Science Journal of Education
2016; 4(6): 191-197
http://www.sciencepublishinggroup.com/j/sjedu
doi: 10.11648/j.sjedu.20160406.15
ISSN: 2329-0900 (Print); ISSN: 2329-0897 (Online)

The Incremental Teaching Project Design for Project-Based Learning and Its Application in Java Programming Course

Hong Huang

College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, China

Email address:
huanghong@zjut.edu.cn

To cite this article:
Hong Huang. The Incremental Teaching Project Design for Project-Based Learning and Its Application in Java Programming Course. *Science Journal of Education.* Vol. 4, No. 6, 2016, pp. 191-197. doi: 10.11648/j.sjedu.20160406.15

Received: October 31, 2016; **Accepted**: November 24, 2016; **Published**: December 1, 2016

Abstract: Project-Based Learning (PBL) is a widely adopted pedagogy that ignites students' interest of a subject through the investigation of an authentic problem and cultivates their abilities of innovation and self-learning. Traditional PBL often involves a project of complexity and significant scale, which, under the time constraint of a course, is often difficult for students to handle and risks insufficient comprehension of the knowledge and negative impact on learning confidence. This paper proposes an incremental teaching project concept and its design strategy on top of the traditional project-based learning. An incremental teaching project is carefully designed for a Java Programming curriculum, based on which we discuss the dos and don'ts of applying incremental PBL to teaching.

Keywords: Incremental Project-Based Learning, PBL, Incremental Teaching Project Design, Java Programming

1. Introduction

Learning and teaching computer programming courses has been a challenge for higher education institutions [10].

The Project-Based Learning (PBL) pedagogy [1] is implemented through an authentic teaching project that requires students to actively collect information, comprehend related knowledge and skills, design project guidelines and fully execute it. The pedagogy encourages students to take ownership of their learning as well as to think independently and innovate. It is widely used in science and engineering education [6]. While the idea sounds relative straightforward, the success of PBL relies heavily on a carefully designed teaching project and also other learning environments [4]. A carefully designed teaching project is the fundamental base for a success implementation of PBL pedagogy [5] [9].

The implementation of the traditional PBL pedagogy requires plenty of time [2] as it usually takes longer for students to hunt and digest knowledge by themselves than to be spoon-fed by teachers. What's more, the traditional teaching projects are usually quite complex and of significant scale, which makes it a challenge to incorporate them into the often-limited teaching hours of, for example, Introduction to Programming

To better employ the PBL pedagogy in elementary courses with limited teaching hours, some redesign is needed for the traditional practices. And one important modification we can make is the teaching project. A good teaching project should be both comprehensive and with a smooth learning curve. If the project requires all knowledge and skills of the course to start with, it would look formidable and unreachable for students. If instead, we use multiple teaching projects to target different knowledge pieces, a significant amount of time would then be wasted on setting the stage for each new project, which drains the already limited teaching hours; the lack of continuity also goes against cultivating student's ability of knowledge integration [7].

The methodology is to design an incremental teaching project [3], which starts with the basics and adds requirements step by step according to the contents of a course and its teaching sequence. Along with the advancement of the course, newly learnt knowledge and skills can be applied to fulfill the new requirements of the same project. That is to divide a teaching project into different requirement phases, with the added requirements of each phase corresponding to the new teaching contents of each course unit. To accomplish the final phase of the project, almost all the knowledge and skills of the course must be applied. This type of teaching projects is called the incremental teaching projects, and the PBL pedagogy using incremental teaching projects the Incremental PBL

(IPBL: Incremental Project-Based Learning) Pedagogy.

The key to Incremental PBL is the design of the incremental teaching project. With the incremental teaching project, two approaches to implementing the Incremental PBL Pedagogy can be proposed: For courses with abundant teaching hours, a corresponding project phase for each course unit can be assigned and then follow the traditional PBL pedagogy within each unit. For courses with limited teaching hours, firstly assign a project phrase to students, then use the cramming teaching method to present main contents of the course unit, and finally ask students to fulfill the project phase. The first approach is called Heavy Incremental PBL; the second, Light Incremental PBL.

Because the design of the incremental teaching project is crucial, some basic principles should be abided by.

