JPEE 실천공학교육논문지
JOURNAL OF PRACTICAL ENGINEERING EDUCATION

실천공학교육
J. Pract. Eng. Educ. 16(3), 259-265, 2024

입출력 정보 조건에 따른 코드 설계 평가 방법 분석

Analysis of Code Design Evaluation Methods According to Input/Output Information Conditions

허 경*

경인교육대학교 컴퓨터교육과

Kyeong Hur*

Department of Computer Education, Gyeong-In National University of Education, Anyang 13910, Korea

[요 약]

대학 학부생들의 SW융합 역량을 향상하기 위해서는, 관련 강좌의 개발과 함께 학부생들의 코드 설계 역량을 평가하는 방법이 연구되어야 한다. 기존 연구에서는 코드 결과물에 대해 정성적인 평가방법과 정량적인 상대평가방법이 있었으며, 정량적인 상대평가방법에서는 문제분해깊이 수, 함수재사용 회수와 함수 개수를 측정하여 평가하였다. 본 연구에서는 기존 연구에서 제시되지 않은 평가방법으로서, 코드 설계 시 입력과 출력 정보종류의 수를 제시하는 문제를 이용한 평가방법을 제안하였다. 본 논문에서 제안한 평가 문제들은 입력 정보 종류의 수와 출력 정보 종류의 수를 3개까지 적용하였다. 이를 통해 5가지 코드설계 평가문제를 제시하고 코드설계 점수를 정량적으로 산출하는 방법을 제안하였다. 제안한 평가방법을 적용한 강좌를 통해 100명 학생 응답자들의 코드들을 수집하고 분석하였다. 결과 분석을 통해, 문제분해깊이 수는 입력 정보의 종류 개수에 비례하고, 함수재사용 회수는 출력 정보의 종류 개수에 비례하며, 함수 개수는 입력과 출력 정보의 종류 총개수에 비례하는 상관성을 나타내었다. 마지막으로 100명 응답자의 평가 점수 분포를 분석하여, 5가지 입출력 정보 조건 평가문제에 따른 코드설계 평가 방법이 유효함을 설명하였다.

[Abstract]

In order to improve the SW convergence capabilities of university undergraduate students, methods to evaluate undergraduate students' code design capabilities should be researched along with the development of related courses. In previous studies, there were qualitative evaluation methods and quantitative relative evaluation methods for code results. In the quantitative relative evaluation method, the number of problem decomposition depth, number of function reuses, and number of functions were measured and evaluated. In this study, an evaluation method that was not presented in previous studies was proposed using the problem of presenting the number of input and output information types when designing code. The evaluation problems proposed in this paper applied up to three types of input information and three types of output information. Through this, five code design evaluation questions were presented and a method to quantitatively calculate code design scores was proposed. Codes from 100 student respondents were collected and analyzed through courses that applied the proposed evaluation method. Through result analysis, the number of problem decomposition depths was proportional to the number of types of input information, the number of function reuses was proportional to the number of types of output information, and the number of functions showed a correlation that was proportional

http://dx.doi.org/10.14702/JPEE.2024.259

Received 31 March 2024; **Revised** 25 April 2024
Accepted 30 April 2024

***Corresponding Author**
E-mail: khur@ginue.ac.kr

to the total number of types of input and output information. Lastly, by analyzing the distribution of evaluation scores of 100 respondents, we demonstrated that the code design evaluation method according to the five input/output information condition evaluation problems is effective.

Key Words: Code design, Evaluation of code design, Non-major undergraduates, Quantitative evaluation, SW education

I. 서 론

SW융합 문제해결 역량에는 컴퓨팅 사고가 필수적이며, 또한, 컴퓨팅 사고를 평가하는 코드 설계 평가 방법도 중요하다. 컴퓨팅 사고(Computational Thinking: CT)는 추상화, 문제 분해, 패턴 인식 그리고 알고리즘 구성, 즉, 코드 설계 역량으로 구성된다[1-3]. 기존 연구에서는 코드 결과물에 대해 정성적인 평가방법과 정량적인 상대평가방법이 있었다. 정성적인 평가방법으로 참고문헌[4]에서는 프로그래밍과 연관된 특정 카테고리의 코드 사용 개수를 측정한 결과와 코드가 내포한 프로그래밍 이해도를 구조화한 평가 매트릭스를 제안하였다. 그리고 참고문헌[5]에서는 컴퓨팅 사고 이론과 실습 성취도 간의 연관성을 컴퓨팅 사고력 4개 요소와 정성평가 8가지 항목으로 분석하였다. 그러나, 정성적인 평가는 컴퓨팅 사고 수준을 분석하는 데 많은 시간이 소요되는 단점이 있다.

