Research and Reformation of Online Practice Oriented Teaching in Programming Foundation Course∗

Guiping Wang1,2
1. Information school, ZheJiang University of Finance and Economy
2. School of Computer Science, ZheJiang University
HangZhou, China
E-mail: w_guiping@163.com

Rui Feng1
1. Information school, ZheJiang University of Finance and Economy
HangZhou, China
E-mail: hzfengr@163.com

Abstract—Programming foundation is a course with very strong practicality. In order to promote students’ programming practice ability, this article presents a new teaching idea, online practice oriented teaching. This teaching idea emphasizes particularly on impartment of elementary thoughts and methods in programming, cultivating students’ practical ability through programming contest, online practice, and course project, etc. Based on this teaching idea, this article redesigns the main teaching contents of the course, including theoretic and practical teaching. Lastly, this article summarizes some effective teaching methods.

Keywords—Programming; Programming Practice Ability; Programming Contest; Online Practice; Course Project

I. INTRODUCTION

Programming foundation is a course established for science and engineering majors in colleges using C, C++, Java or other computer languages. The purpose of this course is to train students with basic programming methods and abilities. This course is generally open for lower grades in colleges, while students do not possess the basis knowledge of a programming language. So in the past, more emphasis was put on grammatical knowledge of a computer language during teaching process. Alternatively, along with grammatical knowledge teaching occasional time was put on programming thoughts and methods teaching.

However, we realize some facts during our teaching process. On the one hand, due to enormous grammatical system and strict rules of a computer language, it is impossible to study every aspect of grammatical knowledge in dozens of class hours. On the other hand, it is undoubtedly difficult for beginners to understand and master the grammatical knowledge in the short term if the teaching contents contain much grammatical knowledge. When begins with the course, students’ enthusiasm are mostly relatively high. They long for solving practical problems using programs after they accomplish this course. However, students soon realize that even after a semester pedantic grammar teaching, they can only write some small programs. These programs cannot solve practical problems at all. It is a great blow to the students’ enthusiasm to learn the follow-up related courses.

Furthermore, nowadays many application software are available, such as various types of visual development tools, Delphi, VC, VC.NET, VB and so on. Many students are keen to develop some simple application programs when they come into contact with these development tools. The training of thoughts and methods in programming, abilities and awareness in algorithm analyzing and designing are neglected. They fall into a mistake for such courses as programming foundation, data structure, algorithm analysis and design. They believe that software development is nothing but dragging some controls, compiling simple scripts. So they hold that it is not necessary to understand many complex data structures, and they do not focus on the development of ability in algorithm analyzing and designing anymore.

These issues and phenomena cause us to reflect on some problems. Which kind of main thread should be chosen in programming foundation course? Which kind of teaching idea should the teaching process be guided by? How to design the teaching contents of this course? What kinds of teaching methods should be adopted to guide students to improve their programming practical ability? These problems will be discussed in next sections.

II. MAIN THREAD OF THE COURSE

When choosing a main thread for programming foundation course, we have two choices. The first one is

∗ This article is supported by ZheJiang provincial Education Science 2010 annual planning project, To promote the planning and constructing of course group regarding collegiate academic competitions as an opportunity. (SCG156)

∗ Related research and work can be found in [8], which is based on the teaching idea presented in this article and awarded as eleventh five-year plan ZheJiang provincial Planning Textbook.
based on the grammatical knowledge of a programming language. The second one is based on programming thoughts and methods. But we insist on the second one. That is, the main thread of the course should be based on programming thoughts and methods, and supplemented by language teaching. The reasons can be summarized as two aspects.

Firstly, the teaching thread based on grammatical knowledge cannot guide students to truly understand and master the thoughts and methods in programming. The teaching of grammatical knowledge should be application problems oriented, because application problems reflect the thoughts and methods in programming. Whereas, the teaching thread based programming thoughts and methods not only avoid boring grammatical knowledge at a great extent, but also arouse the interest of students to understand and master programming thoughts and methods. For example, when we explain the basic knowledge of C/C++ language, we handhold numerical data processing for clues. For most students, numerical data processing is relatively easy to understand and master.

