An Ability-oriented Approach for Teaching Programming Courses

Ying Li[1] You Song[1] Anas Moukrim[1] Shicheng Yu[1]
liying@buaa.edu.cn
[1]School of Computer Science and Engineering, Beihang University, Beijing, China

***Abstract*—This Research-to-Practice Work-In-Progress paper proposed a multidimensional ability-oriented approach for teaching program with the integration of outcome-oriented, student-centered, project-based and contest-driven. The main contributions were: (1) proposed an Ability-Driven Programming Model (ADPM) from two-dimensions of knowledge taxonomy and practice taxonomy which refined Bloom Taxonomy and defined computer programming ability from four hierarchical levels, they are basic ability, comprehensive ability, engineering ability and innovative ability; (2) designed an improved student-centered pattern to reconstructed contents to make each topic associated with six objectives in Bloom's taxonomy and explored some facts to verify the effect of both student-centered teaching and learning; (3) adopted a project-driven method based on "Conceive-Design-Implement-Operate" to solve complex engineering problems and elaborated on how to design good projects and how to measure the quality of projects in detail. Finally, analysis conducted using survey questionnaires and classroom videos indicates that use of Ability-Driven methodology has motivated students to become active learners and improved engineering practice ability. The Ability-Driven methodology provides an applicable model for a student-centered teaching pedagogy to cultivate students' engineering habits and teach them to think like an engineer.**

Keywords—Ability-oriented, Bloom Taxonomy, Student-centered, CDIO

I. INTRODUCTION

The rise of new engineering profiles is changing the education system. The, Sydney Accord and Dublin Accord proposed the concepts of Student-Centered Learning, Outcome-Based Education and Continue-Quality Improvement respectively, indicating that undergraduates need to develop their abilities to solve Well-defined Problems, Broadly-defined Problems and Complex Problems. Further, the cultivation of ability has become the key factor to the future development of higher education. [1]-[4]

As we know, knowledge, skill and ability are the important components of programming courses and they play different roles at different levels. Programming knowledge focuses on understanding programming language and it can be taught and learnt. It helps students define and delimit computer programming problems; Skill refers to applying the knowledge to solve programing problems and it can be developed through training and experience, which helps students design solutions. Ability is the quality of being able to do something through the application of knowledge and skills simultaneously. It helps students create practical engineering solutions. In brief, knowledge is theory, skill is practice and ability is application.

Through research, we found many programing courses introduce engineering-related projects into class and design their curriculums for students' characteristics and industry needs. Paper [5] used an inverted classroom by mixing the application of technology with hands-on activities. Paper [6]-[8] proposed some paradigms to transfer teacher-centered to student-centered. Paper [9]-[11] described a five C's framework to integrate consistency, collaboration, cognition, conception, and creativity in a student-centered teaching pedagogy. Paper [12][13] sought to develop outcome-based methodologies based on Bloom's Taxonomy for improving the quality of instruction.

This paper proposed a multidimensional ability-oriented approach for teaching programming with the integration of outcome-oriented, student-centered, project-based and contest-driven. Outcome-oriented indicates what abilities students need to have; Student-oriented indicates why they need to have these abilities; Project-oriented indicates how to effectively enable students to have these abilities; Contest-driven indicates how to know students have these abilities. Based on the above methods, we establish an outcome-based teaching system which takes theoretical teaching as foundation, programing contest as assistance-teaching means, engineering project as case and application as a direction.

The rest of the paper is structured as follows: Section 2 introduced ability-oriented teaching from four aspects: outcome-oriented teaching, student-centered learning, contest-driven practicing and project-based training. Section 3 analyzed some statistical data from a course which applied the ability-oriented approach to indicate the advantages in terms of the proportion of students who answer the questionnaires. Finally, section 4 showed the conclusions and outlines some challenges and future work.