2. The Principles of Incremental Teaching Project Design

When designing an incremental teaching project, the following principles can be referenced:

Principle 1: The teaching project, with all its phases, should cover all or most of the course contents.

Principle 2: The scale and complexity of the teaching project must be moderate to ensure that it can be accomplished within the teaching hours of the course.

Principle 3: The project's requirements can be divided in a progressive way to form an incremental teaching project.

Principle 4: the incremental requirements of the project phases should match the teaching sequence of the course units.

In other words, the knowledge and skills needed to complete the current project phase should go parallel with that of the current teaching phase. When the incremental requirements match students' increasing capabilities, it creates a positive incentive loop that strengthens the efficiency and effectiveness of learning.

Principle 5: In case of need, adjust the teaching sequence of course contents.

Sometimes, the arrangement of contents in a textbook is not identical to the natural progress of the project. In this case, adjustments can be made to the sequence of teaching in order for the project to evolve in a more natural and interesting way.

3. An Example of Incremental Teaching Project Design for Java Programming

The main content of the Java Programming curriculum includes the following sections:

1) Language basics: demonstrate the fundamental features of Java, including data types, variables, operators, and control flow;
2) Classes and objects: describe how to write the classes from which objects are created, and how to create and use the objects;
3) Object-Oriented Programming features: encapsulation, inheritance, methods overloading and overriding, polymorphism;
4) Abstract classes and interfaces;
5) Strings and arrays;
6) Generics and collections;
7) Exceptions;
8) Basic input output programming using I/O Streams, file system operations;
9) GUI programming;
10) Concurrency programming etc.

According to the contents of the course and following the principles proposed, a teaching project is designed: develop a program to assist elementary school students in practicing integer arithmetic (including addition, subtraction, multiplication, and division).

In accordance with the teaching sequence of the Java programming curriculum, the project is designed in an incremental way. Requirements in the earlier phases of the project are relatively simple and easy to realize with fewer knowledge. As the course advances, the project becomes increasingly complicated and students must use new knowledge and skills to accomplish it.

In order to make the earlier phases of the project more interesting and practical, the teaching sequence is changed slightly: bring forward the teaching sequence of data input using keyboard, and also the usage of arrays and String.

Detailed design of this incremental teaching project is presented below.

Project name: Integer arithmetic operations practice software for elementary school students.

Project Description: This software is designed for elementary school students to practice integer arithmetic operations, helping them boost their ability of math calculation. It can be used as a computer aided instruction tool for teaching mathematics.

Phase 1

Requirements Specification: Write a simple Java program for integer addition exercise. Each time the program is run, it will randomly generate 10 exercises, each worth 10 points. For each exercise, the program prints the question and asks the user to input an answer from keyboard. If the input matches the right answer, it prints "Correct!"; otherwise, it prints "Incorrect!" and shows the right answer. Finally, when all 10 exercises are completed, the program adds up all points and prints the total score of the user.

The main class should be named ArithmeticTest1; the source file, ArithmeticTest1. java. Use arguments of the main method to specify required digit of the operands. For example, if running the program from command line as "java ArithmeticTest1 1", all operands of the integer addition exercises will be single digit number.

Corresponding Course Contents: Intro to Java Programming (basic syntax, structured programming design)

Concepts Covered By the Project:

1) Basic structure of Java programs, statements
2) import statements; random number generation (java. util. Random class or and java. lang. Math class);
3) Control flow statement

4) main method arguments
5) Keyboard input
6) String-to-int conversion;

Phase 2

Requirements Specification: Write a Java program for integer arithmetic (including addition, subtraction, multiplication, and division) exercise. Each time the program is run, it will randomly generate 10 exercises, each worth 10 points. For each exercise, the program prints the question and asks the user to input an answer from keyboard. Finally, when all 10 exercises are completed, the program prints all the exercise questions, the right answers, together with the user's answers and total score. The output results should be aligned.

The main class should be named ArithmeticTest2; the source file, ArithmeticTest2. java. Use arguments of the main method to specify both the arithmetic type and the required digit of the operands. If the first argument is "+", do addition exercises; "-", subtraction; "x", multiplication; "/", division. If the first argument is "r", then each exercise's arithmetic type will be generated randomly. For example, if running the program from command line as "java ArithmeticTest2 + 2", then the arithmetic type will be integer addition, and all operands of the integer addition exercises will be of two digit.

