ASEAN Journal of Education

Journal homepage: http://aje.research.dusit.ac.th/

Learning gains in web programming course with automatic code generator program using a structured flowchart

Chanchai Supaartagorn[1]*

[1] *Department of Mathematics Statistics and Computer, Faculty of Science, Ubon Ratchathani University, 34190, Thailand*

Article info

Article history:
Received: 11 April 2019
Revised: 12 June 2019
Accepted: 14 June 2019

Keywords:
Learning gain, Normalized gain, Programming course, Flowchart

Abstract

Following normalized gain concept, including class average normalized gain, single student normalized gain, single test item normalized gain, and conceptual dimensional normalized gain. The aim of this research was to assess learning gain for the computer programming course using automatic code generator using a structured flowchart (CGF tool). Moreover, the research aims to compare the understanding of the programming course between the experimental group and the control group. Data were collected from 58 students enrolled in web programming course. The data were collected through pre-test and post-test and then analyzed with descriptive statistics and inferential statistics using t-test and normalized gain. The findings, notably revealed that 1) Post-test score average of an experimental group was significantly higher than a control group at the level of 0.05. 2) Class average normalized gain of the experimental group was in the medium gain <g> = 0.59 and control group was in the low gain <g> = 0.23. 3) Single student normalized gain results showed that 3.1) an experimental group was in the high gain (39.02 percent), medium gain (46.35 percent) and low gain (14.63 percent). 3.2) a control group was in the high gain (5.88 percent), medium gain (29.41 percent) and low gain (64.71 percent). 4) Single test item normalized gain was in the high gain (3 items), medium gain (6 items) and low gain (1 item). 5) Conceptual dimensional normalized gain was in the high gain (1 concept) and medium gain (3 concepts).

Introduction

Computer programming course is one of the core subjects in a wide variety of computer-related degree programs, including Information Technology, Computer Science, Computer Engineering. Students learn how to write programs using common programming languages. When they are learning in a program, novices face difficulties in understanding the syntax and logic of programming language. This is a challenge for instructors to find ways to teach their students. In a computer programming class the objective is not only for students to be able to write programs, but also to be accurate in

* Corresponding author
Email: chanchai.s@ubu.ac.th

the syntax and logic of programs.

Several research articles have shown a decline in the number of students in computer programming courses. For example, a study on an active learning approach at the University of Mary Washington confirmed the notion that learning how to program is considered to be a difficult task for the majority of students and this has been a prime reason for students' dropping out of computer courses (Karen, 2008). A study estimated that between 25 to 80 percent of students dropped out of their first year computer science classes due to the difficulty they faced in learning how to program (Janet, & Tony, 2002). Some environments developed for novice students, such as Scratch or Alice, have considered a visual environment and error-free syntax as key points for motivation and support. However, these environments lack a way of expressing a solution in a way familiar to novice students, thus the user cannot easily verify whether the current solution matches their intention (Edgar, 2013).

To alleviate the problem, one of the solutions is to let students present the programming logic in the form of a flowchart. The flowchart could be used to organize the sequence of basic structure steps, including sequence structure, selection structure, and iteration structure. In addition, the complicated structure steps could be described easily in the visual form of easy-to-understand, including stacking structure, nested structure, and nesting structure. A study on teaching computer programming to adult students at Tairawhit Polytechnic, New Zealand showed that 40% of the beginners preferred to use a flowchart to understand programming, 40% wanted to use pseudo code, and 20% wanted to use a real language (Min, 2003). There are many research studies on using flowchart for computer programming courses. The animated flowchart for system programming course of 3rd-year students in Computer Science and Engineering at Walchand Institute of Technology, India, was analyzed using pre-test and post-test. One group pre-test, post-test model is considered to test the effectiveness of this activity. The result showed the significantly higher scores in post-test than in pre-test (Sunita, 2015). The visual programming using flowchart is a tool that allows the programmer to write the program in the format of flowchart, then compiles and run without the coding step. The results showed the powerfulness, easiness, and user-friendliness of the proposed system. It can be used as a tool for teaching computer programming (Kanis, & Somkiat, 2006). In addition, our previous research implemented a tool serving as an automatic code generator using a structured flowchart. The system evaluations showed the average values of the satisfaction levels were 4.48 and 4.27 for the experts and the general users, respectively (Chanchai, 2017). However, the evaluation results of these researches didn't show the effectiveness of teaching methods. Moreover, the evaluation results do not reflect the effectiveness of each topic. The purposes of this research study are 1) to compare the understanding of the programming course between using automatic code generator via flowchart (experimental group) and traditional study (control group); 2) to assesses students' learning gain in programming course using automatic code generator via flowchart. This study uses, a web application for automatic code generator using a structured flowchart in teaching and learning. There are two sample groups of this research: experimental group and control group. Ten questions were performed as pre-test and post-test. The outcome data were analyzed with t-test and normalized gain statistics, including pre-test/post-test scores comparison, class average normalized gain, single student normalized gain, single test item normalized gain and conceptual dimensional normalized gain. The result of the study will help to provide an effective learning plan for the computer programming course by using the flowchart in teaching and learning.