II. ABILITY-ORIENTED TACHING

A. Outcome-oriented teaching

Based on original Bloom's Taxonomy, we propose a new Ability-Driven Programming Model (ADPM) from two-dimensions of knowledge taxonomy and practice taxonomy. Teaching contents are reconstructed to comply with Outcome-Based Education (OBE) requirements and each topic is associated with a specific ability. It mainly reflects the changes from spreading knowledge to improving ability, from organizing teaching in the order of chapters to teaching according to the correlation and connectivity of knowledge points and from over-emphasis on results to process assessment.

[14] We explore a spiral and progressive teaching process from theory to practice to engineering by designing engineering-cases originated from the industry, as shown in Figure 1.

In order to carry out the ability-oriented curriculum teaching reform, we first answer what is computer programming ability. It includes four hierarchical levels and we describe each ability from two aspects, knowledge and practice.

1) Basic Ability

Basic knowledge indicates how to write basic programs and basic practice indicates how to write correct programs. Specifically, students should understand the foundation knowledge of grammar (e.g., simple data type, variables, three control structures and input-process-output pattern), and can both write a hundred lined program in C language to solve some simple problems and debug code. The basic ability is one of the simplest skills of programming, and is also the basic goal of programming course.

2) Comprehensive Ability

Comprehensive knowledge involves how to write larger programs and comprehensive practice involves how to develop efficient programs. More precisely, students should understand the limitations of computation and storage, and master common algorithms, data structure and complex data type (e.g., array, pointer, structure and file), and can analyze, design and solve common problems, and can further write efficient programs which needs to understand two principles, one-dimensional linear storage principle and one-dimensional linear temporal computational principle. The comprehensive ability implies to the increased abilities to deal with real problems and design models, which are necessary for training practical talents.

3) Engineering ability

Engineering knowledge refers to solving complex problems and the engineering practice aims to work in group. At this level, students should understand the trade-off between the scale of the real problem, the complexity of algorithm and the precision of the solution, and can design large-scale and high-quality programs by using modularization thinking, software component technique, interface-based programming, etc. The program code should have both a good design style and good attributes (e.g., good readability, reliability, maintainability, reusability and scalability). Real problems are usually solved by team work. The engineering ability improves students' engineering practice ability, which can lay a foundation for large-scale software development.

4) Innovative ability

Innovative knowledge involves how to solve comprehensive problems in different disciplines and practice indicates to how to internalize knowledge into ability. On one hand, students should understand the essence of computational thinking: model-abstraction and dimension-reduction and the differences between conceptual data structure, logical data structure and physical data structure. And on the other hand, students should learn how to use computational thinking to formulate problems and how to map the problems described in natural language into models described in formal language and then into the programs described in programming language. The innovative ability focuses on developing the skills of creative problem-solving, critical thinking and knowledge integration.

In conclusion, the Ability-Driven Programming Model (ADPM) defines the core objectives of each level from two dimensions of knowledge and practice. The abilities trained in ADPM have a hierarchical relationship which follow the Bloom's Taxonomy.

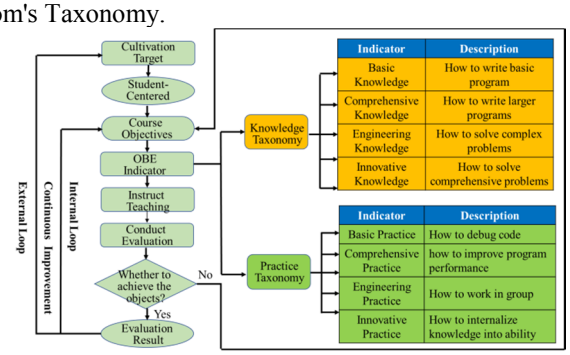

Figure 1 Ability-Driven Programming Model

B. *Student-centered learning*

The concept of "student-centered" focuses on how students learn and that includes three aspects, learning content, learning process and learning effect. [15] Through the study of the cognitive processes of the human brain, we found that abstract ability is the key for humans to understand the world and construct a cognitive model. Abstract ability is exactly one of the essences of computational thinking, and is also the training goal of programming courses. Therefore, the ultimate objective of programming teaching is to develop students' abstract thinking skills and form their own abstract principles. To achieve this, we find "student-centered" to be the most effective means. "Student-centered Learning" requires students to construct cognitive model in their own minds. "Student-centered Teaching" requires teachers to guide students to build the cognitive models and construct their own understanding. Teacher is the guide and developer rather than being the master or commander.

