Interdisciplinary Introductory Course in Bioinformatics

Yana Kortsarts
ykortsarts@mail.widener.edu
Computer Science Department

Robert Morris
rwmorris@widener.edu
Biology Department

Janine Utell
jmutell@mail.widener.edu
English Department

Widener University
One University Place, Chester, PA 19013, USA

Abstract

Bioinformatics is a relatively new interdisciplinary field that integrates computer science, mathematics, biology, and information technology to manage, analyze, and understand biological, biochemical and biophysical information. We present our experience in teaching an interdisciplinary course, Introduction to Bioinformatics, which was developed and taught by an interdisciplinary team of computer science and biology faculty. To integrate the "Ethics, Computing, and Genomics" component into the course curriculum, collaboration with an English faculty member was established. The course was designed as an upper-level elective for computer information systems and computer science majors and was open as a science elective for all science majors and for chemical engineering majors. We discuss the course curriculum, and a way to integrate the Python programming language in the UNIX environment will be presented. The integration of bioinformatics algorithms into the course curriculum will be emphasized.

Keywords: bioinformatics, pedagogy, interdisciplinary collaboration, computer science and information systems curriculum, Python, Biopython, ethics, genomics and computing

1. INTRODUCTION AND MOTIVATION

Bioinformatics is a relatively new interdisciplinary field that integrates computer science, mathematics, biology, and information technology to manage, analyze, and understand biological, biochemical and biophysical information. According to a definition created by the National Institutes of Health (http://www.nih.gov/), bioinformatics is

research, development, or/and application of computational tools and approaches for expanding the use of biological, medical, behavioral or health data, including those to acquire, store, organize, analyze, or visualize such data. Bioinformatics is a computational science and the subset of the larger field of computational biology. Computing and information technologies have transformed research and business in the life sciences, facilitating new advances in molecular biology and consequent new applications in biotechnology, pharmaceuticals, healthcare, the environment, the chemical industry and agriculture. Bioinformatics is first a biological science. However, bioinformatics has affected more than just biology – it has also had a profound impact on the computational sciences. Biology has rapidly become a large source of new algorithmic and statistical problems (Jones & Pevzner, 2004). The IS 2002 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems (Gorgone, 2002) notes several characteristics of the IS profession that are integrated into the curriculum. One of these characteristics is "IS professionals must have strong analytical and critical thinking skills." In our opinion, introducing bioinformatics to CIS students will strengthen these required skills. It will also equip the students with some of the following capabilities as suggested in the IS 2002 guidelines:

- Creativity
- Application of both traditional and new concepts and skills
- Application development
- Problem solving abilities
- Ability to communicate effectively (oral, written and listening)

Integrating bioinformatics into the undergraduate CIS and CS curriculum provides opportunities for students to become familiar with one of the most widely used script languages, Python, and to explore various data structures and algorithmic techniques traditionally not covered in other courses. This helps students to make connections between theoretical topics learned in core CS and CIS courses, such as Data Structures and Algorithms, and to apply their knowledge to real world biology problems. Introducing bioinformatics helps to diversify department course offering and provides interdisciplinary opportunities for CS and CIS students. In addition, CIS and CS students with a bioinformatics background clearly will enhance their employment qualifications in the competitive job market (Cohen, 2004; Cohen, 2005a; Cohen, 2005b).

Bioinformatics has a clear interdisciplinary nature. In this paper, we report and discuss our experience and course results teaching an interdisciplinary course, Introduction to Bioinformatics. The first iteration of the course was offered in Spring 05. A revised version was offered again in Spring 08. We discuss the initial implementation, as well as revision and updates that were made, course results, and future plans. The course was taught by an interdisciplinary team of computer science and biology faculty. To integrate the "Ethics, Computing, and Genomics" component into the course curriculum, collaboration with an English faculty member was established. The course was designed as an upper-level elective for computer information systems (CIS) and computer science (CS) majors and was open as a science elective for all science and chemical engineering majors. Since our department covers both curricula (Computer Science and Computer Information Systems), many courses are designed for both majors. This gives the opportunity for CIS majors to partake of a wide range of elective courses offered by our department.