Literature Review

This first part presents a review of the literature with regard to the overview of web application for automatic code generator using a structured flowchart and then, the normalized gain concept is provided.

Web application for automatic code generator using a structured flowchart

This research uses an application called "CGF tool" for teaching web programming courses during the first semester, of the 2017 academic year. It is a tool for automatic code generator using a structured flowchart based on web application. Programmers drag and drop symbols to draw a visual flowchart by using the drawing tool. There are 9 parts of the drawing tool: basic symbols, sequence symbol templates, selection symbol templates, iteration symbol templates, stacking symbol templates, nested selection symbol templates, nested loop symbol templates, nesting (selection-loop) symbol templates, and nesting (loop-selection). Figure 1 shows the drawing tool example: (a) basic symbols and (b) sequence symbol templates (Chanchai, 2017).

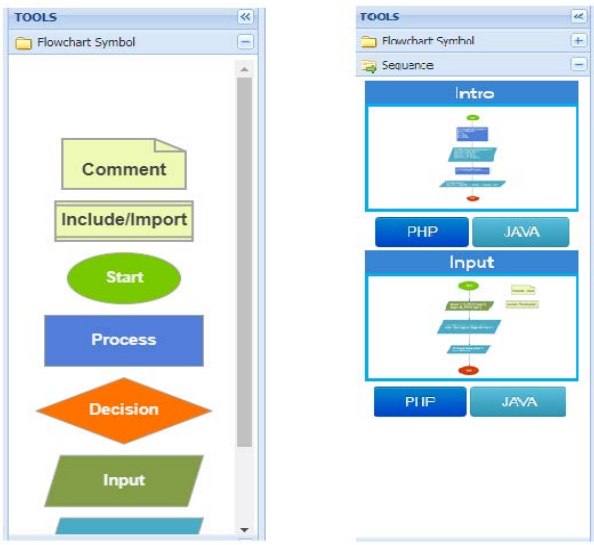

Figure 1 The drawing tool example.

Then, programmers can convert the flowchart into JSON format and Java/PHP source code. There is a debugging module to find errors of the source code in the last step. Figure 2 shows the example of flowchart as converted into PHP source code (Chanchai, 2017)..

Normalized gain concept

The normalized gain was introduced by Richard R. Hake, professor of physics at Indiana University Bloomington. He proposed a method for assessing learning outcomes from pre-test and post-test, has become the standard measure of the effectiveness of a course in promoting conceptual understanding. His research compared between interactive engagement (IE) and traditional methods (T) of mechanics test data for introductory physics courses (Richard, 1997). Average normalized gain <g> measures the fraction of the available improvement that is obtained, defined as

$$\langle g \rangle = \frac{(\%\langle Sf \rangle - \%\langle Si \rangle)}{(100 - \%\langle Si \rangle)}$$

Where <Sf> and <Si> are the final (post) and initial (pre) class average.