In this paper, we use problem-oriented method to implement the student-centered model and revise the teaching contents based on Bloom's Taxonomy. And we propose an improved ISPO (Input-Store-Process-Output) pattern based on the classic IPO (Input-Process-Output) model. Most programing courses focus on algorithm design more so than the process, although both input and storing are the foundation of programming and they influence the performance of algorithms greatly.

- Remembering: students learn how to use computer to input data and teachers teach how to map all kinds of objects into the right data types (int, double, float, char).
- Understanding: students learn how to use computer to store information and teachers teach how to create different storage types in a linear space (variable, constant, array and pointer)
- Applying: students learn how to use computer to process data and teachers teach how to break the method of data process into data flow (input, output) and control flow (if, for and while).

- Analyzing: students learn how to use computer to express the complicated relationship between things in the real world and teachers teach how to choose appropriate data structure to organize data effectively (linear structure, nonlinear structure).
- Evaluating: students learn how to use computer to solve problems while performing tasks more efficiently and teachers teach how to reorganize program structure (function and modular programming).
- Creating: students learn how to think like a computer and teachers teach how to abstract the real world into a computational model.

Student-centered programing course idealizes teaching is more than imparting knowledge, it is applying the knowledge. Learning is more than memorizing the syntax, it is acquiring precise understanding. We explore a new teaching mode consisted of five stages, they are: reading before class, teaching and discussing in theory class, practicing in practice class and writing lab reports after class.

- Stage 1: Before class, teachers distribute some reading materials and students establish their preliminary cognitive models.
- Stage 2: In theory class, teachers impart new knowledge and organize discussion. Students gradually correct and perfect their cognitive models through collision of ideas. Classroom discussion develops students' learning abilities of the Bloom's Taxonomy, that is, the abilities of applying, analyzing, evaluating and creating.
- Stage 3: In practical class, teachers attend to the students' questions. Students use their cognitive models to solve real problems. Through practice, the explicit knowledge is embodied into tacit knowledge and the learning is triggered by performing several processes. Finally, students can develop new knowledge.
- Stage 4: After class, teachers adjust the teaching pace based on students' learning needs. Students consolidate their knowledge by writing lab reports.

How to know students complete the knowledge internalization? It could be measured from three aspects. ① Description: Whether students can describe the real problems as cognitive computational models that computers can understand; ② Abstraction: Whether students can abstract the cognitive computational models into computational models which computers can solve. ③ Solution: Whether students can translate the computational models into program which computer can execute. To sum it , three factors, the correctness of description, rationality of abstraction and accuracy of solution, affect the effect of student-centered learning.

How to know teachers implement the student-centered learning? We have developed the following five aspects. ① Teaching resources: Whether the resources (curriculum syllabus, teaching materials, teaching calendar, teaching plan, assessment) are designed based on students' needs and can be revised and improved regularly. ② Teaching methods: Whether teachers focus on students' engagement and can adopt open and heuristic methods to fully mobilize students learning enthusiasm; ③ Teaching contents: Whether the contents are in line with students' acceptability and can be adjusted according to the students' feedback in class; ④ Teaching cases: Whether the cases are close to real world and daily life and can be understood easily by students; ⑤ Teaching evaluation: Whether the evaluation is precise and reasonable and can be customized according to students' learning behavior.

C. Contest-driven practicing

Introducing competition mechanism into practical teaching can both consolidate students' theoretical knowledge and improve their practical ability. We adopt contest-driven model to develop students' programming ability. It includes,

- Improving teaching method: we take ACM-ICPC programming contest problems as teaching cases that arouse students' interest in solving problems and make them better understand the contents.
- Improving teaching contents: we describe the cases as ACM-ICPC style problems (input, output and test cases) to make the contents more interesting and readable in order to emulate practical application scenarios;
- Improving learning style: Driven by programming contest, both individual leaning and teamwork is used to improve students' ability of communication and practice;
- Improving assessment approach: By taking advantage of Online Judging, automatic and intelligent evaluation methods are used to judge code quality accurately and instantly which can spark their enthusiasm for programming.