2. COURSE DESIGN, GOALS, OBJECTIVES AND CHALLENGES

Recently, many universities and colleges have started to offer bioinformatics courses on different levels and to design bioinformatics minor and major curricula (LeBlanc & Dyer, 2003; Burhans & Skuse, 2004; LeBlanc & Dyer, 2004; Morrow & Wilkins, 2004; Toth & Ronnelly, 2006; Tjaden, 2007; Zhang, Lin, Olsen, Beck, 2007; Goode & Trajkovski, 2007; Gosukonda & Naghedolfeizi, 2007; Bruhn & Jennings, 2007; Khuri, 2008). Some universities and colleges design upper-level elective courses in computational biology that focus on the design and analysis of algorithms with applications in molecular biology. Other departments design upper-level elective courses in bioinformatics that pair a biology major with a computer science major to learn from the challenges in the other discipline from peer-to-peer experience. In some situations, the course emphasizes the informatics issues and can be adapted to fit the interests and experience of a faculty member in computer

science who has only a moderate amount of familiarity with molecular biology. While there are experiences to learn from, the area is still very young, and designing an introductory bioinformatics stand-alone course takes a lot of effort; the individual features of the department should be taken into account, as well as faculty resources and available lab resources. The initial goal was to design an interdisciplinary course that would bring together CS and CIS students and undergraduate students from different science majors and would provide an opportunity for interdisciplinary collaboration in the in-class laboratory assignments and team projects. Mainly, we targeted the course to biology students. We expected that Computer Science and Computer Information Systems students would have a solid background in programming and algorithms, while biology students would have solid biology background. Part of the course experience would be the blending of student expertise in the formation of teams to solve biological problems. With all this in mind, the Introduction to Bioinformatics course was designed and offered for the first time in Spring 2005 with an enrollment of 12 students, where 6 of the students were CIS and CS majors and 6 students were biology majors. This balanced enrollment allowed us to create six interdisciplinary teams. Each team included one biology student and one CIS or CS student. Most of the coursework was done in teams. The course was taught by computer science and biology faculty. The course met in the lab 6 hours weekly. The development of this course was a challenging task for several reasons. First, the diverse background of the students had to be taken into account. Second, the programming language to be used in this course had to address the current and future needs of computational molecular biology, and at the same time had to be easy enough for the students to learn and to apply in bioinformatics problem solving. Third, careful thought had to be put into defining course prerequisites. Fourth, the goal was to design a well-balanced course from the content point of view, and to create a proper ratio between hands-on activities and lecture. Fifth, we experienced difficulties in finding one textbook that would answer all our needs. The following prerequisites for the course were determined for different majors: Biology/Chemistry/Biochemistry majors had as a prerequisite Introduction to Molecular Biology, CS/CIS/MATH majors had prerequisite Introduction to Computer Science I, and Chemical Engineering Majors had as a prerequisite Computer Programming and Engineering Problem Solving. After thoughtful consideration, Python was chosen to be a programming language for the course. There were several reasons for choosing this language for the bioinformatics course: simplicity, power, and object-oriented capabilities. Also, some biology students can take an Introduction to Programming with Python for non-majors offered by our department prior to the bioinformatics course and it would provide programming foundation knowledge for biology students who need it. In addition, most CS and CIS students who would take the bioinformatics course would not have had any prior knowledge of Python, and the bioinformatics course would provide an opportunity to learn Python and enrich their undergraduate curriculum; recently, the Python language has started to be used more often in bioinformatics as a general-purpose programming language (Mangalam, 2002). We were also able to introduce students to the Biopython Project, an international association of developers of freely available Python tools for computational molecular biology (http//biopython.org).

It is critical for students to understand biological problems as well as computational solutions in order to produce useful bioinformatics tools. Since bioinformatics is a computational science, it is important to introduce students to the principles that drive an algorithm's design. One of the important goals of the course was to introduce students to the intellectual content of bioinformatics. To achieve this goal, it was decided to integrate bioinformatics algorithms into the course and to teach the foundations of the algorithms and important theoretical results in bioinformatics along with end-user bioinformatics tools (Jones & Pevzner, 2004). For the first iteration, the biology faculty member taught 4 hours per week and the computer science faculty member 2 hours per week and the idea was to blend biology and computer science topics together. However, while keeping the main goal to blend computer science and biology topics, we realized that there was more computer science material to cover, and the ratio was thus changed to 2 hours per week for biology topics and 4 hours per week for computer

science topics for the second iteration of the course in Spring 08. Additionally, the course curriculum was updated and revised based on the initial experience and the enrollment changes were taken in account. We were prepared for the reality that only CIS and CS students were enrolled in the course in Spring 08. This gave us an opportunity to expand the programming/algorithms component of the course, to have a more complex final project and also to integrate a few additional topics. We will discuss the curriculum issues in detail in the following section.