Secondly, programming foundation course should systematically teach students programming thoughts and methods. The past teaching process occasionally teaches a few programming thoughts and methods along with teaching of grammatical knowledge. But for beginners, occasional teaching does not attract their attention. The main thread of the course should be systematic teaching of programming thoughts and methods.

For example, in the past teaching process, recursion is often placed at the chapter of function design. After this chapter, most students have understood the implementation process of recursive functions, and mastered definition methods of recursive functions. However, it is still difficult for them to understand when recursive functions need to be used. In fact, recursion is an important algorithm in programming. Recursive function is only the implemented method of recursive algorithm.

As another example, in the past teaching process, we usually enumerate many examples using a single loop or dual loop when talking about loop structure. For example, when pursuing positive integer solutions of \( x^2 + y^2 = 2000 \), we need to use a dual loop structure to achieve. But from the algorithm point of view, the thought contained in this dual loop structure is enumeration. Beginners do not realize that this thought is algorithm and algorithm is the essence of programming. What beginners need to really grasp is programming thoughts and methods, that is, algorithm.

After identified the main thread of this course, we begin to think about how to carry out the teaching of the course.

Over the last decade, various programming competitions carried out in full swing, especially the International Collegiate Programming Contest (ACM/ICPC, [1]). The scale and impact of ACM/ICPC expands year after year in many colleges and universities all over the world. Andrew Trotman and Chris Handley([6]) show how to choose a good strategy during ACM/ICPC.

These competitions not only provide an opportunity for many enthusiastic programmers to usher their abilities in analyzing and solving problems, but also provide a practical platform for beginners to implement programming thoughts and methods. In addition, along with the promotion of various programming competitions, a number of program online judge (OJ for short) websites came into being. These OJ websites provide a new practical approach, online practice, for programming courses.

Based on above-mentioned background, we present and carry out a new teaching idea in programming foundation course: contests driven, online practice oriented, course project enhanced teaching idea.

III. A NEW TEACHING IDEA

During the teaching process, we adopt ACM/ICPC to stimulate students’ interest in learning and sense in competition, and develop students’ initiative thinking skills. In addition, based on several famous OJ websites, such as ZOJ([2]), POJ([3]), UVA([4]) and so on, we develop students’ abilities in problems analyzing and solving, group discussion, teamwork([7]), document-ation organization, etc. At the final stage of the course, we strengthen teaching effectiveness by course project.

A. Contest Driven

During our past teaching process, we found that most students still regarded course exam as the objective of course and lost motivation in learning. On the contrary, during our several years’ contest-training work, we are deeply impressed by students’ sense of accomplishment when they solve a problem successfully. The sense of accomplishment is even higher than good results obtained in the exam. But it is difficult for students to obtain this sense of accomplishment in past teaching process of programming foundation course. In addition, during our several years’ contest-training work, we are shocked by lower grad students’ enthusiasm of contest participation. We realized that programming contests brings great incentive for the learning of programming courses.

Therefore, it is necessary to introduce ACM/ICPC training methods and evaluation rules in teaching of programming foundation course. Driven by ACM/ICPC, students’ interest in learning and the sense in competition are stimulated deeply.

B. Online Practice Oriented

The traditional practice in programming foundation courses is usually disposed by teacher. Students are asked to write programs to solve some problems. It is
teacher’s work to determine whether programs are correct or not. This form of practice has two defects. The first one is lack of incentives mechanism, thus it is difficult to stimulate students’ interest in learning. The second one is that evaluation results are subjective and not timely due to its requirement for manually judge.

Online practice is different from traditional practice. The form of online practice is that OJ websites provide problems, and students submit their solution program online. Judge systems in OJ websites evaluate students’ solution and feedback results in real time. The problems on OJ websites are usually interesting and challenging. The judge process and results are fair and timely. Therefore online practice can arouse great interest of students.

The problems collected on OJ websites are mostly form ACM/ICPC at all levels. We will discuss the types, characteristics of these problems, and evaluation methods of OJ websites as well. At the same time, we analyze the roles of these problems in training students’ awareness and ability of algorithm analysis and design.

1) Introduction of problems types in ACM/ICPC

The types of problems in ACM/ICPC cover the range of algorithms such as greedy, dynamic programming and other optimization algorithms, as well as basic algorithms in number theory, graph theory, computational geometry and other fields. According to statistics, the types and proportion of problems in ACM/ICPC are shown in Table I.

