

MIT AFRICA INTERNET TECHNOLOGY INITIATIVE

SUMMER 2003 PROJECT REPORT

ETHIOPIA ■ GHANA ■ KENYA

GHANA

We discuss preparations for and the actual implementation of this year's summer teaching program in Ghana under the two broad categories of curriculum and logistics. The curriculum section describes the process of preparing teaching material, planning and implementing our teaching strategy. The logistics section talks about pre-departure organizational matters, and the daily mechanics of teaching and culturally experiencing Ghana. Each section includes a summary of the major challenges faced and recommendations for remedying these problems in future programs.

CURRICULUM

Overview

The AITI course in Accra, Ghana was taught at the University of Ghana and in two high schools, Presec and Achimota. We taught over a span of six weeks, two days of which were holidays, for a total of 28 teaching days.

In the high schools, our objective was to introduce the students to software design and programming and inspire them to pursue these interests in the future. At the university level, we aimed to reinforce these software concepts, teach good programming methodology, and motivate students to take a more real-world problem-solving approach to programming through community-based software projects.

In addition to these primary objectives, we also hoped to encourage the use of open source software and programming tools in Africa. In accordance with this goal, we designed our curriculum around freely available technologies and software, namely Java 1.4 Standard Edition (J2SE), Sun ONE Studio, Apache Tomcat Webserver, and MySQL database.

Plan

Planning for this year's curriculum started mid-spring term. A curriculum committee headed by Pelei Fan took charge of revising and updating all relevant AITI teaching material from previous years and developing syllabi and material for the two new modules: Unix and JSP.

The committee developed two distinct but similar syllabi, one for the University and one for the high schools, to reflect their difference in ability level. The university syllabus began with 22 Java lectures, which covered all the Java fundamentals, including some advanced topics such as the I/O library, searching, sorting, and hashing. It also included a short series of lectures on Swing, a Java windowing API. After Java, the university course was to continue with a short module on UNIX, followed by a series of lectures on building Web applications with Java Server Pages (JSP), which we planned to apply to a community-oriented group project. Our plan for the high schools was to teach an introductory Java curriculum—Java lectures 1 through 16—and move at a slower pace than the university.

The UNIX component of the curriculum, as designed, consisted of a demonstration of an installation of RedHat 9.1 and a series of lectures on fundamental UNIX concepts, such as privileges, accounts, and pipes. The purpose of teaching UNIX was to present the students with a free alternative to the expensive Microsoft Windows platform, in the hope that it could in turn be adopted by system administrators and other motivated individuals at the university. To facilitate this, we also planned to distribute RedHat OS CDs to interested university students and system administrators.

Both syllabi were designed under the assumption that our students could demonstrate basic computer literacy. That is, we expected them to be able to operate a mouse and keyboard and navigate through a computer directory structure to find files. Beyond this, however, we anticipated the skill levels and programming experience of students to vary greatly.

When the curriculum committee finalized the syllabus and lectures, they were handed over to the members of the newly selected teams. Each team continued to revise the material in accordance with their personal teaching styles. The teams met weekly to discuss the changes to the lecture materials, as well as hash out trip logistics. Among other things, we assigned each lecture in the syllabus to two people for review and revision. The lectures were assigned based on personal preferences and skill level.

Execution

Our team arrived one week prior to the scheduled program commencement date to get settled and to install the required programming tools and environments onto the schools' computers. At Presec, there were approximately 20 available computers which were networked, which allowed for relatively easy setup and distribution of the necessary files. However, the students did not have access to the lab outside of class time, which meant we had to devote valuable class time to homework assignments that required computer work.

At Achimota, the computers were not networked and varied considerably in their make and model, which made it difficult to setup and maintain a uniform development environment across all their machines. Unlike Presec, the Achimota students had limited access to the lab outside of class hours, which gave the more motivated students the opportunity to fully complete, double-check, and generally improve the quality of their homework assignments. The facilities were best at the university, where we were able to get most of the computers setup well in advance of their use.

We had a busy daily schedule. Every morning, all nine of us went together to the university, where usually two of us conducted lectures and labs. What did the other seven do in the meantime? While it would have been nice for the lucky seven to take time off to enjoy the beautiful beaches and tourist spots, that was not the case. The remaining teachers were busy walking around the classroom, answering questions and helping out students out with their labs.