The research used the Force Concept Inventory (FCI) for 62 introductory courses enrolling a total number of student N = 6542. This amount divided into IE method (N =4458) and T method (N = 2084). In addition, there are 3 levels of normalized gain, defined as

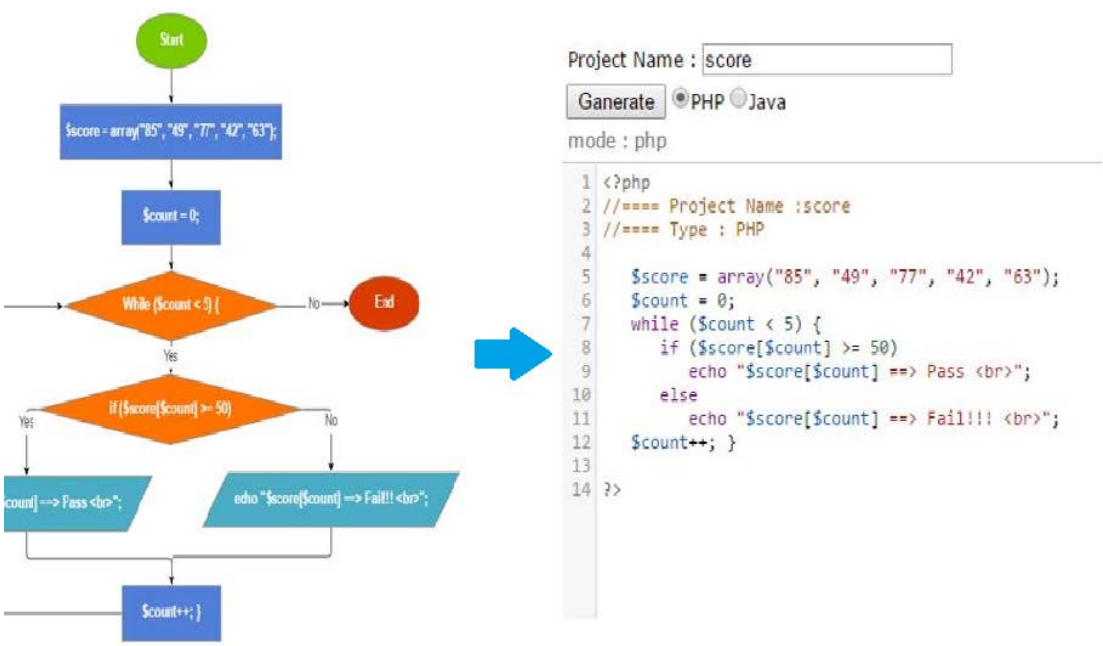

Figure 2 The example of converting flowchart into PHP source code.

"High gain" courses as those with <g> ≥ 0.7

"Medium gain" courses as those with 0.7 > <g> ≥ 0.3

"Low gain" courses as those with <g> < 0.3

The research results are displayed in Figure 3.

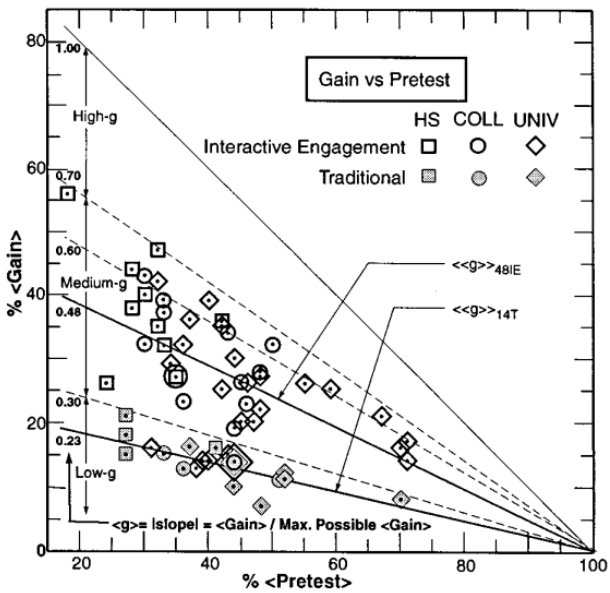

Figure 3 %<Gain> vs %<Pretest> score on FCI test.