<table>
<thead>
<tr>
<th>Types of Problems</th>
<th>Search</th>
<th>Greedy</th>
<th>Construct-ion</th>
<th>Graph Theory</th>
<th>Computational-geometrical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proportion</td>
<td>10%</td>
<td>5%</td>
<td>5%</td>
<td>10%</td>
<td>5%</td>
</tr>
</tbody>
</table>

We find that some of these algorithms are simple. Basic thoughts and methods of such algorithms are fit to be taught in programming foundation course.

For example, some simple algorithms for solving problems, such as enumeration, simulation and so on, do not need complex data structures in realization. Thus they are suited for introduction in the course.

Another example is high-precision calculation. It is an important application in ACM/ICPC. In fact, these problems can be converted into character arrays (or integer arrays) processing. It is the deepening of array applications.

Such algorithms as enumeration and such applications as high-precision calculation are easy enough for beginners in programming. After students master these basic algorithms, and possesses basic abilities in algorithm analyzing and designing, it is relatively easier for them to learn more complex algorithms in the follow-up courses.

2) B. Features of problems in ACM/ICPC

An ACM/ICPC problem typically contains five parts: problem description, input description, output description, sample input, and sample output.

a) Problem description: the problem description usually does not directly tell what is required to solve, but begins with a story or a game as background knowledge. So problem description is usually cumbersome, but interesting as well.

b) Input and output description: these descriptions give specification of input and output format.

c) Sample input and output: in order to understand the problem for students, the problem provides several sets of correct input and output data. Sample input and output are for the need to test program as well.

One of features of problems in ACM/ICPC is that there are multiple sets of test data for processing. This requirement has two purposes. First, multiple sets of test data can test all possible situation and prevent cheating. Secondly, multiple sets of test data are used to calculate running time of solution program and then test the strengths and weaknesses of the algorithm.

Problems in ACM/ICPC are totally different from previous exercises in the feature of multiple sets of test data. Previous exercises usually need to process only one set of data. After this processing, the program is over. The defect of one set of data is that it is a great chanciness to determine whether the program is right or not. When students submit problems on OJ websites, whether programs are correct or not is determined by online judge system objectively. Even the solution program has passed given sample input data, it is not necessarily correct. The reason is that the role of sample input data is only help users to understand problem. Sample input data does not reflect various circumstances of input data. In order to verify the correctness of programs, users are bound to design some more test data for testing their program. Therefore, problems in ACM/ICPC are entitled to develop students’ awareness of testing program.

3) C. Evaluation methods of ACM/ICPC

At the server of OJ websites, there are an input data file and a standard output data file for each problem. The input data file is used to test the user’s submitting program. It is usually capable of testing various special circumstances need to be considered. The standard output data file is gained from the standard solution program based on the input data file. It is a correct output data file. Evaluation system compares standard output data file with user's output data file character after character. After comparison, the judge system feedback results to the user timely.

The problems in ACM/ICPC are very strict with the output requirements. As long as the solution program considers incompletely or produce incorrect format output, the program could not be passed. This evaluation method ensures timely and fair judge results.
IV. REDESIGN OF COURSE CONTENTS

In order to adapt to this new teaching idea, it is necessary to redesign the contents of the course. At the same time, it is important to ensure teaching carried out around the main thread. The contents of the course can be divided into two parts, theoretical teaching and practical teaching.

A. Theoretic Contents

The theoretic contents of the course can be divided into three parts. The teaching contents arrangements and class hours are shown in Table II. For comparison, the traditional teaching contents of the course, which referred from [9], are list in Table III.

<table>
<thead>
<tr>
<th>TABLE II. TEACHING CONTENTS ARRANGEMENT AND CLASS HOURS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapters</td>
</tr>
<tr>
<td>Part 1: Basic knowledge of C/C++ language</td>
</tr>
<tr>
<td>Part 2: Basic thoughts and methods in programming</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Part 3: Course project</td>
</tr>
<tr>
<td>Total</td>
</tr>
</tbody>
</table>

T: theoretic class hours. E: experiments class hours.