After the university, we split up into two groups before leaving for the high schools, one group to teach at Achimota, the other at Presec. How we divided ourselves differed from day to day. This allowed us to all to get to know the students from both schools, as well as to give the students a variety of teaching methods and styles to give them the best possible learning experience.

Because the high school classes were so large compared to the number of available computers and the size of the classrooms, we found it best to divide the classes into two groups, group A and group B. During the first hour of the two-hour class, group A would receive a lecture, while group B was working on a lab from their previous lecture. For the second hour, the groups switched--B in lecture, A in lab. And because only one of us would lecture at a time, dividing the students into groups increased the teacher-to-student ratio for labs, where hands-on assistance proved most critical.

As expected, we made frequent changes to the lectures and syllabus as the course progressed. For example, we decided to devote time to a quiz review prior to the first quiz, and quiz reviews became a regular occurrence after that. We also ended up removing a few of the final Java lectures, swapping others, and adding a lecture on the Collections API. Plus, some lectures took us two days to cover in their entirety.

Taken together, all of these factors required us to recalculate the course for the upcoming days on a daily basis. As a result, the curriculum we followed differed in many ways from the curriculum we had planned. In the end, however, we felt the course we taught was far superior to the one that was planned, and we believe next year's teachers will benefit greatly from our improvements.

Our continually changing curriculum could also be attributed to our inability to accurately predict the skill level and experience of the students we would be teaching. Perhaps our initial expectations were too high, as many students did not perform as well as we hoped in the first exams and problem sets. It was about the end of the first week before we were able to clearly gauge our students' abilities.

Some of us felt that the low performance of some (not all) at the university could be due more to a lack of effort and interest, than genuine ability. We didn't encounter this problem as much at the high school, perhaps because uninterested students could more easily drop the class without penalty. But because the university students paid for their class, those who found themselves quickly uninterested may have felt obligated to stick around. At times, only a few students who required a lot of assistance in lab would delay the entire class from moving on. We did not take any action to resolve this situation, but some felt that our time would have been better spent with the students that were prepared to do the work necessary to keep up with the class. There was also a surprising amount of cheating on problem sets, which some felt we dealt with too lightly.

Some also felt we were partially to blame for the problem set copying by making the problem sets too difficult, for even the most talented students to complete in their entirety. Some problem sets were exceedingly long, some would test the same concept multiple times—resulting in multiple deductions for not fully comprehending a single concept, and because the curriculum changed frequently, some were not revised to reflect the changing content of the course, so occasionally students were tested on content we had yet to teach or decided not to teach at all. Some of the quizzes suffered from the same problems.

So as to not paint an unfair picture of the students' performance, let us clarify that the vast majority of students did not cheat in any way and exhibited sincere effort and a strong desire to learn. Overall, the students performed quite well, especially given the fact that only a few had access to a computer outside of lab and most had no appropriate text book or reference. Though there was certainly room for improvement, all in all, we were pleased with their achievement.

Despite the above-mentioned setbacks, we did succeed at teaching a robust and challenging Java curriculum. At the university, we finished nearly the entire Java curriculum we set forth to complete and even added a lecture on the Collections API, which we hadn't planned. Unfortunately, due to time constraints, the UNIX module never came to pass. But we did complete the introductory JSP module and gave the students time and assistance to make progress on a community-oriented project.

The final project was the most rewarding part of the course from our perspective. We observed the enthusiastic students invest a lot of effort as applying their knowledge to real-world projects excited them. The projects included AIDS awareness web sites, university book swaps, online Java courses, online student record databases, and a Web-based data compression service.

CURRICULUM CHALLENGES & SUGGESTED CHANGES AND IMPROVEMENTS

We learned a lot while teaching this course, and there are a number of ways we feel it could be improved for coming years. We've divided these suggested changes into 'Course Suggestions,' which are proposals for how we could improve the course generally, and 'Lecture Suggestions,' which include specific changes

to lectures and syllabus. They are listed below in no particular order.

Course Suggestions:

1. More reference material

While the in-class materials were taught very effectively, it is difficult to pay attention everyday for a solid two to three hours. It is very easy to zone out or miss very important points. For future years, we suggest preparing reference material for the students to take home and study. Studying from personal notes for a test is very effective, but only if the notes are complete. Reference materials would also help students with excused absences.