한편, 컴퓨팅 사고 평가과정의 복잡도를 낮추기 위해, 참고문헌[6]은 학생들이 제출한 순서도와 코드에 대해 그룹 내 비교 평가와 내부 평가 영역을 제안하였다. 그리고 두 평가 영역에 대해 정량적인 상대 평가 방법을 제안하였다. 코드 설계 평가에 있어, 본 정량적인 상대평가방법에서는 문제분해깊이 수, 함수재사용 회수와 함수 개수를 측정하여 평가하였다. 본 연구에서는 기존 연구에서 제시되지 않은 평가방법으로서, 코드 설계 시 입력과 출력 정보종류의 수를 제시하는 문제를 이용한 평가방법을 제안하였다. 본 논문에서 제안한 평가 문제들은 입력 정보 종류의 수와 출력 정보 종류의 수를 3개까지 적용하였다. 이를 통해 5가지 코드설계 평가문제를 제시하고 코드설계 점수를 정량적으로 산출하는 방법을 제안하였다.

2장에서는 코드설계에 대한 정량적인 상대평가방법을 설명하였다. 그리고 3장에서는 입력과 출력 정보종류의 수를 제시하는 5가지 문제를 이용한 정량적 상대평가방법을 설명하였다. 4장에서는 제안한 평가방법을 적용한 강좌를 통해 100명 학생 응답자들의 파이썬 코드들을 수집하고 분석하였다. 입력과 출력 정보 종류 개수에 따라 100명 응답자들의 문제분해깊이 수, 함수재사용 회수와 함수 개수 분포를 분석하였다[6-9]. 그리고 평가 문제의 입력과 출력 정보 종류 개수가 미치는 영향을 설명하였다. 마지막으로 100명 응답자의 평가 점수 분포를 분석하여, 5가지 입출력 정보 조건 평가문제에 따른 코드설계 평가 방법이 유효함을 설명하였다. 5장에서는 결론을 통해, 제안한 평가 방법의 활용 방안과 추후 연구 과제에 대해 기술하였다.

II. 코드 설계에 대한 정량적인 상대 평가 방법

그림 1은 컴퓨팅 사고를 시각화하여 모델링한 것으로, 문제분해 깊이 1에서 함수 F1-1부터 F1-n까지 n개의 입출력 함수들이 정의되어 있다[6]. F1-1 함수에서 기본 코드(Basic Code: BC), 즉, BC 블록은 최하위 명령어 실행 코드 그룹 영역을 나타낸다. F1-2 함수에서 문제분해 깊이 2에 해당하는 하위 함수 F2-2과 F2-3가 등장하고 있다. 한편, 함수 F2-2는 함수 F1-1과 F1-2에서 재사용되고 있다. 이와 같이, 문제분해 깊이 수, 함수재사용 회수와 함수 개수를 측정하여 코드 설계에 대해 상대 평가할 수 있고, 상대 평가식은 표 1과 같이 정량적으로 정의하였다[6].

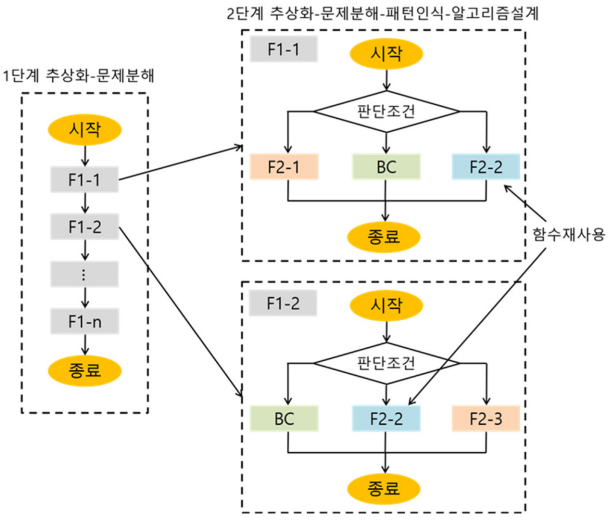

그림 1. 컴퓨팅 사고 시각화모델[6]

Fig. 1. Computational thinking visualization model [6].