TABLE III. TRADITIONAL TEACHING CONTENTS ARRANGEMENTS AND CLASS HOURS

<table>
<thead>
<tr>
<th>Chapters</th>
<th>Contents</th>
<th>T</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Elementary knowledge of C/C++</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Data types, operators and expressions</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Algorithms and control structures</td>
<td>15</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>Functions design</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>Arrays</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>Pointers</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>User defined data types</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>54</td>
<td>36</td>
</tr>
</tbody>
</table>

1) A. Basic knowledge of C/C++ language

The knowledge of C/C++ language taught in this part is only the minimum set of grammatical knowledge required for writing a program. Two principles should be conformed when selecting the knowledge.

The first principle is that the knowledge used first should be taught in priority. For example, the teaching target of "function design" section is only to design functions according to problem-solving requirement. Relevant knowledge such as function overloading, function templates, function with default parameters, etc. can be laid aside and taught in latter applications.
when necessary. The reason is that it is difficult to understand the usefulness of function templates for those students without much programming experience, and therefore they are reluctant to learn function templates.

Another example is the section of "pointer". Past teaching process is always keen to compare various pointers and their usage, while the basic roles of pointer are usually overlooked. In fact, the basic roles of pointer can be summarized as two aspects. One aspect is as function parameters, which can modify the value of the variable in the calling function. The other aspect is used to point to an element in an array, which can easily access each element of the array. For novices, it is enough to master these two basic roles of pointer.

The second principle is that for such grammar knowledge only used in later chapters, we put it in corresponding chapter as the relevant basic knowledge. For example, all the contents for processing character data are put into the second part of "Processing for character and string" as basis knowledge to introduce.

When we explain the basic knowledge of C/C++ language, we handhold numerical data processing for clues. In particular, we illustrate the knowledge with simple mathematical calculations and applications. During the explanation process, we introduce elementary thoughts and basic methods for solving specific problems with programs. For example, when calculating area of triangles, we introduce the use of mathematical functions. When determining whether three sides can form a triangle or not, we introduce branch structure and if statements. Another example is that Fibonacci sequence is adopted in several occasions such as introduction of loop structure, array application, etc.

2) B. Basic thoughts and methods in programming

This part of content is the emphasis of this course. The basic programming thoughts and methods we selected are shown in Table 2. These thoughts and methods are usually simple and do not involve complex theoretical knowledge, thus used frequently in programming. For example, when there are a large number of possible answers in a problem, and it is difficult to exclude some possible answers using logical approaches, then we have to adopt a strategy, which we called enumeration, to test each answer. Another example is that, in reality, it is difficult to find a formula or rule to solve some problems, the only strategy we can adopt, which called simulation, is to simulate in accordance with certain steps or rules.

When explaining these basic programming thoughts and methods, we begin with most simple example to introduce algorithm ideas, and then give further elaboration through a number of classic ACM/ICPC problems. During the explanation, we neglect theoretic knowledge as much as possible.

For example, when we explaining the ideas of enumeration algorithm, we begin with such a problem as, pursuing positive integer solutions of $x^2 + y^2 = 2000$. The solving ideas are: enumerating all the $(x, y)$ combinations, where $x$ and $y$ are in the range of $1, 2, \ldots, 44$, $44$ is the largest integer no more than square root of $2000$; for all the $(x, y)$ combinations, determining whether $x$ and $y$ meets $x^2 + y^2 = 2000$ or not, and if satisfied, it is a group of solution. Further, if $(8, 44)$ and $(44, 8)$ are considered as the same solution, and we need to avoid outputting same solutions, then we should ensure $x \leq y$ during the enumeration.

After this problem, we give further elaboration of enumeration algorithm through several ACM/ICPC problems related to Goldbach Conjecture. These problems study Goldbach conjecture from different angles, for example, outputting all the prime decompositions for a given even number, counting the number of prime decompositions.

Through these problems, we conclude that when adopting enumeration algorithm, we should pay special attention to two points. The first one is that neither repetition nor omission should occur during enumeration for ensuring the result is correct. The other one is that the number of enumeration should be minimized for reducing the running time.

