A Computer Science Educational Program for Establishing an Entry Point into the Computing Community of Practice

Bruria Haberman
Holon Institute of Technology, and The Davison Institute of Science Education, The Weizmann Institute of Science, Rehovot, Israel

Cecile Yehezkel
The Davison Institute of Science Education, The Weizmann Institute of Science, Rehovot, Israel

bruria.haberman@weizmann.ac.il cecile.yehezkel@weizmann.ac.il

Executive Summary

The rapid evolvement of the computing domain has posed challenges in attempting to bridge the gap between school and the contemporary world of computing, which is related to content, learning culture, and professional norms. We believe that the interaction of high-school students who major in computer science or software engineering with leading representatives of the computing community of practice may motivate them to pursue their studies further or pursue a career in the field. Accordingly, our program aims at exposing talented high-school students "directly by leading experts" to state-of-the-art computing research, advanced technologies, software engineering methodologies, and professional norms. The interaction between the students and the experts, who actually become role models for the students, occurs at two levels: (a) during enrichment plenary meetings, and (b) through one-to-one interaction in which students develop software projects under the apprenticeship-based supervision of professionals from the computing community of practice. In the last four years, six hundred students participated in enrichment activities; 86 of these students accomplished high-level software projects under the supervision of experts.

A long-term formative evaluation of the program has been conducted regarding: (1) students' attitudes towards the "different-from-school" style of learning that characterizes the program, and (2) students' performance in developing projects. In this paper, we specifically discuss the contribution of resources that students used for various phases of the project development activity. We found that the following categories of resources were employed by the students: self-learning, mentors, bibliographic resources (the Web, professional articles, professional books), school resources (a school teacher, school learning, and materials), and other human resources (i.e., a classmate the student's age, a family member, a grown-up acquaintance). Importantly, the findings indicated that during the entire development process the students exhibited self-efficacy, since they relied more on themselves than on other resources. Interestingly, during the entire development process, the web was perceived as the most significant bibliographic resource. Specifically, regarding the need to achieve

Editor: Grandon Gill

adequate acquaintance with the theoretical knowledge that was required, self-studying and the web were perceived as the most significant resources, which may imply that the mentors' guidance inspired the students' self-inquiry and self-study. However, during the problem-solving activities, students relied more on the mentors than on bibliographic resources.

Based on the study findings, we concluded that professional experts who supervise students in project development may motivate them to acquire in-depth knowledge in computing, promote creativity, as well as enhance self-learning and inquiry ability. In addition, the interaction with role models may contribute to establishing professional norms.

We hope that further implementation of the program, along with recruitment of more representative experts from academia and hi-tech industry, will promote a culture of learning and work befitting the dynamic world of industrial computing, thus providing the students with an entry point into the computing community of practice.

Keywords: self-learning, out-of-school learning, project-based learning, mentoring, role models.

Introduction

During the last two decades, a program in computer science (CS) (Gal-Ezer, Beeri, Harel, & Yehudai, 1995) and a program in software engineering (SE) (The Ministry of Education, 2004) especially designed for the high-school level have been in operation in Israel. The aim of the computer science program is to expose the students to a fundamental scientific domain whose principles are characteristic of algorithmic thinking. The software engineering program consists of the following components: (a) an elective topic in natural sciences, (b) computer science, and (c) an elective advanced specialized topic. Its aim is to expose the students to computational thinking (Wing, 2006) as well as to system-level perception (Gal-Ezer & Zeldes, 2000). Both programs have evolved over the years in accordance with changes in the discipline of computing; however, a gap still exists between the school programs and the "real world" of computing related to: content, learning culture, and the professional norms governing software development processes. We believe that in order to motivate students to seek expertise in the field, it is important to expose them to up-to-date computing research and development (R&D) processes both in academia and in high-tech industries, and to raise awareness of actual common professional issues that members of the CS/SE community of practice cope with. These considerations motivated us to initiate the Computer Science, Academia and Industry, an extra-curricular enrichment program at the Davidson Institute of Science Education. Similar approaches, which consider the importance of enrichment programs in motivating students to learn computer science, are presented in Sherrell and Shiva (2005) and Sims-Knight and Upchurch (1993).

In a previous paper we described how our program was designed to bridge the gap between school and the "real world" of computing (Yehezkel & Haberman, 2006). In this paper we focus particularly on the benefits of students developing projects under the apprenticeship-based supervision of professional instructors.

The Enrichment Program

Background and Motivation

The gap between school education and the "real world" of computing is especially related to content, learning culture, and professional norms.

The fundamentals and the core technologies that are introduced in school form the basis for understanding the field of computing; however, they rarely resemble the state-of-the-art computing research and development as well as the new, evolving directions in the field. Regarding bridging

the gap between learning in school and real-world situations, Ben-Ari concluded that, most likely, decontextualized schooling will continue to be a fundamental method of computer science education (Ben-Ari, 2004), since, in particular, in this high-technology world, "a newcomer must have a significant amount of basic and background knowledge before entering into meaningful participation in technological communities of practice" (Ben-Ari, 2003). Based on these considerations, we believe that *students should learn fundamentals along with getting acquainted with enrichment advanced topics*.

The traditional style of teaching/learning in school is usually designed so that students can acquire explicit knowledge based on a thorough understanding of the topic learned. However, this approach alone might fail to educate the students to become self-learners who are capable of navigating in the rapidly growing world of knowledge (Computing Research Association, 2005; Long & Ehrmann, 2005; Passig, 2001). Hence, we suggest that *students be taught to employ a breadth-oriented, "tasting"-based learning style*.

The Software Engineering 2004 Curriculum states that incorporating real-world elements into the curriculum is necessary to enable effective learning of software engineering skills and concepts (ACM/IEEE Joint Task Force on Computing Curricula [ACM/IEEE], 2004). Educators should "devote time to analyzing what actually happens in real communities of practice, and then to create learning activities that simulate such tasks as well as possible within the constraints of a school." However, according to Ben-Ari (2003), learning activities should be "influenced by the nature of the activities that occur in the community of practice that students will encounter in the future" rather than by its actual detailed practices. Students should be given tasks that could deepen their sense of meaningful participation in the community, such as working with given complex programs instead of just writing "toy programs" (Ben-Ari, 2004). For example, students should receive experience in examining, finding, and correcting flaws early, as well as in filling in missing parts of given large artifacts "with the intent of showing them the complexity of the field they are about to study" (Bergin, 2005). Accordingly, we believe that *students should be activated by representatives of the computing community of practice in small-scale activities that simulate "real-world" situations*.