III. 입출력 정보 조건에 따른 코드 설계 상대 평가 방법

코드 설계 평가에 있어, 정량적인 상대평가방법은 표 1과 같이 제시한 평가식을 적용하였다[6]. 따라서, 문제분해깊이수, 함수재사용 회수와 함수 개수를 측정하여 평가하였다. 본 연구에서는 기존 연구 참고문헌[6]에서 제시되지 않은 평가방법으로서, 코드 설계 시 입력과 출력 정보종류의 수를 제시하는 문제를 이용한 평가방법을 제안하였다. 본 논문에서 제안한 평가 문제들은 입력 정보 종류의 수와 출력 정보 종류의 수를 3개까지 적용하였다. 이를 통해 5가지 코드설계 평가문제를 제시하고 문제별 가중치를 설계 및 반영하여, 코드설계 역량에 대한 전체 점수를 정량적으로 산출하는 방법을 표 2와 같이 제안하였다.

표 1. 코드 설계 정량적 상대 평가식[6]

Table 1. Code design quantitative relative evaluation formula [6]

평가영역	상대 평가식 (가중치: Weight)
코드설계	• 코드의 문제분해깊이수 점수(P_NCD: Point of Number of Code Depth) P_NCD = Weight_NCD * (NCD/MAX_NCD) (MAX_NCD: 그룹내 최대 NCD, NCD: 해당 코드의 NCD)
	• 코드의 함수재사용회수 점수(P_NCR: Point of Number of Code Reuse) P_NCR = Weight_NCR * (NCR/MAX_NCR) (MAX_NCR: 그룹내 최대 NCR, NCR: 해당 코드의 NCR)
	• 코드의 함수개수 점수(P_NCF: Point of Number of Code Function) P_NCF = Weight_NCF * (NCF/MAX_NCF) (MAX_NCF: 그룹내 최대 NCF, NCF: 해당 코드의 NCF)

표 2에서 제시한 평가문제는 1입력 1출력 정보종류 코드설계, 1입력 2출력 정보종류 코드설계, 2입력 2출력 정보종류 코드설계, 2입력 3출력 정보종류 코드설계와 3입력 3출력 정보종류 코드설계로 5가지이다. 4장에서는 100명 학생 응답자들의 파이썬 코드들의 문제분해깊이 수, 함수재사용 회수와 함수 개수를 측정하여 상대 평가한 결과를 제시한다. 표 2에서 그룹내 최대값은 100명의 응답자 중 해당 값을 나타낸다. 그리고 응답자의 결과값은 100명 중 해당 응답자의 값을 나타낸다. 5가지 평가문제에 대하여 문제의 난이도를 고려하여 가중치를 10%에서 30%까지 할당하였다. 4장에서 분석한 100명 응답자의 코드는 모두 성공적으로 동작한 코

표 2. 입력과 출력 정보종류 수에 따른 코드설계 점수 정량적 평가방법

Table 2. Quantitative evaluation method for code design scores according to the number of input and output information types