The result showed that the Y-axis is an actual gain (%<Gain>) and the X-axis is %<Prestest> and normalized gain (<g>). <<g>>48IE is the average normalized gain of IE method is 0.48 at medium gain level. <<g>>14T is the average normalized gain of T method is 0.24 at low gain level. The square symbol, circle symbol, and diamond symbol are the high school, college, and university. In addition, the normalized gain can be divided into 4 types as follows:

Class average normalized gain

Class average normalized gain is the ratio of the actual gain to the maximum possible gain of the whole class. The ratio is used to display the improvement of students' learning outcomes in the whole class. Instructions can compare class average normalized gain between new technique teaching and traditional teaching.

Single student normalized gain

Single student normalized gain is an average normalized gain of each student. This value is based on the pre-test and post-test scores of each student. Instructors can know the improvement in learning outcomes of individual students. In particular, students with normalized gain at a low level.

Single test item normalized gain

Single test item normalized gain is an average normalized gain of each item. This value determines the student's understanding of each item. Instructors can design and improve examination contents.

Conceptual dimensional normalized gain

Conceptual dimensional normalized gain is an average normalized gain of each concept. Normally, the examination test will measure the comprehension in several concepts. This value shows the students' understanding of each concept.

This is based on CGF tool and normalized gain concept in order to reveal the degree this tool has an effect of the students' learning gain. With this purpose, the research questions that guide this study are as follows;

1. What is the difference of post-test score between experimental group and control group?

2. To what degree do students' learning gain after using CGF tool for teaching web programming courses of

(a) Average class normalized gain,
(b) Single student normalized gain,
(c) Single test item normalized gain, and
(d) Conceptual dimensional normalized gain?

The next section describes the research methodology, including research context, participants of the study, data collection, instruments and procedure, and data analysis.

Method

The research is conducted as a quantitative research. The study was conducted on a web programming course during the first semester, of the 2017 academic year at Ubon Ratchathani University, Thailand. The curriculum of Information Technology offers web programming courses as a compulsory three-credit course for all enrolled students; given by two instructors based on identical curriculum and syllabus, and prepared in accordance with the Thai Qualifications Framework for Higher Education (TQF: HEd). The course aims to teach the basic knowledge of Internet programming, installation. The syntax and statements, such as variables and constant, operator and expression, condition and looping, functions, input form, session and cookie, programming connect to database. Moreover, it aims for students to gain web programming skills in both the client-side and server-side web programming. Learning course management is based on both lecturing and laboratory.

During these sessions, instructors employed direct instruction and demonstration.

Participants of the study

Web programming courses are divided into 2 sections of 81 students (section 1 with 64 students and section 2 with 17 students). Among 81 students enrolled, 58 students were selected after checking the completeness of the pre-test and post-test, representing a response ratio of 71.60% (section 1 had 41 students and section 2 had 17 students). Therefore, section 1 was assigned to be an experimental group and section 2 was defined as a control group.

Data collection instruments and procedure

Data were collected through pre-test and post-test. There were 10 items of program coding, which divided into 4 topics as follows: 1) 3 items of selection statement 2) 3 items of loop statement 3) 2 items of nested statement and 4) 2 items of nesting statement. The procedure of the study is as follows:

1. Data collected through pre-test in both experimental group and control group of each topic.
2. In the learning process, experimental group used the CGF tool while control group used a traditional method.
3. Data collected through post-test in both experimental group and control group of each topic.
4. Check the completeness and correctness of pre-test and post-test.

The detailed explanation of data analysis is in the following subsection.

Data analysis

The analysis of collected data was conducted using IBM SPSS software and Microsoft Excel. The data is analyzed through both descriptive statistics and inferential statistics. The statistics used in studying the data includes frequency, mean, standard deviation, and t-distribution. The normalized gain analysis is chosen since it aims to assess the learning outcomes from pre-test and post-test and will check the reliability of pre-test and post-test. A popular approach to measure reliability is to use the coefficient alpha or Cronbach's alpha (Malhotra, 2006). Moreover, difficulty index and discrimination index are used for reliability test. The next section presents the findings retrieved from the data analysis.

Results

This section presents the findings of the study in accordance with the two research questions. The findings of the first research question of difference in the post-test score between experimental group and control group is shown in Table 1.

Table 1 Descriptive statistics of post-test score between experimental group and control group.