After students understand these thoughts and methods, they will soon be able to solve some exercises, and then submit their solution through OJ websites. They repeatedly modify the solution program by results feedback from OJ websites, until they gain Accept (solution program is correct). During this process, students gain not only improvement of abilities in analyzing and solving problems independently, but also strong sense of accomplishment when solve each problem successfully. In addition, the practice forms, such as group discussion, problem-solving reports compiling, etc. can also train students’ abilities in teamwork, document organization and so on. This novelty of online practice can provoke student's interest in a large extent.

3) C. Course project

After this course, what most students want to know is what can achieve for the program written based on their knowledge. The purpose of course project is to answer this question. Course project aims to let students know that after they learn C/C++ language and master basic programming thoughts and methods, they will be able to write a small software (even the interface is character-based), a simple game software, or a simple information management software. When they master visual development tools in follow-up courses, they can transplant their thoughts and methods into a visual interface and gain a beautiful software.

This part of teaching take "the development of a character-based mine-clearing game" as example, explain the whole program development process step by step through task decomposition.
B. Practical Contents

The practical contents of this course can be divided into three parts (with the theoretical contents sync). Compared with traditional practical teaching, these practical contents have distinctive features. The practical contents arrangements are shown in Table IV.

<table>
<thead>
<tr>
<th>Stage</th>
<th>Practical Teaching Contents</th>
<th>Individual/group</th>
<th>Evaluation forms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part 1: Basic knowledge of C/C++ language</td>
<td>Imitating → Rewriting → Independent programming; Analysis of program results → Analysis of program implementation process → Debugging program</td>
<td>1 person/group</td>
<td>Submission of experiment al report</td>
</tr>
<tr>
<td>Part 2: Basic thoughts and methods in programming</td>
<td>Discussing thoughts of algorithm in groups; Completing the assigned exercises on OJ; Preparing problem-solving report.</td>
<td>3 persons/group</td>
<td>Submission of problem-solving reports and programs</td>
</tr>
<tr>
<td>Part 3: Course project</td>
<td>Completion of a course project</td>
<td>3 persons/group</td>
<td>Submission of program manual and complete program</td>
</tr>
</tbody>
</table>

The first part of practical contents focuses on C/C++ language grammar knowledge and development environment of VC. When arranging these practical contents, we abide by two thoughts. The first one is from imitating example to rewriting example, and thus independent programming, which enabling students to write a complete program from the first class. The second one is from analyzing of program results, to analyzing program implementation process, and thus debugging program, which aim to develop students' abilities of analyzing and solving problems independently.

The second part of practical contents is carried out in the form of 3 persons/group, which simulating real ACM/ICPC. The experiment tasks are discussing algorithm thoughts of example and completing exercises in OJ websites. Group discussion can help to cultivate students' abilities of verbal communication skills, teamwork, and so on. Usually after discussion, students gain further understanding for the algorithm thoughts, and form their own opinions. Evaluation forms includes submission solution programs and problem-solving reports, which can train students' abilities in written expression and document organization.

The last part of practical contents is course project. Students should complete their course project in 2 weeks in the form of 3 persons/group. Course project is the important part of this course. It is a further examination of teaching effectiveness.

V. Teaching Methods

During our research and reformation in programming foundation course, we summarize some effective teaching methods. All of these methods are focused on imparting elementary thoughts and methods in programming, and guiding students to engage in programming practice actively.

A. Case-Based Teaching

The case-based teaching methods, is to introduce language grammars or algorithm ideas, based on specific applications. This method is more effective than simply describing boring grammar knowledge and profound theoretical algorithm knowledge. We adopt case-based teaching method when we explain the C/C++ language, part 1 in table 2. The method can be embodied that we explain the C/C++ language based on numerical data processing, mathematical calculations and applications, etc. Along with the explanation process, we introduce elementary thoughts and methods for solving specific problems with programs.

For example, in most higher mathematics textbooks there is such a problem as, the parabola, \( y = x^2/(2*p) \), rotates around its symmetrical axis, \( x = 0 \), and forms a curved surface. A light source is placed at the focus \( F(0, p/2) \), of the curved surface. The light emitted by the source will be a parallel beam after reflection by the points on the curved surface. People can take advantage of this property to manufacture light lamps with parallel lights, such as searchlight, vehicle headlamps, etc.

In mathematics textbooks, mathematical methods can be used to prove this property. In programming foundation course, we need to write a program to verify this property, which involves use of mathematical functions, function decomposition and function design.