Educators have long noted the importance of teaching software design skills to high-school computer science students (Gal-Ezer et al., 1995; Gal-Ezer & Zeldes, 2000; Sims-Knight & Upchurch, 1993). According to the *Situating Constructionism* learning theory (suggested by Papert & Harel, 1991) meaningful learning-by-making occurs "in a context where the learner is consciously engaged in constructing a public entity". For example, project-based learning and software development assignments, performed by students in meaningful contexts while applying *Systematic Inventive Thinking* methods (Helfman & Eylon, 2003), may facilitate meaningful learning as well as contribute to making computing more appealing. The academic CS community believes that the role of projects in the curriculum is of major importance, since it is a means of effective learning, and it also is a way of demonstrating the student's mastery of skills appropriate to professional practice (Fincher, Petre, & Clark, 2001; Holcombe, Stratton, Fincher, & Griffiths, 1998). Project development enables students to construct knowledge and to enhance cognitive and reflective skills; it also encourages students to become creative, innovative (Denning & McGetrtrick, 2005), and independent learners. It addition, it enables students to encounter real-life experience as a project developer (ACM/IEEE, 2004; Bracken, 2003). School projects enable students to experience software design and development processes and to acquire a system-based perception. However, the school educational milieu is not capable of demonstrating to the students the "real world" professional norms of software development, since the development settings and processes in school do not resemble actual research and development industrial processes. Although the teachers occasionally participate in workshops to develop their knowledge and professional skills, they still do not belong to and do not represent the computing

community of practice. Hence, the students' products are rarely applicable to real-world situations. Accordingly, we believe that *students should develop projects under the supervision of professionals who represent the computing community of practice.*

The Underlying Model

Our enrichment program aims at bridging the gap between the fundamentals learned in school and the "real computing world". Its mains goals are as follows: (a) to expose young students to an up-to-date field of computing research and development; and (b) to establish a learning culture that will provide the students with an entry point into the computing community of practice; and (c) to motivate novices to seek expertise in the field. The model underlying our program is based on the following guiding principles:

Blending formal (in-school) and informal (out-of-school) learning

"It is increasingly apparent that informal and lifelong learning is the key solution to equipping people with the evolving knowledge and skills that will be needed to adapt to the continuously changing nature of society" (Long & Ehrmann, 2005). Computing is a dynamic, rapidly evolving discipline. Hence, students should be taught to employ a *breadth-oriented "tasting"-based learning style*, according to which their initial exposure to an unfamiliar topic will be accomplished by exposing them only to its essence (i.e., with the main high-level-abstract-related ideas). In other words, a complete understanding, including knowing the concrete details and mastering procedural aspects, should not be considered as the immediate aim of an initial exposure to a new topic. We believe that a program based on blending traditional (in-school) learning of CS fundamentals and enrichment (out-of-school) program related to state-of-the-art computing will provide the students with a suitable entry point into the computing community of practice.

Role models

Role models across the science spectrum have a doubly important role to play – both to communicate information and to inspire others to understand and appreciate the work of the scientist and engineer today (National Academy of Sciences, National Academy of Engineering & Institute of Medicine, 1997). We believe that the encounter between novices and representatives of the computing community of practice is very important.

One specific example that motivated us to use role models (National Academy of Sciences, National Academy of Engineering & Institute of Medicine, 1997) in our program was the story of Prof. Ehud Shapiro, a distinguished computer science scientist, who found it important to indicate in his short bio that: "The guiding light for [his] scientific endeavors was the philosophy of science of Karl Popper, with which he became acquainted through a high-school project supervised by Moshe Kroy from…Tel Aviv University" (Shapiro, n.d.).

Diversity

We believe that diversity should be a central motive in encouraging self-learning education, meaning that students should be guided to combine initial breadth-first exposure to new topics with in-depth knowledge of chosen topics in various contexts (e.g., developing a project in a specific domain).

The Setting

Our program interweaves enrichment meetings and the development of software projects in a setting that simulates a "real world" environment. The program is planned to be conducted in two stages.

Stage A

The first stage consists of a 6-month enrichment workshop. Each monthly (after school) meeting consists of 2 lectures by leading representatives of the CS/SE academia and industry, as well as related class activities. (See Table 2 in next section for details.) Advanced topics are introduced, industry's professional norms are discussed, and advanced technologies and methodologies are demonstrated. In addition, "visiting the industry" tours are conducted. The program's activities are supported by a dedicated site that provides an opportunity to communicate with students between meetings. Toward the end of this stage, a small group of students are selected to continue to the second stage of the program.

Stage B

The second stage is totally devoted to the development of software projects under the apprenticeship-based supervision of professional instructors-scientists from academia and practitioners from hi-tech industry. Some of the students actually participate in "real" industry projects, thus solving "real-world" problems for a real client; others utilize advanced industrial development tools. The projects are developed by individuals or by pairs. The school teachers are actively involved in guiding and supporting the students throughout the entire development process. The challenge is to create productive cooperation among all four players: (a) the student, (b) the mentor, (c) the academic management team of the enrichment program, and (d) the school teacher. This challenge has motivated the research described in the present paper.

The duration of the project development process is 8-9 months. During that period the students are requested to submit sub-products (e.g., specification documents and a mid-term report) according to a given time table. Basically, the students (individuals/pairs) develop the project separately; however, several meetings for the whole Stage-B group are conducted. The meetings are dedicated to software development issues that are presented and discussed by experts. In addition, the students report to the group about their progress.

Diversity of population

Selection of attendees is performed toward each stage, and a regional after-school class is composed including excellent students from different schools throughout the country. The first stage of the program is designed for 11th grade excellent students who major in CS/SE. We believe that students who demonstrate excellence relative to their classmates should be appreciated and invited to attend our program; hence, we do not interfere with the students' selection process in the first stage of the program. Toward the beginning of an academic year, we suggest to high-school principals and CS teachers that they select a group of up to 10 excellent students (meaning that they are highly motivated and high achievers) that would like to attend the first stage of our program. As a result, the student population attending the first stage may be very diverse with respect to the students' socioeconomic, cultural, religious, and curricular background, a situation that must be carefully dealt with when planning the agenda of the enrichment workshop and its setting.