Group	Number of students	$\bar{X}$	S.D.	T
Experimental	41	80.27	15.14	5.46**
Control	17	42.76	26.57	p-value = .000

Table 1 indicates that post-test of experimental group have a mean score of 80.27 and with 15.14 standard deviation. Moreover, the post-test of an experimental group was significantly higher than a control group at the level of 0.05 (t-value=5.46, p-value=.000).

Then, considering the second research question that investigates the students' learning gain after using CGF tool for teaching web programming courses. Table 2 presents the findings of average class normalized gain.

Table 2 Descriptive statistics of pre-test and post-test with t-test, average class normalized gain on both groups.

Group	Statistics	Average score		t	Class normalized gain
		Pre-test	Post-test		
Experimental	$\bar{X}$	51.73	80.27	12.31**	0.59
	S.D.	17.47	15.14	p-value = .000	(medium gain)
Control	$\bar{X}$	25.53	42.76	3.15*	0.23
	S.D.	18.75	26.57	p-value = .012	(low gain)

Table 3 below indicates that the experimental group hasa post-test score (80.27) significantly higher than pre-test score (51.73) at the level of 0.05 (t-value =12.31, p-value=.000). The control group hasa post-test score (42.76) significantly higher than the pre-test score (25.53) at the level of 0.05 (t-value=3.15, p-value=.012). Moreover, an average class normalized gain of the experimental group is in the medium gain <g> = 0.59 and control group is in the low gain <g> = 0.23.

The findings of single student normalized gain on both groups is displayed in Figure 4 and Figure 5.

Figure 4 indicates that the experimental group was in the high gain (39.02%), medium gain (46.35%) and low gain (14.63%). While Figure 5 indicates that the control group was in the high gain (5.88%), medium gain (29.41%) and low gain (64.71%). Where ◆ and ▲ are the normalized gain of each student in the experimental group and the control group, respectively.

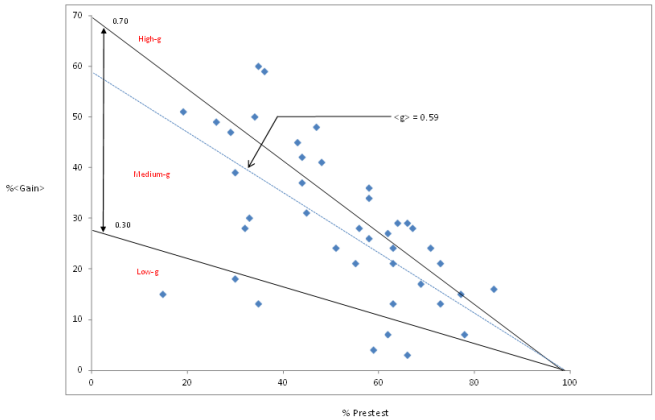

Figure 4 Single normalized gain of experimental group.

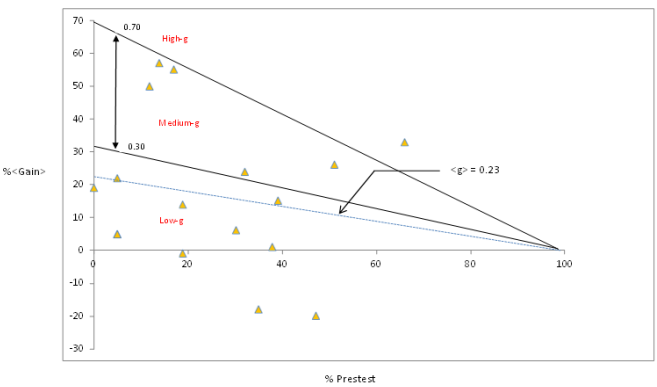

Figure 5 Single normalized gain of control group.

The finding of single item normalized gain on the experimental group is displayed in Figure 6.