D. Project-based training

The Washington Accord states that students need to be able to solve complex engineering problems. In order to achieve this goal, we propose a project-driven teaching method based on CDIO engineering education idea. CDIO stands for Conceive-Design-Implement-Operate and it is an innovative framework for engineering education. We use CDIO syllabus for developing our course goal and the students are required to have four knowledge capabilities: basic knowledge of programming, personal ability, interpersonal teamwork ability and engineering system ability. [16] And all of these capabilities are exactly what the ability-oriented teaching model aims to develop (see section I). It can be seen that the ability-oriented teaching model and the project-driven training model complete and enhance each other.

As we know, designing teaching cases is one of the most important and difficult things. Further, the projects used in project-driven training model have a special characteristic-reality, which indicates that all teaching factors involved of contents, methods, goals and evaluations need to be as close to real world, real life and real work as possible. Then, how to measure the quality of projects?

- Good projects can promote the interaction between teachers and students. Positive and regular interaction is an effective way to increase students' motivation and stimulate students' interest in learning.

- Good projects can enhance cooperation among students. Cooperative learning is an effective strategy for students to achieve a wide range of academic and social outcomes. Teamwork results in much more learning than students' working alone, competitively or individually. It helps them to develop understanding of course contents.
- Good projects can encourage students to keep practicing. Learning is considered as a collective construction of understanding rather than individual memorization of facts. Students must apply what they learned to solve practical problems, and in this way the knowledge can be internalized and the new knowledge can be constructed on prior knowledge.
- Good projects can provide feedback to students. Effective feedback can help students identify the limitations of their knowledge. Before class, they need to receive feedback to know their level of understanding of the knowledge learnt. In class, they want to get feedback on their performance in time. After class, they also need feedback to know what they missed, what they have to learn.
- Good projects can boost students' confidence. When solving complex problems, it encourages students to articulate their ideas with others which helps them to find multiple solutions and improve their skills.
- Good projects can help students achieve higher goals. It creates a culture of high expectations which is an effective way to accelerate students' progress. High expectation teaching can lead to higher levels of engagement, motivation and self-efficacy in students.
- Good projects can help students find their own learning methods. It has a positive impact on personality development and respects the individual differences in abilities, backgrounds, cultures and experiences. It is an important part of the student-centered model.

Then, how to design good projects? Primarily good projects should have two main characteristics, complexity and engineering.

First, complexity and authenticity are mainly reflected in the following aspects:

1) Teaching content is complex and authentic:

Good projects are representations of real engineering situations and they have real data and a real context. We improve our teaching from solving well-structured problems, in which the initial state, goal and constraints are clearly defined, to solving complex engineering problems. Teachers not only impart programming knowledge, but also train students' engineering skills, engineering thinking and engineering literacy.

2) Teaching process is complex and authentic:

We introduce enterprise mentors into our class who can help students master the engineering design process and understand how to integrate it with their classroom learning. Compared with classroom teaching, project-based teaching is much more complex, which involves various tasks: project design, team management, process guidance, ability evaluation, etc. Therefore, an innovative teaching method "teacher-engineering mentor, textbook-project documents, classroom-practice base" is proposed to reduce the difficulty of implementation. Students can directly obtain the background knowledge and engineering skills from enterprise mentors and that is exactly what the industry needs.

3) Teaching evaluation is complex and authentic:

We develop a process model for multidimensional assessment which focuses on students' motivation, ability, attitudes, behaviors, homework, learning, etc. Homework is the reflection of priorly imparted knowledge; project can reflect real competence and proficiency in the knowledge they have acquired; grades reflect the achievement levels of the teaching contents and objectives; attitudes can reflect the self-regulated level which students transform their mental abilities into task related skills.