3. COURSE CURRICULUM

The course started with an introductory lecture that provided several definitions of the term "bioinformatics" to give students an idea of what the course was about. We also explained the structure of the course, the tentative list of topics that would be covered, and the level of the theoretical and technical content. We also emphasized the interdisciplinary nature of the topic and of the course. The first topic covered in the course was "Ethics, Computing and Genomics". This was a project-oriented component and was a new addition integrated into the Spring 08 curriculum. The idea of this project was to give students an opportunity to learn about the ethics, computing and genomics topic independently and to present the results of the self-learning. To learn the topic, students were assigned one or more scholarly articles from the collection *Ethics, Computing, and Genomics*, edited by Herman Tavani. The collection is divided into five different sections, with introductions and lists of questions prior to each section. Students were required to read assigned essays; to prepare 25-minute Power Point presentations that would include a summary of the paper and answers to the questions posed in the introductory part of the corresponding section; and to prepare a mini-quiz to assess the understanding of the presented material by their peers. The goal of this component was to develop oral and written communication skills and to engage students in the knowledge exchange process. To accomplish these goals, an interdisciplinary collaboration with an English faculty member was established. The team of computer science and biology faculty met with the English professor and discussed the goals, structure and requirements of the Ethics, Computing and Genomics component, and decided that in order to improve students' performance, the English faculty member would do a short presentation before students started to work on this assignment. The presentation covered the following issues: (a) discussion of how to read critically and what questions to ask while reading the text; (b) discussion of how to summarize the paper using the structure of the essay as a guide and elucidating key points and key moments of evidence while making connections to the rest of the class material; (c) tips on writing the summary that include three steps: prewriting, drafting, revising; and (d) discussion of how to design an effective presentation of information. The English faculty member was present at all oral presentations and provided detailed notes for each student explaining ways the presentation could have been stronger and also pointing out the positive and negative aspects of the presentation. These valuable comments allowed students to improve their performance and their oral and written communication skills, and to complete the course with a well-designed presentation for the final course project that will be discussed later in the paper. The collaboration with the English faculty member provided an opportunity to accomplish the goals related to the development of oral and written communication skills. This successful and enjoyable experience showed the value of working with colleagues across disciplines to further student learning. The reading and research paper presentation assignment contributed 20% to the final course grade and students were given three weeks to work on this project. The rest of the course was taught by computer science and biology faculty. The list of the topics with explanations about the specific activities and ideas for resources is given below. The goal was to blend biology and computer science topics together to create a real-world atmosphere in the course where students could observe how the same problem was tackled from multiple perspectives.

Introduction to Python

One goal was to acquaint students with introductory Python very quickly during the first few weeks of the course, which would allow for working on different problem solving algorithmic techniques. Advanced Python

topics were taught later throughout the course, building students' knowledge and their abilities to tackle biology real-world problems. The programming examples were all biology-oriented and motivated students to learn in order to solve practical problems. During the first few weeks of the course, we introduced introductory topics such as Python arithmetic, decision and loop structures and functions, and as well simple manipulations with strings, lists, tuples and dictionaries. In Spring 05, we had six biology students without any prior programming experience and six CS and CIS students without any prior experience in Python. In order to make this part of the course successful, the class was divided into six interdisciplinary teams and all concepts were practiced within the team with the help of CS and CIS majors. In Spring 08, all students enrolled in the course were CS and CIS majors with prior programming experience in C and Java, and some with introductory knowledge in Python, as well. Special handouts were prepared to walk students through the introductory topics toward advanced Python concepts. Each topic was supported by a list of examples in increasing order of complexity. Students were required to run the proposed programs in order to gain understanding of basic Python structures. To assess the understanding of each concept, students were required to write short programs solving biology-oriented problems. A few examples of the problems, given here in increasing level of complexity, include computation of the alignment score between two DNA sequences using different score matrices; finding the maximal alignment score if no internal gaps are allowed using different score matrices; finding all occurrences of one sequence in another sequence; writing a program that reads a DNA sequence, first transcribing DNA into RNA and printing the resulting RNA sequence, then translating RNA into a protein sequence through the following: first, the program divides RNA into codons and prints the list of codons, and second, the codons are translated into the protein using genetic code table and finding the maximal alignment score if internal gaps are allowed using different score matrices. In Spring 08 we had students with different levels of programming and computational experience and the best way to cover this topic was through independent learning. Students used provided handouts and Python and BioPython tutorials (www.python.org, www.biopython.org) and worked each at their own pace. We provided grading rubrics for each programming concept, specified minimal requirements to pass the specific concept, and provided a list of more advanced examples for students with prior Python experience. This approach allowed students with previous Python knowledge to further advance their experience and students new to Python to learn the new programming language independently using structured guidance. Python also provides an opportunity to solve some problems in very short ways, and it was a very enjoyable experience for students to try to find a shortest solution for the proposed problems using Python functions and libraries.