The second stage of the program is designed for a small group of graduates of the first stage who are interested in developing "real" software projects under the supervision of experts from academia or the high-tech industry. In our opinion, this stage is suitable only for the "cream of the crop" students who exhibit the following characteristics: high motivation, creativity, self-learning and inquiry ability, persistence, consistency, and the ability to follow a timetable. Accordingly, the selection of students for the second stage is more rigorous than the first stage, and it is based on the following criteria: (a) the teachers' recommendation, based on their acquaintance with the students and according to students' achievements and the above characteristics; (b) the applicant's resume, which should provide information about his CS knowledge, programming experience,

knowledge of programming languages, participation in other relevant enrichment programs, and experience in developing software projects; and (c) the applicant's ability to persuade us that he is seriously interested in developing the project, and that he is capable of successfully accomplishing the development and can submit a working product (according to the specifications). The mentors are representatives of the following streams of the computing community of practice: (a) faculty members in CS/SE academic departments, (b) M.Sc and Ph.D students, and (c) practitioners in hi-tech industry.

Student-mentor matching

Student-mentor matching is the key for successful development of a project. To establish optimal matching, we developed a model (schematically illustrated in Figure 1). An Employment Fair meeting is conducted, where, in a plenum session, the mentors present to the students a variety of project subjects for which they can serve as advisors. After the presentation, a face-to-face mentor-student interaction takes place where students are asked to present to mentors their "CV-like" applicant's resume. The process ends when all possible interactions take place. During the interaction, the students ask the mentors questions about the suggested projects and examine whether the topics seem attractive and can be dealt with and whether they want the mentor to guide them. At the same time, the mentors implicitly investigate whether the students are qualified enough to develop the project that they suggested. Next, the students are required to submit a list of projects in order of their preference, and the mentors are asked to choose students according to their assessment. Finally, the managers of the program perform the mentor-student pair matching.

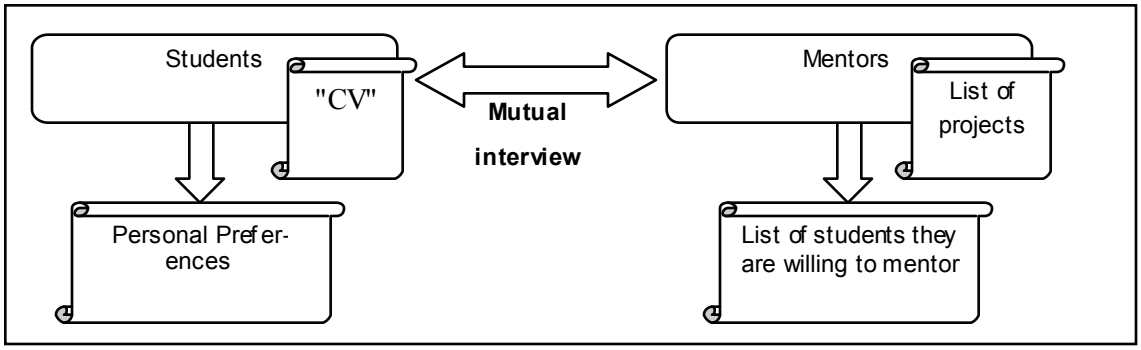


Figure 1: The Employment Fair Simulation

Three Years of Experience

Three full cycles of the program were conducted so far (illustrated in Table 1). The fourth cycle started at November 2007 with an audience of 210 students.

The Enrichment Meetings

The pilot implementation of the program began in November 2004. A group of 71 high-school students, accompanied by their teachers from 9 schools located in the area of the Davidson Institute of Science Education, attended the program. The second cycle began toward the end of 2005 with 140 students from 20 schools, some of which are located in distant areas. The geographical spread, as well as the growth of a number of attendees continued in the third round. The third group of attendees consisted of 180 students from 30 schools. Students from distinct localities throughout the country came after a full day of school-studies to participate in monthly afternoon (17:00-20:30) enrichment meetings. Each group of students is accompanied by their CS schoolteacher. Thus, both the students and the teachers are given an opportunity to enrich their knowledge by participating in the program. For many attendees it took about two hours of transporta-

tion in each direction, ending in returning home late at night. Interestingly, the student population presents an amazing socioeconomic, ethnic, and religious diversity. Students from secular and orthodox schools belonging to Jewish and Arabic communities attended plenary sessions acting as a common interest study group while (literally) sitting in a conference hall in mixed positions, and collaborated in small mixed groups in enrichment activities. Table 1 summarizes the evolution of the program in terms of participation in the first three cycles.

Table 1: The participation to the program – Three years of experience

Years		2004-2006	2005-2007	2006-2008
Stage A	# Students	71	140	180
	# Schools	9	20	30
Stage B	# Candidates	25	50	80
	# Graduates*	13	28	unknown**

* Students who succeeded to successfully finish their project.
** Stage B of the cycle that began in 2006 and will end in June 2008.

Aimed at exposing talented high-school students to state-of-the-art computing research, advanced technologies, software engineering methodologies, and professional norms, the program offers a wide-spectrum of topics and activities, as illustrated in Table 2.

Table 2: Sample of topics and activities

Topics	
• *Advanced programming paradigms* - scenario-based programming, aspect-based programming; • *Development of complex systems* - model-based development, advanced software development tools, computing in space; • *Artificial intelligence* - machine learning; neural networks, the control of motion in biological and robotic systems; • *The synergy between computer sciences and biology* – biological computers; modeling of biologic evolution; using computational models to comprehend behavior of biological systems; transmission of odor by a computer. • *Professional norms* - standards, the importance of testing and controlled reuse of code; • *Computer science educational research* - misconceptions and their implication regarding the quality of software.	
Activities	
• Construction and programming robots; • Challenging algorithmic problems; • Role-playing simulation games;	• Creative thinking in computer science; • Model-based development; • Competition in testing software;

The Project Development Activity

Two cycles of project development (Stage B) were completed and the third cycle began. Table 1 illustrates the distribution of stage B participants. So far, 41 students (13 from the first cycle; 28 from the second cycle) succeeded in submitting a final product - a working software system (according to specifications) and a report that describes the system and its development, along with

reflective self-assessment and recommendations for further development. The low percentage of students who successfully completed their project is the result of time-consuming investment in the development process. Obviously the students must carefully plan and manage the implementation of their projects in order to successfully accomplish the project development task and avoid a negative impact in learning other school subjects. The third group of 80 students recently began Stage B; it is planned that they submit their projects around June 2008. We hope that the percentage of students who successfully complete the task will increase as a result of better mentoring them how to plan and to manage their timetables. At the beginning, the students developed their projects under the supervision of 12 mentors (5 talented CS graduate students, 3 lecturers who are active researchers in the department of computer science, and 4 professionals from the high-tech industry). Since then, the group of mentors increased to some extent (N=18), and there has been a slight personnel turnover.