B. Discussion in group

For part 2 and 3 practical teaching contents, we adopt the forms of discussion in groups. In experiments classes, we arrange students into groups to discuss algorithm thoughts in examples or exercises, and compile the discussion results into problem-solving reports. Such form of practical teaching can train not only students’ teamwork ability, but also document organizational ability.

C. Explanation of exercises

During the first half hour of each experiment class, we arrange several students to explain exercises. Due to small class teaching, we can ensure that every student has a chance to explain an exercise in a semester.

The explanation of exercises is evaluated by all of the teacher and other students. We design an evaluation table as shown in Table V. This evaluation form and contents can fully examine students’ understanding for programming methods and algorithm thoughts.

| Courseware | Animation, graphics, contents. (20 points). Perfect, 19-20; Very good, 17-18; good, 15-16; general, 12-14; bad, <12. |
| Problem    | Comprehensive analysis, easy to understand. (20 points) |
D. Reply to Course Project

The course project is evaluated in the form of reply. For each group, we randomly select a student to reply the course project. Other groups of students and teacher evaluate the course project at the same time. We design an evaluation table as shown in Table VI, where "Linguistic expression" and "Question and Answer" are rated by the performance of the student who reply the course project.

This evaluation form and contents can fully examine students’ ability of comprehensive applying programming methods and algorithm thoughts in completing course project. 

TABLE VI. EVALUATION TABLE FOR COURSE PROJECT

<table>
<thead>
<tr>
<th>Program Manual</th>
<th>Document contents, document layout. (30 points). Perfect, 27-30; very good, 24-26; good, 21-23; general, 18-20; does not meet the requirements &lt;18.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Novelty</td>
<td>10 points). Very novel, 9-10; relatively novel, 7-8; general, 5-6; nothing new, &lt;5.</td>
</tr>
<tr>
<td>Independent Creation</td>
<td>10 points). Complete independent, 9-10; mostly independent, 7-8; partly independent, 5-6; non-independently, &lt;5.</td>
</tr>
<tr>
<td>Technology Depth</td>
<td>Compact logical structure of the program, sophisticated program, fault-tolerant processing ability. (10 points). Very good, 9-10; good, 7-8; general, 5-6; poor, &lt;5.</td>
</tr>
<tr>
<td>Technology Breadth</td>
<td>Some humanistic tips when running; rich in output contents, etc. (10 points). Very good, 9-10; good, 7-8; general, 5-6; poor, &lt;5.</td>
</tr>
<tr>
<td>Code Specification</td>
<td>Code formatting; correct indentation, spaces and line breaks; more comments. (10 points). Very good, 9-10; good, 7-8; general, 5-6; poor, &lt;5.</td>
</tr>
<tr>
<td>Linguistic expression</td>
<td>10 points). Very fluently, 9-0; relatively fluently, 7-8; general, 5-6; poor, &lt;5.</td>
</tr>
<tr>
<td>Question and Answer</td>
<td>10 points). Quite right, 9-10; relatively correct, 7-8; general, 5-6; poor, &lt;5.</td>
</tr>
</tbody>
</table>

VI. CONCLUSION

The contests driven, online practice oriented, course project enhanced teaching idea, which presented in this article, is accumulated and summarized during our continuously exploring and teaching practice in programming foundation course. 

In the actual teaching process, this teaching idea is showing its advantages gradually. Several aspects are listed below.

a) The novel process of teaching provides students more programming practice opportunities. Through programming practice, students’ abilities and senses have been raised remarkably as a whole, particularly in program designing and debugging, algorithm analyzing and designing.

b) The success of teaching in programming foundation course effectively promotes the study of follow-up courses. For example, through course project, students’ senses in data structure designing and software engineering are improved remarkable, which helps them in the study of follow-up courses, such as data structure, algorithm analysis and design, software engineering, etc.

c) Some of excellent students obtain Rank Awards in Zhejiang Province Collegiate Programming Contests, which spirits up other students’ interest and enthusiasm in programming.

We believe that students will be able to master programming courses as long as teachers continuously carry out exploration and reformation in teaching, develop students’ interest in learning, mobilize students’ enthusiasm in programming. Based on their experience in enjoying successful learning, students’ innovation senses and abilities will be promoted.

REFERENCES