Introduction to Bioinformatics Algorithms

To introduce students to the principles that drive an algorithm's design and to the intellectual content of bioinformatics, the bioinformatics algorithms component was integrated into the course (Jones & Pevzner, 2004). Students were introduced to the topics of sequence alignments, scoring matrices, and gaps. We covered exhaustive search and dynamic programming algorithm design techniques. To support the dynamic programming techniques with specific examples, the Needleman and Wunsch algorithm (Jones & Pevzner, 2004; Krane and Raymer, 2002) for finding an optimal global alignment and the Smith-Waterman algorithm (Jones & Pevzner, 2004; Krane and Raymer, 2002) for finding an optimal local alignment were introduced. We also discussed the concept of semi-global alignment and the slight modification of the Needleman and Wunsch algorithm for this purpose. Dynamic Programming technique usually is not covered in a core algorithms course, and including this topic in our course curriculum provided an opportunity to expand the theoretical background and to make connections between theory and practice. It also helped to maintain an appropriate level of theoretical content required for upper-level elective courses in our department. Additionally, this topic was very well blended with biology topics and students had an opportunity to learn the concept of sequence alignments from biology and computer science points of view.

It is a challenging topic for the instructor to teach and for the students to learn. For the first iteration of the course, the topic was studied in interdisciplinary teams, similar to the idea that was employed to teach Python. To enhance students' understanding of the theoretical aspects, the visualization of bioinformatics algorithms through the ALGGEN – EMBER website was used (http://alggen.lsi.upc.es/docencia/ember/frame-ember.html). This website provides a suite of multimedia bioinformatics educational tools and allows us to create a set of hands-on activities to help students to gain understanding of the dynamic programming technique in general and specific algorithms in particular. In this part of the course we also introduced the BLAST algorithm and the concept of multiple sequence alignment, and provided students with a brief theoretical background. We mainly focused on practical applications of these tools to solve biological problems. BLAST was the main tool that students used to complete a final course project. The multiple sequence alignment tool CLUSTALW was used in some of the hands-on activities of the biology part of the course, which are described below.

Biological Research on the Web

In this part of the course, which focused on both biology and computer science, students got a familiarity with public biological databases and data formats, and learned how to search biological databases to find required information (Gibas & Jambeck, 2001). The main website that students worked with in this course was NCBI – National Center for Biotechnology Information (http://www.ncbi.nlm.nih.gov/) – the site that was established in 1988 as a national resource for molecular biology.

Biology Topics

The biology part of the course started with a review of molecular biology and biochemistry concepts that covered DNA and protein structure, gene expression (transcription and translation), and molecular biology central dogma. Next, the concept of sequence alignment was introduced and discussed from different perspectives (Krane & Raymer, 2002). Students learned to use sequence alignment to understand relatedness among species and to use sequence alignment forensically. The hands-on activities to support this part of the course included the exploration HIV evolution lab developed by Sam Donovan and Anton E. Weisstein from Microbes Count!, which is a collection of multimedia resources, simulations, and tools that offer an interactive, open-ended environment for learning about microbiology (Jungck & Stanley, 2003). Human Immunodeficiency Virus (HIV), like other retroviruses, has a much higher mutation rate than is typically found in organisms that do not go through reverse transcription (the copying of RNA into DNA). The HIV genome is very small and relatively simple. It is made up of nine genes and about 9,500 nucleotides. In this lab students worked with HIV sequence data collected from 15 individuals from an intravenous-drug-using population in Baltimore. The goal of the study was to determine if the HIV isolated from particular subgroups of subjects derives from a common source. In order to approach this question students needed to characterize the strains of HIV within an individual and quantify the differences between individuals (Donovan & Weisstein, 2003). To accomplish the goals of the lab, students used Biology Workbench (http://workbench.sdsc.edu), a free, web-accessible suite of resources for working with molecular sequence and structure data. Biology Workbench provides a unified interface to access a variety of tools and databases; it is online and can be accessed with a standard web browser and it is available at no cost to academic users. The Biology Workbench was developed by Shankar Subramaniam and currently resides on a supercomputer at the San Diego Supercomputing Center. Thus, it provides access to both extensive computational power and huge data storage capacity. To complete the project the CLUSTALW multiple sequence alignment tool was used. In this part of the course, students also learned about microarrays. The microarray lab was developed by Campbell and Heyer and is sold by Carolina Biologicals and called DNA Chips: Genes to Disease. An interesting lab activity was designed to help students to understand how microarrays are used to identify gene changes in disease. This simulation provided an opportunity to introduce students to microarrays, the complexities of gene expression, and the role of gene expression in cancer. Using simulated microarray technology, students compared the relative expression levels of six different genes in healthy lung