2. Clearly defined cheating and collaboration policy

We suggest composing a clearly defined policy on what constitutes ‘cheating’ and what falls under ‘permissible collaboration.’ This policy should be well articulated, possibly in the form of a hard copy handout, on the first day of the course.

3. Group projects

While we did not want the students to cheat, we did hope that they would work together to share ideas and solve problems together. We believe peer collaboration is a beneficial learning tool that we could have employed more often during the course.

4. More exciting projects for high school students

We often found the high school students to be bored with the given lab assignments. Rather than teaching boring tasks such as computing the factorial of a number, we suggest reworking labs, wherever possible, to involve programming some sort of fun game or fun concept. For instance, we captured their interest as soon as we assigned the task of designing a Racecar object, capable of racing other racecars.

5. Make sure every teacher knows all the material

Towards the end of the course, when the subject matter became more advanced, only the more experienced programmers in our group conducted the lectures and labs. But unfortunately, we did not ensure everyone was prepared to help students with all the labs and all the material. For future teaching groups, we suggest that the night before each class, the lecturer assigned for the next day deliver a practice lecture to the other members of his or her team. This practice will benefit the lecturer and the learning experience will benefit the rest of the team.

6. Fairer problem sets and quizzes

As mentioned above, the problem sets and quizzes were often too long, tested the same concept multiple times, and occasionally tested things they hadn’t learned. We suggest the problems sets and quizzes be kept short and to the point, testing exactly those things we expect them to know. As a check, we suggest one or more teachers do the problem set or take the quiz the night before it is due. This will ensure (1) that the assignment does not test concepts they haven’t learned and (2) that the assignment takes a proper amount of time to complete. Expect the students to take at least twice as long to complete the assignment as the teachers.

7. At the university, more time for the community project

As stated, the community project was probably the most rewarding and most fun part of the University curriculum. Though the students did have some time to make headway on their projects, they were not given enough time to make serious progress towards a full-fledged project. By applying some of the suggestions listed in this section, and possibly others, hopefully the course will proceed more quickly and smoothly in the future, allowing a greater amount of time for the community project.

8. In high schools, more time with Swing

In line with the suggestion for more exciting high school projects, we suggest a greater amount of time devoted to Swing in the high schools. Throughout much of the course, many of the high school students were discouraged by the lack of visual interaction with the programs they were writing. They continually asked us when they were going to learn graphics, but unfortunately we were left cramming Swing into a few days at the very end of the course. Nevertheless, the feedback we received was that they enjoyed those lectures the most. We suggest trying to leave as much as a week, if possible, at the end of the term to learn Swing and give the students time to code a fully graphical program.

9. Proper coding conventions

We expended a lot of time and effort fixing mundane typographical errors in students' code, including continual cases of mismatched braces. We believe many of these errors could have been pre-empted had students been following proper indentation conventions in their code, and consequently saved us a considerable amount of time in the long run. Proper coding conventions help to make code more readable, easier to debug, and generally more understandable. Unfortunately, we didn't always follow the conventions ourselves on our slides and materials, but we suggest they be so revised.

Lecture Suggestions:

1. Simplified introductory material.

The first few lectures attempted to be too complete in their coverage of Java fundamentals. For instance, when covering variable types, we taught about boolean, int, short, long, byte, float, double, char, and String. However outside of quizzes, we never used short, long, or float, because their uses are relatively rare. And byte was used only in the context of learning the I/O library. We propose simplifying the beginning lectures by leaving out short, long, and float entirely and postponing the introduction of byte until the lecture on I/O streams. Any other unnecessary basics of Java should be similarly eliminated from the first lectures as well, so as to ensure as simple and straightforward a beginning to the course as possible.

2. Braces are not a control structure

In the lecture on 'Control Structures', the initial sections on 'braces' should be eliminated. It's not a control structure and makes no sense in that section. Later on, in the 'Scope and Access' lecture, it's important to emphasize that curly braces always establish a new variable scope.

3. Restructure the 'Arrays and Vectors' and 'Collections API' lectures

The `java.util` package, also known as the 'Collection API,' is an integral part of the Java library, and it was very important for us to add the lecture on this topic. However, this lecture was too long and jammed packed full of information. It included everything from the entire Collections API, to the performance of different list data structures, to the use of Iterators, and even included a discussion of the equals method. In contrast, the 'Arrays and Vectors' lecture was short and was mostly review.