Ensure that minuend is always greater than or equal to subtrahend when the arithmetic type is subtraction; and divisor nonzero when the arithmetic type is division.

When the arithmetic type is division, the answer requires both quotient and remainder.

Corresponding Course Contents: Array, String class

Concepts Covered By the Project:

1) All concepts of the previous project phase.
2) Array
3) String
4) Usage of method printf () (by self-learning)

Phase 3

Requirements Specification: Write a simple Java program for integer addition exercise. Each time the program is run, it will randomly generate 10 exercises, each worth 10 points. For each exercise, the program prints the question and asks the user to input an answer from keyboard. Finally, when all 10 exercises are completed, the program prints all the exercise questions, the right answers, together with the user's answers and total score.

This time, design addition as a class, named "Addition". Operands and user's answer can be declared as Addition's instance variables. Methods of this class should include the add operation, exercise question printing, judging whether user's answer is correct or not, etc. Each addition exercise is an object created using Addition's constructor with the two operands randomly initialized.

The main class should be named ArithmeticTest3; the source file, ArithmeticTest3. java. Use arguments of the main method to specify required digit of the operands. For example, if running the program from command line as "java ArithmeticTest3 1", then all operands of the integer addition exercises will be single digit number.

Corresponding Course Contents: Object-Oriented Programming Basics

Concepts Covered By the Project:

1) All concepts of the previous project phase.
2) The design of typical Java classes
3) Creating and using objects

Phase 4

Requirements Specification: Write a Java program for integer arithmetic (including addition, subtraction, multiplication, and division) exercise. Each time the program is run, it will randomly generate 10 exercises, each worth 10 points. For each exercise, the program prints the question and asks the user to input an answer from keyboard. Finally, when all 10 exercises are completed, the program prints all the exercise questions, the right answers, user's answers, and user's total score. The output results should be aligned.

This time, an abstract class, "Operation" is to be designed, and four subclasses of "Operation", "Addition", "Subtraction", "Multiplication", and "Division" are also to be designed. Use polymorphism to write the main class and try to make the code as concise as possible.

The main class should be named ArithmeticTest4; the source file, ArithmeticTest4. java. Use arguments of the main method to specify both the arithmetic type and the required digit of the operands. If the first argument is "+", do addition exercises; "-", subtraction; "x", multiplication; "/", division. If the first argument is "r", then each exercise's arithmetic type will be generated randomly. For example, if running the program from command line as "java ArithmeticTest2 + 2", then the arithmetic type will be integer addition, and all operands of the integer addition exercises will be of two digit.

Ensure that minuend is always greater than or equal to subtrahend when the arithmetic type is subtraction; and divisor nonzero when the arithmetic type is division.

When the arithmetic type is division, the answer requires both quotient and remainder.

After finishing the above task, change the abstract class Operation into an interface Operation Interface, write four arithmetic type class to implement the interface, and refactor the program. In order to distinguish between the two implementations, the relevant class names are defined as Addition Using Interface, Subtraction Using Interface, Multiplication Using Interface, and Division Using Interface. The main class should be named Arithmetic Test 4 Using Interface.

Corresponding Course Contents: Object-Oriented Programming (encapsulation, inheritance, abstract classes, method overload and override, interfaces, polymorphism)

Concepts Covered By the Project:

1) All concepts of the previous project phase.
2) Inheritance, abstract classes
3) Method overload and override
4) Interfaces
5) Polymorphism

Phase 5

Requirements Specification: Write a Java program for integer arithmetic (including addition, subtraction, multiplication, and division) exercise. Each time the program

is run, it will randomly generate an exercise question, the program prints the question and asks the user to input an answer from keyboard, then asking the user whether to continue, if user choose yes, continue with the exercise, otherwise, the program displays all the questions, standard answer, user's answer, and user's total score (The points for each question is 100.0 divided by the number of questions).