평가 문제	측정항목	그룹내 최대값	가중치	응답자의 결과값	응답자 점수
1입력 1출력 정보종류 코드설계 (10%)	문제분해깊이수(NCD)	3	2.5	2	1.67
	함수재사용회수(NCR)	2	2.5	1	1.25
	함수개수(NCF)	10	2.5	5	1.25
	코드동작오류	PASS	2.5	PASS	2.5
1입력 2출력 정보종류 코드설계 (15%)	문제분해깊이수(NCD)	3	3.75	2	2.50
	함수재사용회수(NCR)	4	3.75	2	1.88
	함수개수(NCF)	19	3.75	10	1.97
	코드동작오류	PASS	3.75	PASS	3.75
2입력 2출력 정보종류 코드설계 (20%)	문제분해깊이수(NCD)	6	5	4	3.33
	함수재사용회수(NCR)	4	5	2	2.50
	함수개수(NCF)	28	5	19	3.39
	코드동작오류	PASS	5	PASS	5.00
2입력 3출력 정보종류 코드설계 (25%)	문제분해깊이수(NCD)	6	6.25	4	4.17
	함수재사용회수(NCR)	5	6.25	3	3.75
	함수개수(NCF)	42	6.25	30	4.46
	코드동작오류	PASS	6.25	PASS	6.25
3입력 3출력 정보종류 코드설계 (30%)	문제분해깊이수(NCD)	8	7.5	6	5.63
	함수재사용회수(NCR)	6	7.5	3	3.75
	함수개수(NCF)	59	7.5	44	5.59
	코드동작오류	PASS	7.5	PASS	7.50
합계(100%)			100		72.09

드로서 표 2에 있는 코드동작오류 부문은 모두 PASS를 받고 해당 가중치의 점수가 부여되었다. 표 2의 응답자 점수는 표 1에서 설명한 코드설계 정량적 상대평가식에 따라 산출된 것이다.

각 평가문제에 할당하는 만점, 즉, 100점 총점에 기준하여 가중치를 정하는 방법은 다양하다. 본 연구에서는 각 평가문제의 입력 정보 수와 출력 정보의 수를 합한 총 수, 즉, 20(=2+3+4+5+6)으로 100을 나눈 값(5)을 각 평가문제가 갖는 입력 정보와 출력 정보의 수의 합과 곱하여 각 평가문제의 가중치를 결정하였다. 따라서, 표 2와 같이, 각 평가문제는 10(=2*5), 15(=3*5), 20(=4*5), 25(=5*5), 30(=6*5)의 가중치(최대 점수)를 갖는다. 한편, 각 평가문제는 4가지 상세 평가 항목(문제분해깊이 수, 함수재사용 회수, 함수 개수, 코드 동작오류)으로 나누어 점수가 결정되었다. 본 연구에서는 4가지 상세 평가 항목이 갖는 중요도, 즉, 가중치(최대 점수)를 각 평가문제에 배정된 최대 점수를 4로 나눈 값(10/4. 15/4, 20/4. 25/4, 30/4)으로 동일하게 설정하였다. 이 부분에 대한 가중치도 다양하게 설정할 수 있다.

학생 응답자는 5가지 평가문제를 통해 코드 5종을 제출하고, 상대평가에 의한 평가점수를 받는다. 코드 설계 역량에 대해 표 2와 같이 해당 응답자는 100명 중에서 72.09 점수를 받는다. 예를 들어, '3입력 3출력 정보종류 코드설계' 문제는 '3가지 서로 다른 특성을 갖는 정보를 입력 받아서, 3가지 서로 다른 특성을 갖는 정보를 출력해야 하는 코드를 작성하는 문제'를 의미한다. 이 문제에서 발생하는 서로 다른 실행 조건 함수들은 다수 발생하게 되고, 각 실행 조건을 만족할 때, 실행하게 되는 출력 함수들도 다양한 경우의 수로 나타나게 된다. 이에 따라, 응답자의 코드에는 다수의 함수들이 포함되었다.

IV. 결과 분석

4장에서는 표 2와 같이 제안한 '입력과 출력 정보종류 수에 따른 코드설계 점수 정량적 평가방법'을 적용한 강좌를 통해 수집된 100명 학생 응답자들의 파이썬 코드들을 분석하였다. 입력과 출력 정보 종류 개수에 따라 100명 응답자들의 문제분해깊이 수, 함수재사용 회수와 함수 개수 분포를 분석하였다[6-9]. 그리고 평가 문제의 입력과 출력 정보 종류 개수가 이 세가지 분포 결과에 미치는 영향을 제시하였고, 100명 응답자의 코드설계 전체 점수 분포는 제안한 입출력 정보 조건 문제에 따른 코드설계 평가 방법이 유효함을 나타내었다.