We suggest replacing the lecture on 'Arrays and Vectors' with a lecture entitled 'Lists.' The 'Lists' lecture can use the slides that cover Lists and Iterators found in the lecture on the 'Collections API'. This new lecture can also include the discussion of the differences between Arrays and Vectors found in the 'Arrays and Vectors' lecture, though each reference to Vector should be replaced with a reference to ArrayList. In sum, the new 'Lists' lecture should introduce ArrayList and LinkedList, discuss how an ArrayList differs from an Array, describe the performance differences between ArrayList and LinkedList, and show how to iterate through lists with Iterators. The new 'Collections API' lecture, therefore, can now be shortened by skipping the discussion of Lists and Iterators.

Finally, the 'Arrays and Vectors' lecture noted that one advantage of Vectors over arrays is that Vectors are polymorphic (can contain objects of different types) but arrays are not. This is incorrect. An array of type `Object[]` is equally as polymorphic as a Vector. On the contrary, it is an advantage of arrays that they can be restricted to be homogenous (such as creating an array of type `String[]`) whereas Vectors cannot. This point should be clarified during the restructuring of this lecture.

LOGISTICS

Pre-departure Logistics

Communication with the two high schools, Achimota and Presec, began in January via email, as this was the fastest and most convenient way of communicating. In February, the Computer Science Department of the University of Ghana was invited to be a part of the program. All 3 educational institutions were positive and very enthusiastic.

In February, the application form for MIT students interested in the program was available online. After the March 31, 2003 deadline, the selection committee members went through over 110 applications and interviewed 40 of the applicants. Initially, 7 MIT students were to go to both Ghana and Kenya with 5 going to Ethiopia. Due to the threat to British Airways flights into Kenya and the possibility of a terrorist attack, the Kenyan program was cancelled and the Ghana team grew to 9 members.

The team leader was responsible for arranging travel visas, housing and transportation. All the team members were issued visas to Ghana before the start of the program. After going through a number of living options, Monroe Guest House was decided on. Transportation options were to be researched by the team leader and decided on when the team arrived in Ghana. Transportation allowance was included in each members travel advance.

Housing

The underlying idea behind housing was to find a safe and comfortable place to live within a reasonable budget. In light of recent terrorist activities around the world, we wanted a somewhat isolated facility¹, relatively far from the centre of Accra yet close to the educational institutions where we were scheduled to teach. This year, the team occupied a fully furnished 5-bedroom house, Monroe Guesthouse, with a living room, dining and kitchen facilities. 8 team members paired up to occupy four of the bedrooms, while the last member singly occupied the fifth bedroom. The housemaids and front desk at the guesthouse was helpful and friendly. For food, we prepared our own breakfast and bought lunch at the University campus. A caterer was hired to provide dinner Monday to Friday nights.

Overall, we were content with the housing arrangements as each room was air-conditioned and had a television, a little fridge and a bathroom. There were a few minor problems however. Hot showers were not consistent and the water pressure was pretty low at times. Also there were occasional water shortages. These shortages were however fixed promptly by the management. These challenges however just added to the team's experience of Ghana.

Transportation

The trip from Adenta to Legon was approximately 10 minutes, but usually took about 25 minutes due to rush hour traffic. For our daily trips to the schools, we contracted the services of Royal Shuttle Services, a private mini van operator. A van picked the team up at 7:45am and dropped us at the University of Ghana, where we began teaching at 8:30 am. At 2pm, the van came back to pick the team up and drop half of us at Achimota and then remaining half at Presec. Members of the group were quite free to go to either school. This arrangement worked out very well. At 5 pm, the van picked the teams from Achimota and Presec respectively and drove us back to our guesthouse.

To get around town outside of teaching hours, we used public transportation (taxis, busses and trotros (privately owned minivans)). Taxis were readily available; whatever time of day (or night) we were able to find economical and quick transportation in the form of taxis. Trotros on the other hand, ran from the approximately between the hours of 6 am to 9 pm. Trotros were much cheaper to use than taxis, but entailed a little bit of walking to get to designated bus stops. Riding the trotros was quite a cultural experience as we got to sit and talk with the locals. Overall, getting around the city was easy as well as safe.