External examiners, appointed by the Ministry of Education, evaluated the projects, which received very high grades. The grade that students receive in their project is recorded in their standard matriculation diploma. In addition, students receive a certificate from the Davidson Institute of Science Education confirming their participation in the project.

The subjects of the students' projects (Table 3) are usually related to topics of the enrichment meetings and actually reflect the mentors' background. Most of the projects mentored by industry representatives have practical characteristics; for example, computerized homes, programming a robot, and managing a multimedia-shop. On the other hand, the projects sponsored by CS faculty and graduate students focus on theoretical or research-based subjects such as computerized graphics, image processing, automatic text categorization, modeling-based development of a control system, simulation of the theory of natural selection, and games based on learning machine theory.

Table 3: Sample of subjects of student projects

Topics of the enrichment program	Projects' subjects
Advanced programming paradigms	
Development of complex systems	System development based on modeling
Artificial intelligence	Sentence completion with learning algorithms Web-Content Management
Professional norms	Antivirus Anti-worm
CS educational research	Management of online educational material
Cryptography	RSA Encrypted Messenger
Robotics	Sun-heated water tank Home-device control Programming a robot
Computer Sciences and Biology	Disassembling and reassembling DNA.
Computer Graphics and Mathematics	3D drawing with mathematical functions 3D geographical mapping

Image processing and computerized vision	Processing hand movements for remote control
Neural networks	Recognition of characters in a picture
Signal processing	Voice signal recognition for remote control Voice identification
Game Theory	Game management shell Reversi Game Chess Game

Assessment of the Project Development Activity

Since our aim is to improve the program, we decided to accompany its implementation with an ongoing evaluation. In Yehezkel and Haberman (2006) we described a preliminary assessment of students' attitudes toward the enrichment meetings (Stage A of the program) and the "different from school" style of learning characteristics of the program. In this paper we focused on the assessment of the project development activity (Stage B).

Procedure

The students from the second cycle were asked to answer a reflective questionnaire just after submitting their final projects (N=28; N represents the number of stage B participants). The questionnaire referred to: (a) resources used during the development of the project; (b) challenges and management of the development process; and (c) benefits gained from the project development activity (and the enrichment program as a whole).

In addition, a preliminary qualitative assessment was performed to follow up the project development processes. The students were instructed to organize a portfolio that includes progress reports and intermediate products. We asked the mentors to document the mentoring process and to record interesting events. So far, we also conducted a few open interviews with teachers, students, and mentors.

In this paper we focus on students' assessment of the resources they used for developing a project. In addition we present the experience of one specific pair of student-mentors.

Students' Assessment of Resources Used

One main goal of our study was to assess the contribution of resources that students used for various phases of the project development activity (see Table 4).

Questions from the following template were posed to the students: *When performing Phase # X, to what extent (high=5, low=1) did you use the following resources?* (see Appendix).

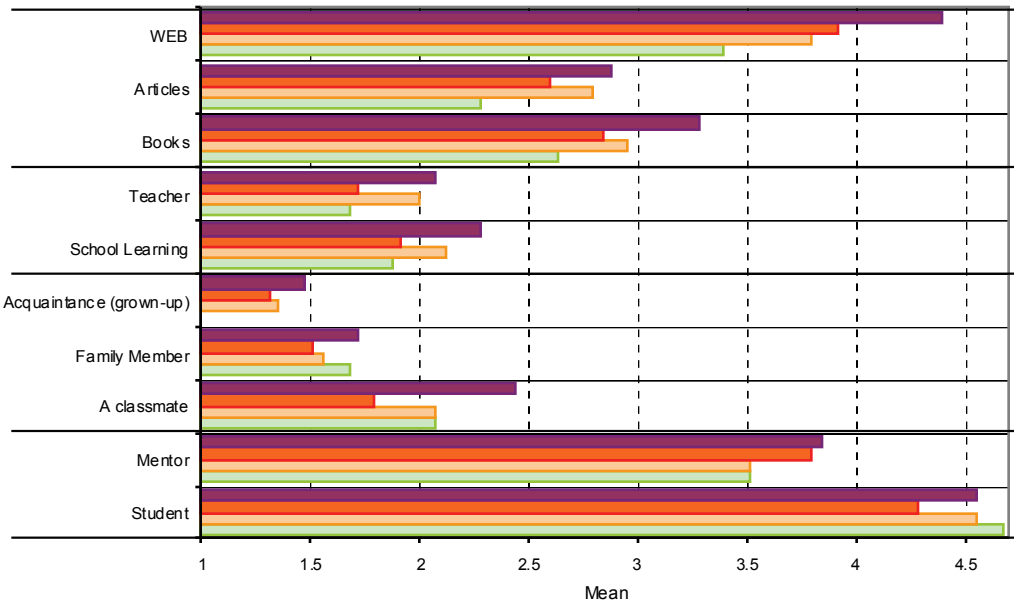
Table 4: Phases and resources used for the project development activity

Phase #	Description
Phase I	Studying the theoretical background needed for developing the project development
Phase II	Identifying the main algorithmic ideas
Phase III	Acquiring the needed technical knowledge - studying a suitable programming language and a development environment
Phase IV	Implementing the project (writing and testing the code)
Resources	
Self-studying, the mentor, the Web, professional books, professional articles, the school teacher, school learning (materials and methods), a classmate of the student's age, a family member, a grown-up acquaintance	

In addition to the contribution of the distinct resources on project development, we were interested in determining the influence of groups of resources with common characteristics. Accordingly, we grouped the resources into the following classification groups:

- ***Bibliographic Resources (B)*** – the Web, professional articles, professional books.
- ***School (Sc)*** – The school teacher, school learning (materials and methods).
- ***(Informal) Human Resources (A)*** - A classmate of the student's age, a family member, a grown-up acquaintance.
- ***Mentoring (M)*** - The mentor.
- ***Self-Studying (St)*** - The student.