A. 문제분해깊이 수(NCD) 분석

그림 2는 입력과 출력 정보 종류 개수에 따라 100명 응답자 코드들의 문제분해깊이 수의 분포, 즉, 최소값, 1사분위수, 평균, 중앙값, 3사분위수, 최대값과 극단치를 상자 수염 그래프로 나타낸 것이다. 그림 2에서 1-1NCD는 입력정보 1개 종류. 출력정보 1개 종류인 코드의 문제분해깊이 수를 나타낸다. 본 연구에서 수집한 그림 2 데이터 결과에서 문제분해깊이 수는 입력 정보 개수에 비례하는 상관관계를 갖고 있다. 그림 3에 나타낸 문제분해깊이 수의 100명 응답자 분포는 입력과 출력 정보 종류 개수에 따라 달라진다. 그림 4에서도 문제분해깊이 수의 대표값, 즉, 최대값, 중간값과 최소값이 입력 정보 개수에 비례하는 상관관계를 갖고 있다.

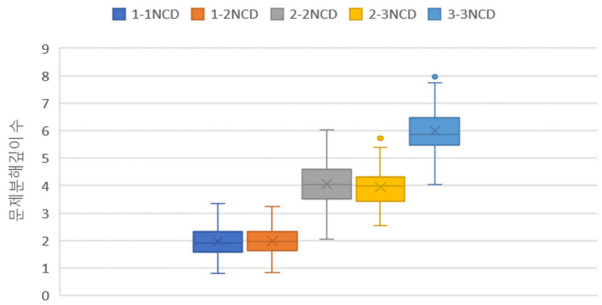

그림 2. 입력출력정보종류 개수별 문제분해깊이 수의 분포

Fig. 2. Distribution of problem resolution depth by number of input and output information types.

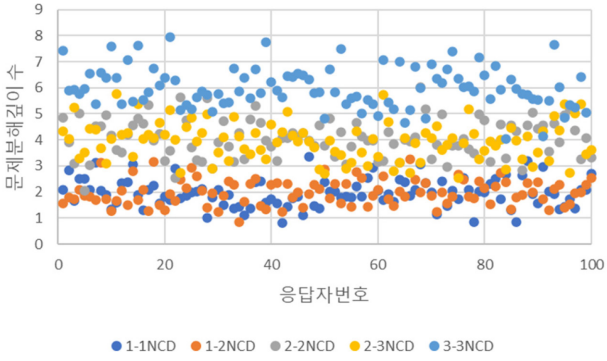

그림 3. 입력출력정보종류 개수별 문제분해깊이 수의 응답자 분포

Fig. 3. Distribution of respondents in the number of problem resolution depths by number of input and output information types.

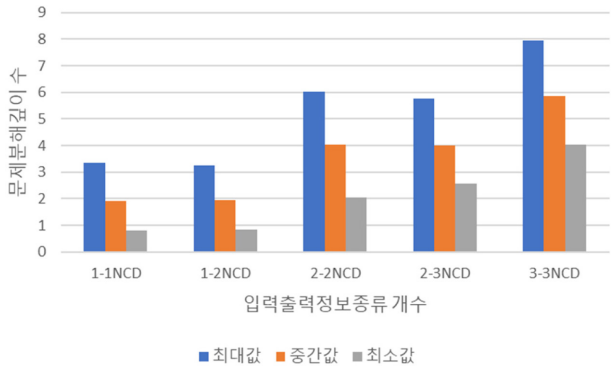

그림 4. 입력출력정보종류 개수별 문제분해깊이 수의 대표값 비교

Fig. 4. Comparison of representative values of the number of problem resolution depths by number of input and output information types.

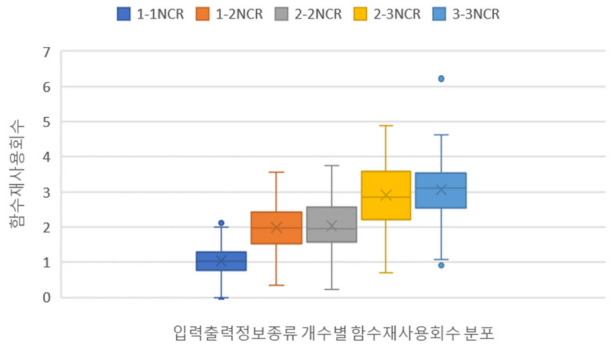

그림 5. 입력출력정보종류 개수별 함수재사용회수의 분포

Fig. 5. Distribution of number of function reuses by number of input and output information types.

