approach:**
In what specific ways will this course engineer student engagement? In what specific ways will this course further personalization? How is at least one of the core competency design principles for problem-centered, inquiry-based, integrative student learning experiences central to the design and implementation of this course? Consult IMSA Core Curriculum Template for guidelines noted in Essential Experiential Aspect.
This course is designed to be open-ended and project-driven, and is integrative by its very nature. Students are expected to be actively designing and programming for most of each class period. While some assignments will be structured and lead students through the methodology, others will be very open-ended, thus encouraging inquiry. Projects will be oriented around science problems, and thus the class is heavily problem-centered.

Student Expectations:
Delineate expectations for student behavior governing social interactions and academics. Some important considerations include late assignments, tardiness, attendance, and academic honesty. Maintain consistency with the Student Handbook.
Students are expected to show maturity and the ability to work well outside of class. Late assignments will be penalized 10% per 24-hour period or fraction thereof. However, students are given five *indulgences* per quarter, by which they can be late without penalty. Tardiness and unexcused absences will be dealt with according to standard IMSA policy. Students are expected to remain on-task on their computer for the entire class period. Students who are consistently and conspicuously off-task are a detriment to the learning environment and may be penalized without warning by loss of points on the relevant assignment.

Students are expected to write their own code themselves. They are encouraged to talk to others, including the teacher, with the understanding that all the actual typing was done individually. Tangential algorithms for a large assignment may come from online sources and should be properly identified in the code.

Assessment Practices, Procedures, and Processes:
Indicate the projected number of formative and summative assessments, projected assessment timeline, kinds of assessments consistent with the content and approach of the course, a brief philosophy of assessment particular to the proposed course, and a grading scale. Maintain consistency with the Student Handbook.

All assessments are programming assignments, categorized as homework or projects. Homework assignments consist of programming tasks of increasing difficulty that build upon one another, labeled as Level I, Level II, Level III, and Challenge. Assignments will be graded with a maximum score based on which level the students successfully complete:

Level I – 70%
Level II – 80%
Level III – 90%
Challenge – 100%

Code will be evaluated on design, clarity, and robustness, in roughly that order. "Design" refers to a sensible translation of physical quantities and variables into computational constructs. This includes lack of cut-and-paste in code, good use of functions, class structure, proper use of inheritance, and well-chosen data structures. "Clarity" refers to proper physical structure of code: good use of space and indentation, good comments, well-named variables, and lack of "magic numbers" or confusing code constructs. "Robustness" refers to program behavior in situations of unusual inputs and includes error checking, array bounds-checking, and exception handling. Note that there is little emphasis on algorithmic structure, which is not the focus of the class.

Homework assignments will generally be weighted equally. Each of the four sub-units will have a final project which will be worth three times a homework assignment. Students will complete the last project during finals period, but there are no timed examinations in this class.

Sequence of Topics and Activities
*Identify the sequence of topics the course will address and any special activities such as field trips, student presentations, etc., included in the sequence. Provide *an estimate of time allotted for each topic. A calendar which incorporates the sequence of topics, activities, and major assessments and assignments is strongly recommended.*

1/16-1/17: Introduction of the class, C# and Visual Studio, and Euler's method.

1/18-1/19: Collect kinematics homework, with one-dimensional motion and forces.

1/22-1/23: Collect first try at projectile motion homework. Discuss concrete classes, operator overloading, and properties.

1/24-1/25: Collect vector class library. Introduce concepts of problem domain and solution domain. Introduce classes as models of the problem domain.

1/28-1/29: Collect projectile class library. Discuss the use and importance of interface inheritance and adapter classes.

1/31-2/1: Collect adapter class for graphical interface. Discuss Debug vs. Release modes.

2/4-2/5: Collect final projectile motion assignment. Introduce simple finite element analysis for modeling behavior of extended objects.

2/8-2/9: Discuss Snell's law. Discuss raycasting as a method of modeling optics.

2/11-2/12: Collect optics assignment. Discuss principles of sound propagation.

2/14-2/19: Work on measuring room for sound project calculations

2/21-2/22: Collect predictions of volumes at different points in the room. Measure to see the result.

2/25-2/26: Discuss Monte Carlo methodology and drawing from a random distribution.

2/27-2/28: Collect thermodynamics assignment, which models the random motion of molecules to model temperature.

3/4-3/5: Discuss principles of Monte Carlo as applied to chemical reactions

3/7-3/8: Collect chemical reactions assignment. Introduce iodine clock.

3/11-3/12: Work on chemical model of reaction.

3/14-/3/15: Collect chemical predictions. Introduce principles of inference.

3/25-3/26: Dr. Dong not here. Work on optimization.

3/28-3/29: Dr. Dong not here. Collect optimization homework.

4/1-4/2: Dr. Dong not here. Discuss machine learning and decision trees.

4/4-4/5: Dr. Dong not here. Collect machine learning homework.

4/8-4/9: Test chemical reactions. Discuss parallel processing.

4/11-4/12: Collect write-up and reflection on chemical reactions.

4/15-4/16: Collect parallel processing assignment

4/18-4/23: Discuss scattering, Rutherford and crystallography.

4/24-4/25: Collect crystallography setups.

4/29-4/30: Work on discerning crystallography designs.

5/2-5/3: Collect crystallography predictions.

5/6-5/7: Discuss modeling of natural selection.

5/9-5/10: Collect natural selection assignment.

5/13-5/14: Discuss evolutionary algorithms and Bayesian inference.

5/15-5/16: Collect predator/prey assignment.

5/20-5/21: Work on Hunger Games creatures.

5/23-5/24: Collect Hunger Games creations.

Finals: Host the fifth annual Hunger Games in in2.