cells and lung cancer cells. After completing the lab, students had an opportunity to discuss the significance of the relative expression levels with respect to the genes' roles in causing cancer (Heyer & Campbell). In addition, the biology part of the course provided an opportunity to use computer simulations to test hypotheses about disease spread. Students learned the concept of epidemiology, the study of the distribution of diseases in populations and explored factors that influence disease spread throughout populations with the software *Epidemiology. Ebola* was used as a model organism and epidemiology was presented from both a microbiological and social perspective (Jungck & Stanley, 2003). Also, students used a vast computer database of genetic information to explore the structure and function of insulin and to generate a phylogenetic tree demonstrating evolution of insulin amongst the vertebrates (Campbell & Reece, 2004).

4. COURSE PROJECT

In the final course project students were engaged in the process of DNA sequence annotation. Students worked with the real data. The microbial genome resources were used for these purposes and *Bacillus anthracis str. Ames* project at J. Craig Venter Institute was used as a specific organism (http://www.ncbi.nlm.nih.gov/). The input DNA sequence was about 50,000 nucleotides long, and students worked on different such sequences from the same organism. The project consisted of several steps. First, students were asked to find a list of all potential genes and pseudo-genes in the input DNA sequence, using the information about start and end codons, and to arrange the found sequences in two separate lists, one for potential genes (length of the sequence is larger than 300) and one for pseudo-genes (length of the sequence is less than 300), in order of increasing length. Second, students located the potential promoters in the given DNA sequences for each potential gene that they found in the first step, and calculated the strength of the promoter. A promoter is a region of DNA near the beginning of a gene that controls if and when the gene is actually expressed. As an output for this step, students were asked to list potential genes in order of decreasing promoter strength.

To complete the project students were asked to BLAST all potential genes and pseudo-genes that were found, and to perform an analysis of the results. Students were required to summarize the results in a table and provide the following information for each potential and pseudo-gene that they found: start position, length, promoter score, BLAST results, summary and conclusion. For each sequence, we asked students to determine whether a potential gene could be a real gene based on the strength of the promoter and BLAST results. Students were asked to present project results in 15-minute in-class presentations that included the presentation of the Python program, including the description of all Python functions that were used and the purpose of each function, as well as all algorithms or/and programming techniques that were used and the presentation and explanation of the summary of the project results, including the information about the specific organism whose DNA was used as the input. In the first iteration of the course, this was a team project. Each team consisted of a computer science and a biology student. For the first iteration, the programming part of the project was mostly done by the computer science students, and the biology students were required to understand and to explain the programming techniques and algorithms that were used. The project provided a possibility for truly interdisciplinary collaboration between computer science and biology students. In the second iteration of the course, in Spring 08, the students worked on the project individually. For future iterations of the course we are planning to return to team work in the project in order to enhance the collaborative component of the course.

5. COURSE RESULTS

For the first iteration of the course no formal assessment survey was conducted. An informal discussion about the course was conducted at the end of the semester and we asked students to provide their feedback. Also, as part of our regular departmental procedure, students completed teaching evaluations and provided their comments there as well. All students showed satisfaction with the course and we were very pleased to receive the request to extend the programming component of the course from

almost all students. Biology students showed interest in programming and asked that an environment be created where they would be able to more fully participate in all stages of the course project. In the second iteration of the course, we designed a short post-survey in order to assess the students' experience which included a list of the topics that were covered in the course. We asked students to rate the level of learning for each topic on a scale of 1 (not well) to 5 (very well). Six students were enrolled in the Spring 08 and the table with the list of the topics and the average ratings is given in the Appendix.