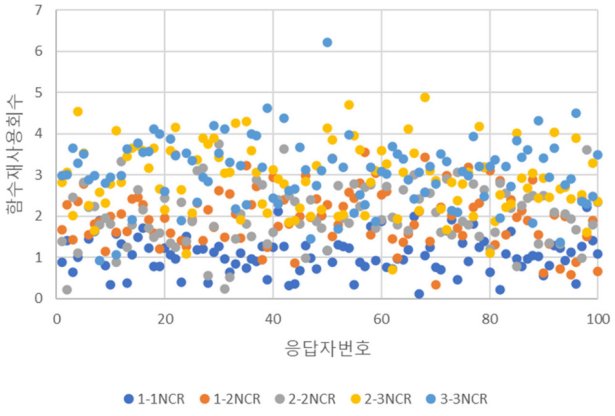

그림 6. 입력출력정보종류 개수별 함수재사용회수의 응답자 분포

Fig. 6. Distribution of respondents in number of function reuses by number of input and output information types.

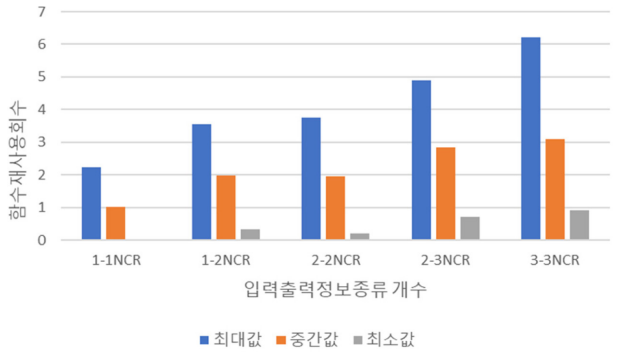

그림 7. 입력출력정보종류 개수별 함수재사용회수의 대표값 비교

Fig. 7. Comparison of representative values of number of function reuses by number of input and output information types.

B. 함수재사용회수(NCR) 분석

그림 5는 입력과 출력 정보 종류 개수에 따라 100명 응답자 코드들의 함수재사용회수의 분포를 상자 수염 그래프로 나타낸 것이다. 그림 5에서 1-1NCR은 입력정보 1개 종류. 출력정보 1개 종류인 코드의 함수재사용회수를 나타낸다. 그림 5 데이터 결과에서 함수재사용회수는 출력 정보 개수에 비례하는 상관관계를 갖고 있다. 그림 6에 나타낸 함수재사용회수의 100명 응답자 분포는 입력과 출력 정보 종류 개수에 따라 달라진다. 그림 7에서도 함수재사용회수의 대표값이 출력정보 개수에 비례하는 상관관계를 갖는다.

C. 함수개수(NCF) 분석

그림 8은 입력과 출력 정보 종류 개수에 따라 100명 응답자 코드들의 총 함수개수의 분포를 상자 수염 그래프로 나타

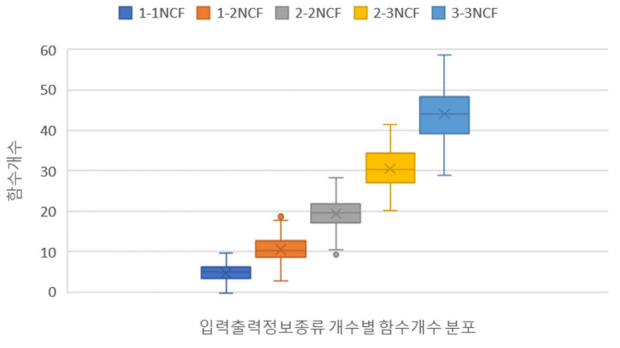

그림 8. 입력출력정보종류 개수별 함수개수의 분포

Fig. 8. Distribution of number of functions by number of input/output information types.