Figure 2 presents (graphically and numerically) students' assessment of the resources used for project development (Likert Scale 1 (low) -5 (high)). The resources were presented on the perpendicular axis of the graph according to the group classification. Interestingly, the graph has a U shape, illustrating (even without further statistical analysis) that *Self-Studying*, *Mentoring*, and *Bibliographic Resources* were evaluated highly by the students compared with the other resources.



Phases	Web	Articles	Books	Teacher	School Learning	Acq. Growmup	Family Member	A Class-mate	Mentor	Student
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
I Theoretical background	4.4 (1.0)	2.9 (1.4)	3.3 (1.3)	2.1 (1.2)	2.3 (1.3)	1.5 (0.9)	1.7 (1.1)	2.5 (1.6)	3.8 (1.5)	4.6 (0.6)
II Identification of main ideas	3.9 (1.3)	2.6 (1.6)	2.8 (1.6)	1.7 (1.1)	2.0 (1.4)	1.3 (0.8)	1.6 (1.1)	1.9 (1.5)	3.8 (1.5)	4.3 (0.7)
III Technical Knowledge	3.8 (1.3)	2.8 (1.5)	3.0 (1.6)	2.0 (1.3)	2.1 (1.4)	1.4 (0.8)	1.6 (1.1)	2.1 (1.4)	3.5 (1.5)	4.6 (0.8)
IV Project Implementation	3.4 (1.5)	2.3 (1.6)	2.6 (1.7)	1.7 (1.1)	1.9 (1.3)	-	1.7 (1.2)	2.1 (1.5)	3.5 (1.8)	4.7 (0.8)

Figure 2: Students' assessment of resources used for project development

Two-way ANOVA with Repeated Measures on two factors, (a) phase of project development and (b) classification group, was performed in order to identify significant differences between students' assessment of the resources used for project development. Specifically, we were interested to perform the following investigation: (a) Comparison within each phase – between groups, between distinct resources; and (b) Comparison between phases: for each group and for each distinct resource (For a discussion of two-way ANOVA see Jones, n.d.).

Table 5 illustrates the results of the ANOVA test. Each group was presented by the mean of its elements' grades. Since no interaction was found, we concluded that a systematic behavior of factors exists. The ANOVA results indicated that the groups: *Bibliographic Resources* and *(Infor-*

mal) Human Resources demonstrated significant differences (due to resources and phases). In contrast, no such significant differences were found for *School, Mentoring,* and *Self-Studying.*

Table 5: Differences due to resources and phases

Classification group	Due to resources	Due to phases	Interaction
Bibliographic Resources (B): • Web (#1) • Articles (#2) • Books (#3)	$F_{2, 48} = 17.3$ $p < 0.0001$ Differences between resources during the entire process and within each phase as well	$F_{3, 72} = 6.42$ $p = 0.0007$ Differences between phases for the group and for its components as well	$F_{6, 144} = 1.04$ $p = 0.4044$
School (Sc): • A school teacher (#4) • School learning (#5)	$F_{1, 24} = 0.58$ $p < 0.4540$ No differences between resources during the entire process and within each phase as well	$F_{3, 72} = 2.65$ $p = 0.0554$ No differences between phases for the group and for its components as well	$F_{3, 71} = 0.05$ $p = 0.9838$
(Informal) Human Resources (A): • A grown-up acquaintance (#6) • A family member (#7) • A mate of student's age (#8)	$F_{2, 48} = 3.94$ $p < 0.0261$ Differences between resources during the entire process and within each phase as well	$F_{3, 71} = 3.85$ $p = 0.0129$ Differences between phases for the group and for its components as well	$F_{5, 117} = 1.44$ $p = 0.2148$
Mentoring (M): • The mentor (#9)	-	$F_{3, 72} = 1.0$ $p = 0.3979$ No differences between phases	
Self-Studying (St): • The student (#10)	-	$F_{3, 71} = 0.37$ $p = 0.7750$ No differences between phases	

The results of the two-way ANOVA with Repeated Measures (Table 5) justified further investigation of the following dimensions: (a) Comparison within each phase – between groups, between all distinct resources, and between each group's elements; and (b) Comparison between phases: for each group and for each distinct resource.

Comparison within a phase

We grouped the results of one-way ANOVA tests with Repeated Measures relating to comparisons within a phase in Table 6.

Table 6: Comparison within each phase

Phase	Groups	All distinct resources	Bibliographic resources (B)	(Informal) Human resources (A)
I	$F_{4,96} = 28.56$ $p < 0.0001$ (St)>(M,B)>(Sc,A)	$F_{9,239} = 23.86$ $p < 0.0001$ (10, 1)> ...>(4,7,6)	$F_{2,48} = 20.53$ $p < 0.0001$ (1)>(3,2)	$F_{2,47} = 6.03$ $p = 0.0047$ (8)> (7,6)
II	$F_{4,94} = 27.6$ $p < 0.0001$ (St)>(M)>(B) >(Sc,A)	$F_{9,234} = 23.86$ $p < 0.0001$ (10,1,9)>...>(5,8,4,7)	$F_{2,48} = 14.85$ $p < 0.0001$ (1)>(3,2)	$F_{2,46} = 1.85$ N.S.
III	$F_{4,96} = 28.17$ $p < 0.0001$ (St)>(M,B) >(Sc,A)	$F_{9,240} = 21.44$ $p < 0.0001$ (10)>(1,9)>…>(7,6)	$F_{2,48} = 9.92$ $p = 0.0002$ (1)>(3,2)	$F_{2,48} = 3.33$ $p = 0.0042$ (8,7)>(7,6)
IV	$F_{4,96} = 28.1$ $p < 0.0001$ (St)>(M)>(B) >(A,Sc)	$F_{8,216} = 18.47$ $p < 0.0001$ (10)>(9,1)>...>(5,7,4)	$F_{2,48} = 6.43$ $p = 0.0033$ (1)>(3,2)	$F_{1,24} = 1.45$ N.S.