We also asked students to comment on the Ethics, Computing and Genomics component and received positive feedback from most of the students. In addition, we asked students to provide any comments regarding the course, and most of the students mentioned that they loved the course and would recommend it to their peers; they expressed their satisfaction with the level of the course and the amount of material covered and the depth of the coverage. They also mentioned that the final project was very interesting but at the same time they proposed that we be more careful with the project description and to provide clear rules for finding genes on the main and complement strings to avoid confusion.

6. LESSONS LEARNED AND FUTURE PLANS

Based on the students' post-survey results, informal discussions, and comments from teaching evaluations, we could state that the integration of an Introduction to Bioinformatics course into the CS/CIS curriculum was successful. For future iterations of the course we are planning to follow the existing structure in general, but based on our experience and taking into account some of the student's suggestions, several changes will be introduced. As shown in the assessment table in Appendix, all topics that were covered in the second iteration of the course were learned on an above average level, but it is easy to see that some of the topics required our special attention and should be revised for future iterations. The teaching approach will try to foster student learning through a research-based process. We will try to blend biology topics with computer science topics throughout the course more effectively, to make sure that students have a clear motivation for each CS/CIS task, and to make sure that each topic receives coverage from the two different perspectives. In addition, we think that a guest speaker from the field could enrich the course curriculum and we will try to arrange the talk or perhaps even a visit to a bioinformatics lab facility. We will continue to emphasize the interdisciplinary nature of the course and will try to design more collaborative teamwork even if the course enrollment consists of CS/CIS majors only. We will continue the integration of hands-on activities to make the course a fun and enjoyable experience. We will definitely continue the integration of the Ethics, Genomics and Computing component, but we would try to find different set of papers to stay current in the field and also to cover a wide range of aspects related to the topic. We will work on improving the post-survey by asking more specific questions. We will also try to attract more students to the course which will allow a larger sample for the course assessment. A special word should be said about the textbook. It was not an easy task to find one textbook that would answer all our needs since our course has several unique features; when combined together they didn't allow for an easy solution to the problem of textbook choice. For both initial iterations of the course we used several textbooks (see References section for the complete list of all textbooks that were used for the course). For succeeding iterations of the course we are planning to design our own custom text for the course using several textbooks and our own lecture notes with the help of a professional publisher. While the two initial implementations of the course were successful, all the above-mentioned ideas would definitely help to improve the course and we are looking forward to the future iterations.

7. REFERENCES

Biology Workbench Website: http://workbench.sdsc.edu

Biopython Website: http://biopython.org

Bruhn, Russel, Jennings, Steven (2007), "A multidisciplinary bioinformatics minor." Proceedings of the 38th SIGCSE technic-

al symposium on Computer science education, Pages: 348 - 352

Burhans, Debra, Skuse, Gary (2004), "The role of computer science in undergraduate bioinformatics education." Proceedings of the 35th SIGCSE technical symposium on Computer science education, Norfolk, Virginia, USA, Pages: 417 – 421

Campbell, Neil, Reece, Jane (2004), Biology, Benjamin Cummings; 7th edition

Cohen, Jacques (2004), "Bioinformatics—an introduction for computer scientists." ACM Computing Surveys (CSUR), Volume 36, Issue 2

Cohen, Jacques (2005), "Updating Computer Science Education." Communications of the ACM, 48(6), 29-31

Cohen, Jacques (2005), "Computer Science and Bioinformatics." Communications of the ACM

Gibas, C, Jambeck, P (2001), Developing Bioinformatics Computer Skills, O'Reilly

Goode, Elizabeth, Trajkovski, Goran (2007), "Developing a truly interdisciplinary bioinformatics track: work in progress." Journal of Computing Sciences in Colleges, Volume 22 , Issue 6 Pages: 73 – 79

Gorgone, John T., Gordon B. Davis, Joseph S. Valacich, Heikki Topi, David L. Feinstein, and Herbert E. Longenecker, Jr., (2002), IS 2002 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems, Association for Information Systems.