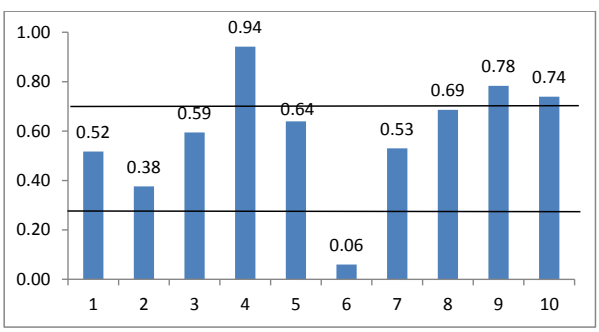

Figure 6 Single item normalized gain.

The 10 questions of pre-test/post-test are as follows:

1. Selection structure of dual-alternative statements.
2. Selection structure of multiple-alternative statements.
3. Selection structure of switch-case statements.
4. Loop structure of For statements.
5. Loop structure of While statements.
6. Loop structure of Do...While statements.
7. Nested structure of nested loop.
8. Nested structure of nested selection.
9. Nested structure of nesting (loop-selection).
10. Nested structure of nesting (selection-loop).

The majority of single item normalized gains are in the medium gain (6 items). The only item in the low gain is the question about the loop structure of Do...While statements.

Finally, the findings of concept dimensional normalized gain on the experimental group is displayed in Table 3.

Table 3 Comparing of pre-test, post-test and normalized gain of each concept.

Concept	%pre-test	%post-test	Normalized gain <g>
Selection structure	65.53	82.11	0.48 (medium gain)
Loop structure	69.67	84.15	0.48 (medium gain)
Nested structure	22.07	67.93	0.59 (medium gain)
Nesting structure	33.78	84.02	0.76 (high gain)

Table 3 indicates the conceptual dimensional normalized gain. The majority are in the medium gain. There only concept in the high gain is the topic about nesting structure.

Reliability is evaluated by assessing the internal consistency of the items representing each pre-test and post-test using Cronbach's alpha. The value of coefficient alpha or Cronbach's alpha with the range of greater than 0.60 is considered acceptable and good (Hair, Babin, Money, & Samouel, 2003), as indicated in Table 4. In addition, we have calculated the difficulty index and discrimination index of pre-test and post-test. The result of the difficulty index ranged from 0.29 to 0.71, which is considered as optimal level of difficulty. The result of the discrimination index ranged from 0.57 to 1.00, which is considered as optimal level of discrimination. The detail of each question is shown in Table 5.

Table 4 Reliability statistics.

Cronbach's Alpha	Number of items
0.83	10

Table 5 The difficulty index and discrimination index of pre-test and post-test.

Question item	Difficulty index	Discrimination index
1	0.71	0.57
2	0.48	0.96
3	0.52	0.96
4	0.59	0.84
5	0.59	0.85
6	0.49	0.97
7	0.29	0.58
8	0.50	1.00
9	0.44	0.87
10	0.41	0.83

Discussion

The current study assesses student learning outcome in a programming course using automatic code generator via flowchart. Moreover, we aim to compare the understanding of the programming course between the experimental group and control group. With this aim, data from 58 students are analyzed through descriptive statistics and normalized gain concept. The findings are discussed and concluded separately for each of the research questions.

Flowchart represents an important tool in teaching programming courses. The post-test score of an experimental group is significantly higher than a control group. Similarly, Danial et al (2016) develops an online formative assessment game that incorporated a Flowchart-based Intelligent Tutoring System (FITS), in order to improve students' performance in learning computer programming. The result shows that the mean values of the post-test scores for the experimental and control groups were 78.15 and 59.40, respectively.

In terms of average class normalized gain, a class normalized gain of the experimental group is higher than the control group. Similarly, Joshua et al (2016) used an evidence-based instructional strategies for enhancing conceptual learning in introductory physics courses. The research compared learning gain between "interactive engagement" (IE) and traditional lecture-based instruction (TRAD). The topics to compare where Force Concept Inventory (FCI) and Force and Motion Conceptual Evaluation (FMCE).The result shows that the class normalized gain of IE is significantly higher than TRAD on both topics. In FCI topic, normalized gain of IE and TRAD were 0.39 and 0.22, respectively. In FMCE topic, normalized gain of IE and TRAD were 0.52 and 0.19, respectively. Moreover, the single student normalized gain was used to investigate the distribution of student' learning gain. The research of Joshua et al shows the distribution of normalized gain on both IE and TRAD as well. The results indicate that the class size does not impact gains.