The main class should be named ArithmeticTest5; the source file, ArithmeticTest5. java. Use arguments of the main method to specify both the arithmetic type and the required digit of the operands. If the first argument is "+", then do addition exercises; "-", subtraction; "x", multiplication; "/", division. If the first argument is "r", then each exercise's arithmetic type will be generated randomly. For example, if running the program from command line as "java ArithmeticTest5 + 2", then the arithmetic type will be integer addition, and all operands of the integer addition exercises will be two digit number.

Ensure that minuend is always greater than or equal to subtrahend when the arithmetic type is subtraction; and divisor nonzero when the arithmetic type is division.

When the arithmetic type is division, the answer requires both quotient and remainder.

Corresponding Course Contents: Generic and Collections

Concepts Covered By the Project:

1) All concepts of the previous project phase.
2) Generic
3) Collection classes (using ArrayList or Vector to save arithmetic questions).

Phase 6

Requirements Specification: Write a Java program for integer arithmetic (including addition, subtraction, multiplication, and division) exercise. Each time the program is run, it will randomly generate an exercise question, the program prints the question and asks the user to input an answer from keyboard, then asking the user whether to continue, if user choose yes, continue with the exercise, otherwise, the program displays all the questions, standard answer, user's answer, and user's total score (The points for each question is 100.0 divided by the number of questions).

When starting the program, if no command line arguments are specified or the given arguments are insufficient, the user is prompted with a reminder, and the program stops running.

If the specified arithmetic type is not one of "+, -, X, /, r", the program prompts " arithmetic type error!"

If user's answer is not numeric data, the program prompts "Input data format error! You must enter numeric data!"

If the answer entered is outside the possible range (for example, if the user entered an answer greater than 20 when doing a 1-digit addition), the program prompts "The answer you entered is out of the possible range! It should be less than xx." Use a self-defined exception class (NumberTooBigException) to deal with this problem.

If the user enters an answer other than numeric data, or if the answer is out of range, the user should be allowed to re-input an answer.

The main class should be named ArithmeticTest6; the source file, ArithmeticTest6. java.

Corresponding Course Contents: Exceptions

Concepts Covered By the Project:

1) All concepts of the previous project phase.
2) Exception handling
3) User-defined exception classes

Phase 7

Requirements Specification: Write a Java program for integer arithmetic (including addition, subtraction, multiplication, and division) exercise. Each time the program is run, it asks for username, then randomly generate an exercise question, the program prints the question and asks the user to input an answer from keyboard, then asking the user whether to continue or not, if user choose yes, continue with the exercise, otherwise, the program displays all the questions, standard answer, user's answer, and user's total score (The points for each question is 100.0 divided by the number of questions), and at the same time, writes these contents to a text file (named as username+datetime+.his. eg. Sam20161020032630.his) in the directory named after the username. Users can use Notepad to view the contents of the file later.

The program ought to provide complete exception handling.

The main class should be named ArithmeticTest7; the source file, ArithmeticTest7. java.

Corresponding Course Contents: I/O Streams and file system operations

Concepts Covered By the Project:

1) All concepts of the previous project phase.
2) I/O Streams (usage of FileWriter class, BufferedWriter class, etc.)
3) File system operation (usage of class File, etc.),

Phase 8

Requirements Specification: Write a Java program for integer arithmetic (including addition, subtraction, multiplication, and division) exercise with GUI. After the program starts, user enters the user name, and then select arithmetic type (addition, subtraction, multiplication, division, random), digits of the operands in the start window (these settings have initial default values; after the first running of this program, the latest set values will become the default values). When click the "Start" button, the program randomly generates 10 exercises, and displays them in the window. After the user enters the answers and Click "Submit" button, it displays the standard answers and the user score. At the same time, it writes these contents to a text file (named as username+datetime+. his. eg. Sam20161020032630.his) in the directory named after the username. User can use this program to view the contents of the file later in another window.

Once submitted, user can click "Continue" button to start another round practice, or just close the window to exit.

The graphic user interface should be beautifully and reasonably designed.

The main class should be named Arithmetic Test 8; the source file, ArithmeticTest8. java.