4) Teaching goals are complex and authentic:

We intend to develop students' scientific analysis ability, engineering practice ability, innovation ability and comprehensive design ability and it places greater demands on the students and teachers. For this, we create a new practical teaching system consisting of three progressive parts - experiment, exercise and project. Experiment can help students to understand basic programing knowledge; exercise can improve their advanced practice skills and project can develop their engineering literacy.

Second, engineering is mainly reflected in the following aspects:

- Teaching the most advanced knowledge and technology required by the industry: quite different from common teaching, in our class, we spend much more time teaching how to use modern technology to solve complex engineering problems and how to cope with unforeseen events and most of the basic knowledge is learned by MOOC.
- Teaching different domain knowledge: engineering problems usually require the expertise of various disciplines. In order to overcome the limitation of knowledge gained in one discipline, we use a real synthesis of approaches to integrate knowledge and methods from different disciplines and break the barrier of different majors and create an agreeable atmosphere of vast-engineering majors. We gradually cultivate students' mental engineering habits and teach them how to think like engineers.

III. EXPERIMENTS

We did two experiments. First, the ability-oriented approach proposed in this paper is analyzed through a questionnaire. Second, the benefit of student-centered learning is analyzed by Flanders interaction analysis.

A. Questionnaire

A well-designed questionnaire is used to assess the effectiveness of our ability-oriented approach for teaching and some closed questions are shown in Table 1. The survey is given to 1200 students randomly selected in our programing course and SPSS (a world's leading statistical software) is used to analyze and model the reliability of survey data. As can be seen from Table 1, the reliability coefficient (Cronbach's alpha) of our collected data is 0.917, which is

considered to be very high reliability. In SPSS, The reliability coefficient normally ranges between 0 and 1 and the closer the coefficient is to 1.0, the greater is the internal consistency of the items. The reliability coefficient in Table 1 accurately indicates that the ability-oriented approach proposed in this paper is very well received by students.

B. Flanders interaction analysis

In order to verify the effectiveness of the multidimensional ability-oriented approach proposed in this paper, we use an observational tool, Flanders Interaction Analysis System, to classify the verbal behavior of teachers and students. We collect more than 1000 teaching videos and compare the traditional teaching method (TM) with our method (OM) from nine categories and parts of results are shown in Table 2.

From Table 2, we get the following conclusions: (1) the lectures in OM is reduced by 32% of TM and it indicates OM pays more attention on students to create a student-centered environment. (2) Student Discussion in OM is increased by 160% compared with in TM. The successful and effective classroom discussion can renovate the students, the instructors, and their collective experience.

In the course, a specific Online Judge (OJ) system is applied to test programs and some of records are shown in Table 3. Figure 2 shows the programming skills of students in recent three years which are evaluated based on the test standards obtained from OJ.

Table 1 Some Questionnaire Questions and Results

No.	Closed Questions	Results
1	The content is more practical and interesting	48%
2	Teachers play a very important guiding role	70%
3	The contents is very difficult to self-study	30%
4	Getting a lot of help through teamwork	42%
5	Willing to spend more time learning	64%

Table 2 Parts of Flanders Interaction Analysis Results

Categories	TM (%)	OM (%)
Teacher Lecture	58.9	40.3
Gives Directions	20.4	43.2
Ask Question	6.2	19.5
Student Discuss	14.3	36.5

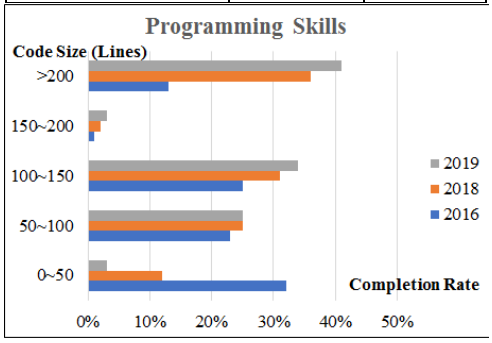

Figure 2 Students' Programming Skills

Table 3 Online Judge Record

User ID	Topic ID	Result	Score	Code Size (Bytes)	Execution Time (ms)	Memory Space(KB)
2322	3400	Accepted	1	403	34	1684
2321	3400	Wrong	0	402	23	1648
2317	3398	Wrong	0	1187	2	1708
2316	3419	Accepted	1	280	7	1684
2315	3398	Wrong	0	1186	3	1788
2314	3397	Accepted	1	711	10	1692
2312	3414	Accepted	1	196	2	1704