Differences between groups within a phase: The findings indicated a resemblance between phases that concern learning and acquaintance with new material (Phase I- Studying the theoretical background needed for developing the project, and Phase III- Acquiring the needed technical programming knowledge). Regarding both phases, *Self-Studying* was significantly more appreciated than *Mentoring* and *Bibliographical Resources* (with a higher, though not significantly, appreciation of the mentor), which in turn were more appreciated than the *School* environment and the *(informal) Human Resources* (with higher, though not significantly, appreciation of the school).

There is also a resemblance between the phases that concern problem-solving aspects (Phase II- Identifying the main algorithmic ideas, and Phase IV- Implementation). Regarding both phases, *Self-Studying* was significantly more appreciated than other groups of resources. *Mentoring* was significantly more appreciated than the *Bibliographical Resources*, which in turn were more appreciated than the *School* environment and *(informal) Human Resources*. Interestingly, in Phase II the School was more appreciated (though not significantly) than the *(informal) Human Resources*, in contrast to Phase IV, where the order is reversed, meaning that in the implementation phase the students (on average) relied less on the school setting.

Differences between all distinct resources within a phase: The findings indicated that during the entire development process the students exhibited self-efficacy since (on average) they relied more on themselves (self-studying: resource #10) than on other resources. This was significantly demonstrated in phases III and IV that lead toward the complete implementation of the project, which obviously requires acquaintance with the needed technical programming knowledge. However, in phases I and II there was no significant difference between self-studying (resource #10) and using the Web (resource #1) – these two resources seemed to be dominant compared with

others. Apparently the school teachers (resource #4) were low-appreciated, though in phases I and II (which are usually considered as the beginning of the project development process) they were better assessed (though not significantly) than a family member (resource #7) and a grown-up acquaintance (resource #6). Interestingly enough, the students exhibited (on average) low reliance on classmates of their age (resource #8) when identifying the main algorithmic ideas (Phase II); moreover, they "even" appreciated more (though not significantly) the school learning (resource #5) when performing in that phase. In other phases the classmate resource was not extremely assessed (i.e., neither high nor low).

Differences between a group's elements within a phase: No significant differences were found between the elements of the *School* setting. The behavior of the distinct bibliographic resources is consistent within all phases (Table 6). The Web seems to be the dominant bibliographic resource in each phase and the books were more appreciated than articles (though not significantly). As for the group of *(informal) Human Resources*, differences were observed only in the phases that concern learning and acquaintance with new material (phases I and III). Specifically, a classmate of a student's age was indicated as most dominant in both phases, though in phase III it was not significantly different from a family member.

Comparison between phases

One-way ANOVA with Repeated Measures was performed, aimed at identifying differences between phases: (a) for each group, and (b) for each distinct resource.

The findings indicated significant differences between phases for the same groups (A and B) that exhibited differences between their elements within distinct phases. Specifically, the students used more *Bibliographic Resources* (group B) in phases that concern learning and acquaintance with new material ($F_{3, 72} = 6.42$, $p=0.0007$, (I, III)>(III, II)>(IV)). The *(informal) Human Resources* (group A) were mostly used when learning the theoretical background and for identification of the main algorithmic ideas ($F_{3, 72} = 4.05$, $p=0.0102$, (I, II)>(III, IV)).

Significant differences between phases were observed for only 4 distinct resources (3 of which are the elements of the *Bibliographic Resources* group). Interestingly, apparently the students consulted with classmates of their age mostly in the phase of learning the theoretical background, and then in the following (though not significantly different) descending order: when implementing the project, acquiring the technical programming knowledge, and finally, when identifying the main algorithmic ideas ($F_{3, 70} = 3.03$, $p = 0.035$, (I)>(IV, III, II)).

A concise view of the findings

Figure 3 illustrates the differences between groups of resources with respect to phases of project development. We classified the phases according to common characteristics into the following super-phases: (a) learning new material, and (b) doing problem solving. Three levels of the groups' contribution were identified. On the higher level is *Self-Studying*, which in all phases was significantly more appreciated compared with other groups. Regarding the second level, the groups *Mentoring* and *Bibliographical Resources* are located with differences in the *Problem-Solving* super-phase (benefiting the mentor) and similar appreciation in the *Learning New Material* super-phase. Finally, *School* environment and *(informal) Human Resources* are located at the third level. Interestingly, in Phases I, II, and III the students relied more on *School* whereas in Phase IV they relied more on *(informal) Human Resources*.

Super Phase	Learning new material		Doing problem solving	
Phase / Level	Phase I (Theoretical)	Phase III. (Technical)	Phase II (Algorithmic)	Phase IV (Implementation)
High	*Self Studying*			
Intermediate	*Mentoring ~ Bibliographic*		*Mentoring > Bibliographic*	
Low	*School > (inf.) Human*			*(inf.) Human >School*

Figure 3: Levels of groups' contribution

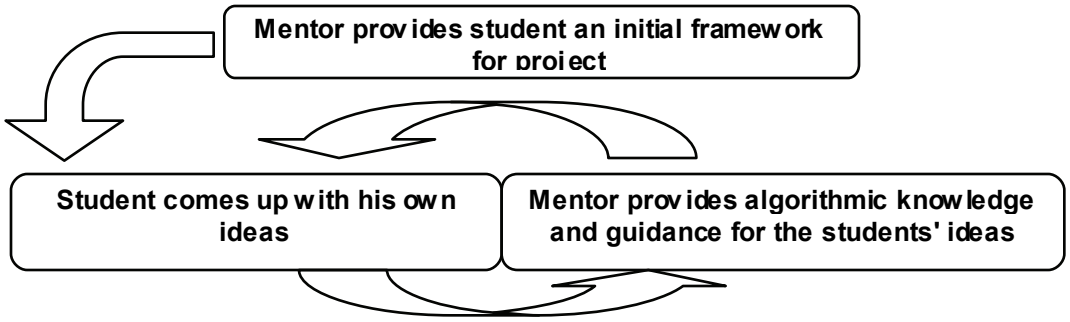
The story of one pair of student-mentors

Here we relate "the story" of the joint work of one pair - a high-school student and his young mentor, demonstrating the educational benefits of the program. The 12th grade student studied in a high-school located in an underprivileged Israeli town that specializes in students majoring in the sciences. We matched the student with a computer science graduate student who specialized in statistical machine learning. Being aware of the educational benefits of our program and of the project development activity in particular, the mentor was determined to succeed in his mission and therefore accompanied the mentoring process with ongoing reflection that was much deeper than just routine documentation.