Corresponding Course Contents: Java GUI programming

Concepts Covered By the Project:

1) All concepts of the previous project phase.
2) GUI programming (using swing to construct the GUI, GUI events handling.)
3) Complex requirements,

Phase 9

Requirements Specification: On the basis of phase 8, add a time limit for user doing arithmetic exercises. User can set a time limit (time unit is seconds) in the window together with the parameters as those in phase 8. When click the "Start" button, the time left for this practice will be dynamically displayed. If time runs out, the program will submit automatically just as if the user clicked the "Submit" button, and it displays the standard answers and the user score on the window. At the same time, it writes these contents and the time used to a text file (named as username+datetime+. his. eg. Sam20161020032630.his) in the directory named after the username. User can use this program to view the contents of the file later in another window.

Once submitted, user can click "Continue" button to start another practice, or just close the window to exit.

The main class should be named ArithmeticTest9; the source file, ArithmeticTest9. java.

Corresponding Course Contents: Concurrency programming

Concepts Covered By the Project:

1) All concepts of the previous project phase.
2) Thread (using a thread to show the time left).
3) Multi-Thread programming,

A summary of the of the project phases and its covered course contents is shown in table 1.

Table 1. *Project phases and corresponding course contents.*

Phase Name	Consolidated Contents	New Covered Contents
Phase 1	-	language basics: basic syntax, data type and variables, structured programming design
Phase 2	Concepts of the previous project phase	Strings and arrays;
Phase 3	Concepts of the previous project phases	Classes and objects:
Phase 4	Concepts of the previous project phases	Inheritance, abstract classes and interfaces, polymorphism
Phase 5	Concepts of the previous project phases	Generics and collections
Phase 6	Concepts of the previous project phases	Exception handling
Phase 7	Concepts of the previous project phases	Basic input output using I/O streams, file system operations
Phase 8	Concepts of the previous project phases	GUI programming
Phase 9	Concepts of the previous project phases	Concurrency programming

4. Using the Incremental Teaching Project to Implement PBL in Java Programming Course

The design of the above-mentioned incremental teaching project embodies the design principles proposed previously, the requirements specification of the project are clear and easy for students to understand; the scale and complexity of the project are moderate, suitable for the limited teaching hours to complete.

This project design can be used in the teaching of Java programming. It is also easily generalizable and customizable based on characteristics of different programming languages for either lectures or lab sessions.

Depending on the number of lessons, this incremental teaching project can be used in conjunction with different teaching methods.

If the course's teaching hour is sufficient, the incremental PBL pedagogy can be simply adopted. At the beginning of each course unit, assign the corresponding phase of the project to students, ask them to search and get information, learn the relevant knowledge, complete the project, and then make summary and evaluation. This approach is called the heavy incremental PBL pedagogy.

If the teaching hour is relatively insufficient, then the teaching efficiency must be weighed more heavily on. It is a good choice to combine the incremental PBL pedagogy together with the Cramming teaching method: at the beginning of each course unit, assign the corresponding phase of the project to students. Then the teacher presents the main points of knowledge needed to complete the new requirements of the project, and instruct students to study some of the contents by themselves. They can then hop on to the project; finally, a summary and evaluation are made. This approach is called the light incremental PBL pedagogy.

Beginning in 2013, the authors start to use the incremental teaching project in the teaching of Java programming courses. Since the school's Java programming course only have 48 teaching hours, the light incremental PBL pedagogy is adopted and achieved good results. This can be proved by a statistical analysis of the test scores of students in Java programming course over the last few years as show in table 2. The students' average scores in the course increased from 70 to about 80 after the introduction of the new teaching methods. Their practical programming ability have been Significantly improved.

A survey is also conducted to find out if the students accept the new teaching method at the end of the course by ask students to fill a questionnaire simply contains three items as show in table 3. The result show in Table 4 is quite reassuring that most of the students like this way of teaching. Among those who have positive attitudes, most of the reasons they give is that the project phases are interesting and motivative, and they gain more confidence and a sense of accomplishment each time they finish the task. Among those who have negative attitudes, most of the reasons they give is that they have to do too much programming work.

Table 2. *Statistical analysis of test scores.*

Average Scores / Course year	Objective Questions (total 40)	Programming Questions (total 60)	Total Scores (total 100)
2012 (Conventional Method)	31	39	70
2013 (New Method)	28	47	75
2014 (New Method)	31	48	79
2015 (New Method)	32	48	80

Table 3. *Questionnaire of students satisfaction of the incremental PBL.*

Do you like the incremental PBL pedagogy? (select Yes/Not care/No)	Reasons	If you have any suggestion, write down here.