Gosukonda, Ramana, Naghedolfeizi, M (2007), "Initial evaluation of bioinformatics curriculum at Fort Valley State University." Journal of Computing Sciences in Colleges, Volume 23, Issue 2, Pages: 236 – 242

Heyer, Laurie J. and Campbell, A. Malcolm. Microarray Lab: DNA Chips: Genes to Disease. Carolina Biologicals, http://www.carolina.com/product/dna+chips+genes+to+disease+lab+kit.do?keyword=21-1520&sortby=bestMatches

Jones, Neil, Pevzner, Pavel (2004), An Introduction to Bioinformatics Algorithms (Computational Molecular Biology), The MIT Press

Jungck, John, Stanley, Ethel (2003), Microbes Count! The BioQUEST Curriculum Consortium, ACM Press, http://bioquest.org/microbescount/

Krane, D, Raymer, M (2002), Fundamental Concepts of Bioinformatics, Publisher: Benjamin Cummings

Khuri, Sami (2008), "A bioinformatics track in computer science." Proceedings of the 39th SIGCSE technical symposium on Computer science education, Pages 508-512

LeBlank, Marc, Dyer, Betsey (2004), "Bioinformatics and computing curricula 2001: why computer science is well positioned in a post-genomic world." Annual Joint Conference Integrating Technology into Computer Science Education, Working group reports from ITiCSE on Innovation and technology in computer science education, Pages: 64 - 68

LeBlank, Marc, Dyer, Betsey (2003), T"eaching together: a three-year case study in genomics." Journal of Computing Sciences in Colleges, Volume 18 , Issue 5, Pages: 85-95

Mangalam, Harry, (2002), The Bio* toolkits-a brief overview. Brief Bioinform 3(3):296-302

Morrow, Charles, Wilkins, Dawn, (2004), "A bioinformatics course in the computer science curriculum." ACM International Conference Proceeding Series; Vol. 61, Proceedings of the 2nd annual conference on Mid-south college computing Pages: 192 - 199

National Institutes of Health (NIH): http://www.nih.gov/

National Center for Biotechnology Information Website (NCBI): http://www.ncbi.nlm.nih.gov/

Python Website: http://python.og

Tavani, Herman (2005), Ethics, Computing, and Genomics, Jones and Bartlett Publishers

Tjaden, Brian (2007), "A multidisciplinary course in computational biology." Journal of Computing Sciences in Colleges, Volume 22, Issue 6, Pages: 80 – 87

Toth, Charles, Connelly, Richard (2006), "A bioinformatics experience course." Journal of Computing Sciences in Colleges, Volume 21 , Issue 6, Pages: 100 – 107

Zhang, Mingrui, Lin, Chi-Cheng, Olsen Gayle, Beck, Barbara (2007), "A bioinformatics track with outreach components." Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education, Pages: 186 - 190

APPENDIX

Topic	Average Score of the Level of Learning Number of Students Answered is 6
1. Introductory Python and ability to design simple Python programs	3.7
2. Advanced Python topics: functions, loops, if-else statements, string manipulations, lists, and list manipulations	3.5
3. Designing complex Python programs using advanced Python features	3.3
4. Understanding the concept of sequence alignment: global, local, semi-global, multiple sequence alignment	3.2
5. Understanding dynamic programming algorithmic technique	3.7
6. Understanding Exhaustive Search (brute force) algorithmic technique	4
7. Understanding Needleman-Wunsch algorithm and be able to trace the algorithm to produce the final result	3.8
8. Understanding Smith-Waterman algorithm and be able to trace the algorithm to produce the final result	3.8
9. The ability to work independently on the research – based project applying computer science and biology knowledge to solve problems	4.3
10. Understanding how to use BLAST tool and to read the results of BLAST	4.2
11. Using sequence alignments to understand relatedness among species	3.8
12. Using sequence alignments forensically (HIV experiment)	4.2
13. Understanding how microarrays are used to identify gene changes in disease	3.3
14. Understanding the flow of information from DNA to protein	3.3
15. Using computer simulations to test hypotheses about disease spread	3.8
16. The ability to read a research paper in the Ethics, Computing and Genomics	3.8
17. The ability to communicate effectively through the participation in the Ethics, Computing and Genomics project	4
18. The ability to create an informative power point presentation to present the results of the Ethics, Computing and Genomics project	4.3
19. The ability to learn the topic by yourself and the ability to present results of learning in clear way	3.8

Table 1: The results of the students post survey – list of topics with the average ratings for level of learning