In result of the single item normalized gain, the current study finds that Do...While statements are in the low gain level. Since the while and Do...While statements are similar to conditional statements, which are blocks of code that will execute if a specified condition results is true. The major difference between the two statements is Do...While loop will always execute once, even if the condition is never true. Programmers or novice users may be confused in their use.

In result of the conceptual dimensional normalized gain, the current study finds that no topics are in the low gain level. Students perceive programming concept at a fairly substantial level in this study. The flowchart helps to develop the complexity of the programming concepts.

Conclusion and Future Research

This current study investigated the learning gains in web programming courses with automatic code generator using a structured flowchart, and concludes with the significant contributions as presented in the previous section. The application called “CGF tool” can be used as a tool for teaching a computer programming course. It is superior to the traditional programming system in readability, easy-to-debug, effectiveness, and user-friendliness. In addition, gain analysis using the normalized gain calculations can tell us many things about programming courses as follows: 1) Class normalized gain value of an experimental group is significantly higher than a control group. 2) Single student normalized gain value is in the medium gain level 3) Single item normalized gain value is in the medium gain level and 4) Conceptual dimensional normalized gain value shows the effectiveness of the tool for teaching programming courses in every concept. In future research, we will add other factors that may impact learning gains, such as class size, curriculum group (4 years program and 2 years continuing program), grade levels, etc.

Acknowledgements

This research was supported by the Office of Research, Academic Services, and Art & Culture Preservation at Ubon Ratchathani University. We thank Asst.Prof.Dr. Banthom Suraporn and Mrs. Karnarnong Nittharak who provided insight and expertise.

References

Chanchai, S. (2017). Web application for automatic code generator using a structured flowchart, *8th IEEE International Conference on Software Engineering and Service Science,* November 24-26, Beijing, China.

Danial, H., Rodina, B. A., Moslem, Y., Moein, F., Shi-Jinn, H., & Heuiseok, L. (2016). Applying an online game-based formative assessment in a flowchart-based intelligent tutoring system for improving problem-solving skills. *Computer & Education An International Journal, 94,* 18-36.

Edgar, C. (2013). Supporting novice programmers with natural language in the early stage of programming. *IEEE Symposium on Visual Language and Human-Centric Computing,* September 15-19, San Jose, California, U.S.A.

Hair, J. F. Jr., Babin, B., Money. A. H., & Samouel, P. (2003). *Essential of business research methods.* John Wiley & Sons: United Stated of America.

Janet, C., & Tony, J. (2002). Gender differences in programming?. The *7th Annual. Conference on Innovation and Technology in Computer Science Education.* June 24-26, University of Aarhus, Denmark.

Joshua, V.K., Benjamin, A., K. Alison Gomez, Tyrel, H., Sarah B. Mckagen, Eleanor C. Sayre, Edward W. Schenk, Chase, S., & Lane, S., (2016). Secondary analysis of teaching methods in introductory physics: A 50k-student study. *American Journal of Physics, 84*(12), 969-974.

Kanis, C., & Somkiat, W. (2006). Visual programming using flowchart. *International Symposium on Communications and Information Technologies,* October 18-20, Bangkok, Thailand.

Karen, A. (2008). Making CSO fun: an active learning approach using toys, games and Alice. *Journal of Computing Sciences in Colleges, 23*(3), 98-105.

Min, H. (2003). A Case study in Teaching Adult Students Computer Programming. The *16th Annual NACCQ,* July, Palmerston North, New Zealand.

Malhotra, N. K., Hall, L., Shaw, M., & Oppenheim, P. (2006). *Marketing research: An applied orientation* (3rd ed.). Prentice Hall: New South Wales.

Richard, H. (1998). Interactive-engagement versus traditional methods: A six-thousand-student survey of mechanics test data for introductory physics courses. *American journal of physics, 66,* 64-74.

Sunitra M. Dol. (2015), Fe.q.: An animated flowchart with example to teach the algorithm based courses in engineering. *IEEE 7th International Conference on Technology for Education,* 10-12 December, Warangal, India.