Table 4. *Results of the satisfaction survey.*

Satisfaction degree	Proportion
Yes	82%
Not care	15%
No	3%

When using the PBL pedagogy, attention needs to pay to some of problems [8]. For instance, this pedagogy sometimes leads students to focus too much of their attention on the implementation of the project, while ignoring the mastery of the basic concepts knowledge. When doing written examinations, because no programming IDE can be used, and no other materials can be referenced, they would lose many points in questions that test basic concepts, although they are quite capable in implementing a programming project. This problem can be alleviated by introducing tests of basic concepts in class or homework, which serves as a good complementary strategy alongside the incremental teaching project. Table 2 shows that in the first year introducing PBL pedagogy, the average scores declined 3 points, but improved after taking measures.

When using this incremental teaching project, combining the Incremental PBL with MOOC (Massive Open Online Courses) will achieve better results, and that is the future work to do.

5. Conclusions

Project-Based Learning, when properly implemented, can be both effective and efficient in achieving the desired goal of teaching. Crucial to its success, however, is to design a good teaching project that adapts to the various requirements and constraints of different course curriculums. For applied programming courses, for example, the traditional PBL approach would suffer great disadvantages given the need to cover a wide range of knowledge and skills within limited teaching hours.

Through years of exploration and practice, we've come up with the concept and design of incremental teaching project and formed a unique incremental PBL pedagogy with a proven track record. This paper discusses the design principles of incremental teaching project and presents a concrete example in Java programming. It will be a great gratification if this experience could help remove the obstacles of applying PBL in different curriculum designs and better promote efficient active learning.

Acknowledgement

This work was supported by the Teaching Reform Project of Higher Education in Zhejiang Province, China (Project number: kg2013057): Research and Practice of Classroom Teaching Reform of "Java Programming" Based on Small Incremental Teaching Project.

References

[1] https://en.wikipedia.org/wiki/Project-based_learning#Overcoming_Obstacles_and_Criticisms.

[2] Kun Ma, Hao Teng, Lixin Du, Kun Zhang. Project-Driven Learning-by-Doing Method for Teaching Software Engineering using Virtualization Technology. International Journal of Emerging Technologies in Learning, 2014 Volume 9, Issue 9: 26-31.

[3] Huang Hong, Zhao Xiaomin, Zhang Fan, Ye Lei, Wang Ben. Application of task-driven pedagogy in teaching Java programming [J], Computer times, 2012 (4):49-51.

[4] Muhamad Hugerat. How teaching science using project-based learning strategies affects the classroom learning environment. Learning Environments Research, October 2016, Volume 19, Issue 3, pp 383–395.

[5] C. S. Keator, D. Vandre A., M. Morris. The Challenges of Developing a Project-Based Self-Directed Learning Component for Undergraduate Medical Education. pp 1–5.

[6] Umair Mujtaba Qureshi, Zuneera Aziz, Faisal Karim Shaikh, Nafeesa Bohra, Aftab Ahmed Memon. Project Oriented Problem Based Learning: A Wireless Sensor Network Perspective. Wireless Personal Communications, June 2014, Volume 76, Issue 3, pp 463–477.

[7] Qiang Hao, Robert Maribe Branch, Lucas Jensen. The Effect of Precommitment on Student Achievement Within a Technology-Rich Project-Based Learning Environment. TechTrends, September 2016, Volume 60, Issue 5, pp 442–448.

[8] K. J. Chua, W. M. Yang, H. L. Leo. Enhanced and conventional project-based learning in an engineering design module. International Journal of Technology and Design Education, November 2014, Volume 24, Issue 4, pp 437–458.

[9] Carlos Vega, Camilo Jiménez, Jorge Villalobos. A scalable and incremental project-based learning approach for CS1/CS2 courses. Education and Information Technologies, June 2013, Volume 18, Issue 2, pp 309–329.

[10] Wilfred W. F. Lau Email author Allan H. K. Yuen. The impact of the medium of instruction: The case of teaching and learning of computer programming. Education and Information Technologies June 2011, Volume 16, Issue 2, pp 183–201.