IV. CONCLUSION

It is found that the multidimensional ability-oriented approach for teaching program is an effective teaching method to improve students' both programming skills and computational thinking. It places students at the center of the learning process and reconstructs teaching contents based on Bloom's taxonomy to make it more in line with students' cognitive habits. The purpose of programming course is to teach students how to solve practical problems and we adopted a project-driven method based on "Conceive-Design-Implement-Operate" to integrate knowledge, ability and quality effectively. But ability is hard to measure and how to know students have developed their programing skills is our challenge and we will identify the important factors that influence teaching and learning and establish a set of scientific and reasonable evaluation standards.

REFERENCES

[1] Weimer, Maryellen.Learner-centered teaching: Five key changes to practice. John Wiley & Sons, 2002.

[2] Tom, Mary. "Five C Framework: A Student-Centered Approach for Teaching Programming Courses to Students with Diverse Disciplinary Background." Journal of Learning Design (2015): 21-27.

[3] Mahmood, Khalid, et al. "Implementation of outcome based education in Pakistan: A step towards Washington Accord." 2015 IEEE 7th international conference on engineering education (ICEED). IEEE, 2015.

[4] Jones, Bryan, Emma Vaux, and Anna Olsson-Brown. "How to get started in quality improvement." Bmj (2019).

[5] Gannod, Gerald, Janet Burge, and Michael Helmick. "Using the inverted classroom to teach software engineering." 2008 ACM/IEEE 30th International Conference on Software Engineering. IEEE, 2008.

[6] Faraon, Montathar, et al. "Learning by coding: A sociocultural approach to teaching web development in higher education." Education and Information Technologies (2019): 1-25.

[7] Zeki, Canan Perkan, and Ahmet Güneyli. "Student teachers' perceptions about their experiences in a student centered course." South African Journal of Education (2014).

[8] Wright, Gloria Brown. "Student-centered learning in higher education." International Journal of Teaching and Learning in Higher Education (2011): 92-97.

[9] Ortin, Francisco, Jose Manuel Redondo, and Jose Quiroga. "Design of a Programming Paradigms Course Using One Single Programming Language." New Advances in Information Systems and Technologies. Springer, Cham, 2016. 179-188.

[10] Ortin, Francisco, Jose Manuel Redondo, and Jose Quiroga. "Design of a Programming Paradigms Course Using One Single Programming Language." New Advances in Information Systems and Technologies. Springer, Cham, 2016. 179-188.

[11] Sicilia, Miguel-Angel, et al. "Programming Paradigms for Computational Science: Three Fundamental Models." International Conference on Computational Science. Springer, Cham, 2019.

[12] Davis, S. "Using Bloom's taxonomy to write learning outcomes." (2014).

[13] Fuller, Ursula, et al. "Developing a computer science-specific learning taxonomy." ACM SIGCSE Bulletin 39.4 (2007): 152-170.

[14] Daly C, Waldron J. Assessing the assessment of programming ability[J]. ACM SIGCSE Bulletin, 2004, 36(1): 210-213.

[15] Akram B, Min W, Wiebe E, et al. Assessing Middle School Students' Computational Thinking Through Programming Trajectory Analysis[C]. Proceedings of the 50th ACM Technical Symposium on Computer Science Education. 2019: 1269-1269.

[16] Ostrander T, ElKharboutly R, Jin K. Surviving" Open-ended Projects" in Project-Based Learning: A Teacher's Perspective[C]. Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education. 2017: 727-727.