Communication & Teaching Equipment

Last year's team took 2 laptops donated to AITI, while a team member brought her personal laptop. This was a major setback as the 3 laptops had to be shared amongst 6 members and only one laptop had a CD writer. The members found it difficult to write lecture slides at their convenience and it took a huge amount of time to make the electronic copies of lecture notes for the students. To transfer files between the 3 computers, one team member hand made a cross over Ethernet cable by rewiring a straight-through cable. The team also had one LCD projector to present lecture material.

This year, the MIT Laptop Computing Project generously supplied each team member with a laptop computer. Each computer was equipped with a CD writer, a floppy drive and a DVD drive. AITI also purchased a wireless hub to make file transfer easier. We did not face the problems that were encountered last year. AITI acquired a second LCD projector this year to facilitate running the program simultaneously at Achimota and Presec.

Generally people found it easy to communicate with family and friends back home because calling cards

¹ Ghana is a very safe country, with no history of terrorist activity

of the Ghana Telecom Company were readily available, including at the front desk of our guesthouse. Team members were often seen sending postcards, and writing emails at home to send via Internet cafes. The computer science department offered us free Internet usage at one of the many Internet cafes on the university campus.

LOGISTICAL CHALLENGES & FUTURE RECOMMENDATIONS

- ☐ Students were unfamiliar with teaching style: Many of the students had problems grasping the material taught because they were not used to being taught with PowerPoint presentations. They felt that lectures were rather fast and they are unable to take notes. Many asked for lecture notes before the class so that they could familiarize themselves with the material before class. Even though last year's lecture notes were available, many of us were not satisfied with them and we ended up rewriting all the lecture notes.
- 💡 It would be great if the lecture material were ready before the team sets off to Ghana. The lecture material would then be given to the students at the beginning of the program instead of at the end. If the students were able to go through the material before class, they would be able to grasp the concepts better in lecture.
- ☐ The team was disappointed with the general attitude and qualifications of the high school students: AITI does not specify any stringent requirements to the schools concerning the choice of students and leaves that up to the discretion of the school. However, it is obvious that the school chooses students who are interested in the program. The program spans over a period of 6 weeks and is very intensive. As much as we would like to teach anyone who is interested, due to time restrictions, it is best to teach students who are bright enough to absorb the material as quickly as we teach it. We were definitely slowed down by students who were finding the program too fast.
- 💡 In future, AITI should come up with a more formal set of guidelines to be used by the schools in selecting interested students. The schools should be asked to choose smart, serious students to participate in the program.

Secondly, many of the students viewed the program as an extracurricular activity and as such did not take it as seriously as they should have.

We suggest that AITI develop a formal set of criteria for assessing student understanding of material taught and based on this perhaps issue certificates of participation and excellence as an incentive for students to treat the program with a little more seriousness.

- ☐ Program date was too close to students' final exam date: At Achimota School, more than half the original number of students dropped out of the program. Majority of the students felt that the program was too close to their end of term exams and much more time consuming than expected. Students who remained in the program also complained about how close the program was to their finals and we believe that affected the amount of time they gave to the program.
- 💡 AITI should consider going to Ghana much earlier, possibly at the end of May. This will also be great for the University part of the program as the University will be in session and we would have more University students participating.

- 📅 Unreliable and pricy printing facilities: In general it was a great hustle and expensive to print tests and other handouts in Ghana. Much of our grading and testing woes were due to unreliable printing.
- 💡 Two suggestions for dealing with this problem are suggested: 1) Invest in a small portable printer/copier for each country. 2) Explore the possibility of getting old Athena printers donated and shipped to participating countries. Doing so will save hours of administrative time and effort. The second option is however preferred because Athena printers are capable of high volume jobs.
- 📅 Problem sets difficult to grade: While a lot of effort was put into revising lecture slides before the teams departed, no such work was done on accompanying problem sets.
- 💡 Revise all homework and tests to be more grader friendly. This would be a lot of work, but would save even more work in the long run. Having homework and tests include more multiple choice questions, and less complete code writing would help increase the homework turnaround rate. It is difficult for teachers to spend every night grading after teaching all day. Homework and exams should be designed to be graded within an 8 to 10 hour working day per teacher.