The high-school student chose to develop a project (offered to him by the mentor) for automated completion of human sentences, motivated by knowing that the product of such a project may have useful applications for new communication technological devices such as SMS, palm computer devices, and internet messenger. The mentor exposed the student to a relevant theoretical background - Markovian and Bayesian algorithms - which are known as powerful tools of vast use in computer vision, voice recognition, and computational biology (Korb & Nicholson, 2003).This could not be achieved by the mentoring of a school teacher, often lacking high academic background and acquaintance with up-to-date technologies. The student lacked, of course, the necessary formal knowledge in advanced computing, but since he was highly talented, he quickly grasped the intuitive understanding of the advanced algorithms. Moreover, his own creativity, coupled with the proper mentor's guidance, led to a novel idea – a statistical tree structure that can handle Markov chains of variable length.

According to the mentor, a challenging part during the project development process was to achieve a proper working relationship between the mentor and student based on confidence and independence. Although the relatively small gap of age may have contributed to the building of a fruitful relationship, the out-of-school mentor's lack of official authority and expertise in disciplining high-school students was evident. The project development requires a great deal of maturity and self-discipline from the student and at the first stages of development, the student was reluctant to contact his mentor and follow timetables. At this point, cooperation between the mentor, teacher, and project management team synergized to solve these problems, which could otherwise have emerged as a major obstacle for the success of the program.

The student eventually created a project of significant scientific value, bringing about his own creativity and talent within the limited time framework of the project, as illustrated in Figure 4.



	Actual Flow of the Work
I	**Initial Idea by Mentor:** developing a tool for analyzing texts by computer using a statistical Markovian model.
II	**Student responded with his own ideas:** suggests developing a tool for automatic completion of texts upon typing.
III	**Algorithmic guidance from mentor:** the mentor introduced to the student essential background on Markovian statistical models for representing text.
IV	**Initial implementation = student hands-on experience:** the student wrote down a very simple implementation of known algorithms in order to get hands-on experience and intuition for understanding Markovian statistical models.
V	**Analysis of the initial model:** the initial algorithm's performance was analyzed by the mentor and the student.
VI	**Learning from experience:** the mentor asked the student to suggest his own ideas for developing the initial algorithm.
VII	**Novel algorithm by the student:** despite the lack of formal background, the student's own creativity, coupled with proper guidance, led to a novel idea – a statistical tree structure that can handle Markov chains of variable length.

Figure 4: Project Design & Implementation
A feedback loop between mentor and student

The final project won him a prize in a national contest of young scientists in Israel, and he later represented the country in a European contest.

The mentor indicated the benefits of the process to the community - strengthening the efforts to attract young people to careers in computing and his satisfaction to lead an interactive process in which he learned from teaching in his own field of interest.

Concluding Remarks

In this paper we describe a novel enrichment program designed to bridge the gap between school education and the "real world" of computing that is especially related to content, learning culture, and professional norms. We believe that the interaction of high-school students who major in computer science or software engineering with leading representatives of the computing community of practice may motivate them to pursue their studies further or pursue a career in the field. Accordingly, our program aims at exposing talented high-school students "directly by leading

experts" to state-of-the-art computing research, advanced technologies, software engineering methodologies, and professional norms. The interaction between the students and the experts, who actually become role models for the students, occurs at two levels: (a) during enrichment plenary meetings, and (b) one-to-one interaction in which students develop software projects under the apprenticeship-based supervision of professionals from academia and the hi-tech industry.

Implementation of the program for the last three years was accompanied by an ongoing evaluation. In Yehezkel and Haberman (2006) we described a preliminary assessment of students' attitudes toward the enrichment meetings and the "different-from-school" style of learning that characterized the program. The findings indicated that the students fully appreciated the experience of meeting researchers and professionals in the field, and they were very interested in the program. Specifically, they highly appreciated the wide exposure to a variety of subjects (more than focusing in-depth on one subject). In addition, the students indicated that the enrichment meetings increased their motivation to pursue further studies and professional careers in computer science and software engineering.

The study presented here focuses on assessing the project development activity under the apprenticeship-based supervision of professionals. Specifically, we discuss the contribution of resources that students used for various phases of the project development activity. The findings indicated that during the entire development process the students exhibited self-efficacy since (on average) they relied more on themselves than on other resources. This was clearly demonstrated in phases that lead toward the complete implementation of the project. Interestingly enough, the students exhibited low reliance on classmates their own age and usually appreciated more (though not significantly) the school learning that took place. Apparently, the school teachers were low-appreciated, though at the beginning of the project development process they were better appreciated (though not significantly) than a family member and a grown-up acquaintance. In the phases that concern acquaintance with theoretical technical knowledge, the mentoring and bibliographic resources were viewed by the students as having similar value. In the phases that concern problem-solving activities, students relied more on the mentors than on bibliographic resources; however, in these phases they used bibliographic resources more than they consulted the (informal) human resources. Interestingly, during the entire development process, the web was perceived as the most significant bibliographic resource. Specifically, to achieve acquaintance with the needed theoretical knowledge, self-studying and the web were perceived as the most significant resources, which may imply that the mentors' guidance inspired the students' self-inquiry and self-study. However, in other phases, no significant differences were indicated between consulting the mentor and using the web.

Based on the study findings, we concluded that project development experience under the supervision of professional experts may motivate students to acquire in-depth knowledge in computing, promote creativity, as well as enhance self-learning and inquiry ability. In addition, the interaction with role models may contribute to establishing professional norms.

We hope that further implementation of the program, along with recruitment of more representative experts from academia and hi-tech industry, will promote a culture of learning and work befitting the dynamic world of industrial computing, thus providing the students with an entry point into the computing community of practice.

References

ACM/IEEE Joint Task Force on Computing Curricula. (2004). *Software engineering 2004: Curriculum guidelines for undergraduate degree programs in software engineering. A volume of the computing curricula series*. Retrieved June 25, 2007, from http://sites.computer.org/ccse/SE2004Volume.pdf

Bracken, B. (2003). Progressing from student to professional: The importance and challenges of teaching software engineering, *JCSC*, *19*(2), 358-368.

Ben-Ari, M. (2003). Situated learning in this high-technology world. *The 7th International History, Philosophy and Science Teaching Conference*, Winnipeg, Canada. Journal edition published in *Science & Education*, *14*(3-4), 2005, 367-376.

Ben-Ari, M. (2004). situated learning in computer science education. *Computer Science Education*, *14*(2), 85-100.

Bergin, J. (2005). *Fourteen pedagogical patterns for teaching computer science*. Retrieved March 25, 2006, from http://csis.pace.edu/~bergin/PedPat1.3.html

Computing Research Association. (2005) *Cyberinfrastructure for education and learning for the future: A vision and research agenda* [Report]. Retrieved June 25, 2007, from http://www.cra.org/reports/cyberinfrastructure.pdf

Denning, P. J., & McGettrick, A. (2005). Recentering computer science. *Communications of the ACM*, *48*(11), 15-19.

Fincher, S., Petre, M., & Clark, M. (Eds.). (2001). *Computer science project work principles and pragmatics*. London: Springer-Verlag.

Gal-Ezer, J., Beeri, C., Harel, D., & Yehudai, A. (1995). A high-school program in computer science. *Computer*, *28*(10), 73-80.

Gal-Ezer, J., & Zeldes, A. (2000). Teaching software designing skills. *Computer Science Education*, *10*(1), 25-38.

Helfman, J., & Eylon, B. (2003). *Systematic inventive thinking* [in Hebrew]. Rehovot, Israel: Department of Science Teaching, Weizmann Institute of Science.

Holcombe, M., Stratton, A., Fincher, S., & Griffiths, G. (Eds.). (1998). Projects in the computing curriculum. *Proceedings of the Project 98 Workshop*. London: Springer-Verlag.

Jones, J. (n.d.). *Stats: Two-way ANOVA*. Retrieved May 20, 2008 from http://people.richland.edu/james/lecture/m170/ch13-2wy.html

Korb, K. B., & Nicholson, A. E. (2003). *Bayesian artificial intelligence*. Chapman & Hall / CRC.

Long, P. D., & Ehrmann, S. C. (2005). Future of the learning space: Breaking out of the box, *Educause*, 42-58.

The Ministry of Education, Israel. (2004). *A high-school software engineering program* [in Hebrew]. Retrieved June 25, 2007 from http://csit.org.il

National Academy of Sciences, National Academy of Engineering, Institute of Medicine. (1997). *Adviser, teacher, role model, friend: On being a mentor to students in science and engineering*. Washington, DC: National Academy Press. Retrieved June 25, 2007, from http://www.nap.edu/readingroom/books/mentor

Papert, S., & Harel, I. (1991). *Costructionism*. Ablex Publishing Corporation.

Passig, D. (2001).Taxonomy of IT future thinking skills In H. Tailor & P. Hogenbirk, *Information and communication technologies in education: The school of the future* (pp. 152-166). Boston: Kluwer Academic Publishers.

Shapiro, E. (n.d.). Prof. Ehud Shapiro [Biographical notes]. Retrieved June 20, 2008 from http://www.wisdom.weizmann.ac.il/~udi/bio.html

Sherrell, L. B., & Shiva, S. G. (2005). Will earlier projects plus a disciplined process enforce SE principles throughout the CS curriculum? *ICSE*, St. Louis, Missouri, USA, 619-620.

Sims-Knight, J. E., & Upchurch, R. L. (1993). Teaching software design: A new approach to high school computer science. *Annual Meeting of the American Education Research Association,* Atlanta, GA, April 1993.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Yehezkel, C., & Haberman, B. (2006). Bridging the gap between school computing and the "real world", *LNCS, 4226*, 38-47.

Appendix

The following four questions were posed to the students, who were asked to evaluate the resources for each question in the following table format:

1) When studying the theoretical background needed for developing the project, to what extent (high=5, low=1) did you use the following resources?[Phase I]
2) When identifying the main algorithmic ideas, to what extent (high=5, low=1) did you use the following resources?[Phase II]
3) When acquiring the needed technical knowledge, to what extent (high=5, low=1) did you use the following resources?[Phase III]
4) When implementing the project (writing and testing the code), to what extent (high=5, low=1) did you use the following resources?[Phase IV]

Level **Resources**	**Low** **(1)**	**2**	**3**	**4**	**High** **(5)**
The mentor					
The schoolteacher					
Self-studying					
School learning (materials and methods)					
Professional books					
Professional articles					
The Web					
A classmate of your age					
A family member					
A grown-up acquaintance					

Biographies

Bruria Haberman received her Ph.D. degree in Science Teaching from the Weizmann Institute of Science. She is currently a faculty member in the Department of Computer Science in the Holon Institute of Technology, teaching Logic Programming, Data Base Systems and Expert Systems. She is also a member of the computer science team in the Davidson Institute of Science Education in the Weizmann Institute of Science, where she leads the Computer Science, Academia & Industry educational program for talented high school students and their teachers. She is also a leading member of Machshava, the Israeli National Center for computer science teachers. She has developed learning materials for high school level in the areas of logic programming and artificial intelligence, and algorithmic patterns. She has developed academic programs for undergraduate level in computer science. Her primary research interests are computer science educational research, students' conceptualization of computer science, as well as in-service teacher education and distance learning.

Cecile Yehezkel received her Ph.D. degree in Science Teaching from the Weizmann Institute of Science. She holds a M.Sc. in Bio-Medical Engineering from Tel Aviv University and a B.Sc. in Electrical Engineering from the Technion, Haifa. She is currently teaching courses related to microcomputers and low level language at the School of Engineering at Bar-Ilan University. She leads the Computer Science, Academia & Industry educational program for talented high school students and their teachers in the Davidson Institute of Science Education in the Weizmann Institute of Science where she is member of the computer science team. She has developed learning materials and a software environment to teach Computer Architecture at introductory level. Her research interests focus on human-computer interaction, modeling and simulation design and evaluation, computer architecture and engineering education.