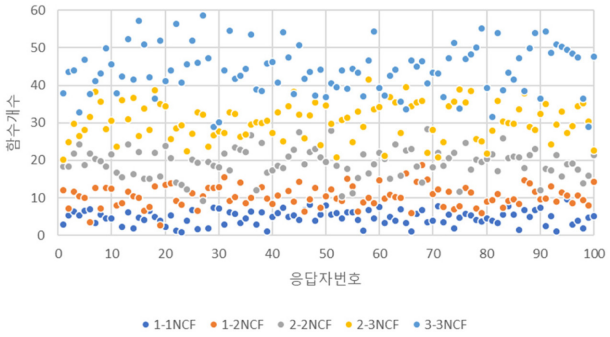

그림 9. 입력출력정보종류 개수별 함수개수의 응답자 분포

Fig. 9. Distribution of respondents by number of functions by number of input and output information types.

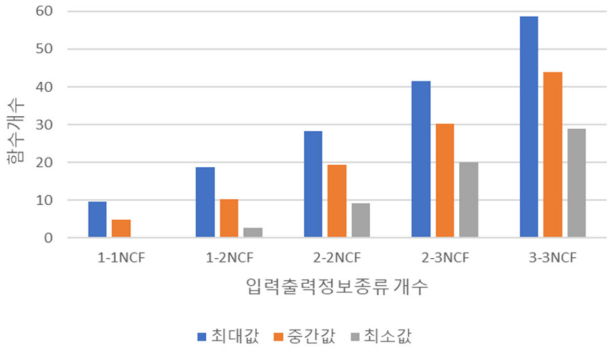

그림 10. 입력출력정보종류 개수별 함수개수의 대표값 비교

Fig. 10. Comparison of representative values of the number of functions by number of input and output information types.

낸 것이다. 그림 8에서 1-1NCF는 입력정보 1개 종류. 출력정보 1개 종류인 코드의 총 함수개수를 나타낸다. 그림 8 결과로부터 함수개수는 입력 정보 개수와 출력 정보 개수에 모두 비례하는 상관관계를 갖는다. 그림 9에 나타낸 총 함수개수의 100명 응답자 분포는 입력과 출력 정보 종류 개수에 따라 크게 달라진다. 그림 10에서도 함수개수의 대표값이 입력 정보 개수와 출력 정보 개수에 모두 비례하는 상관관계를 갖는다.

D. 응답자 평가 점수 분석

그림 11은 100명 응답자의 코드설계 전체 점수 분포를 나타낸다. 그리고 표 3에 그림 11의 전체 점수 분포의 대표값들을 정리하였고, 100점 만점에서 최소값이 65.6이고 최대값이 80.9를 나타내어 적절한 난이도와 변별력을 갖는 평가 방법이었다. 그리고, 적절한 평균값을 평가점수로 도출할 수 있다. 이와 같은 결과로부터, 제안한 입출력 정보 조건 문제에 따른 코드설계 평가 방법은 유효하다.

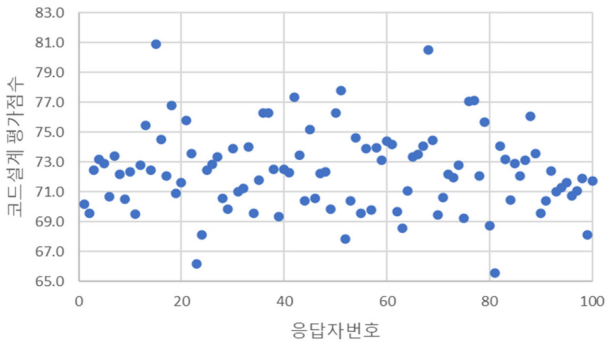

그림 11. 응답자별 전체 점수의 분포

Fig. 11. Distribution of total scores by respondent.

표 3. 응답자 전체 점수 분포 대표값

Table 3. Representative value of total score distribution of respondents

전체 점수 분포 대표값				
최대값	중간값	평균	최소값	표준편차
80.9	72.3	72.4	65.6	2.7

V. 결 론

본 논문에서는 입력 정보 종류의 수와 출력 정보 종류의 수를 지정하여 조건으로 제시하는 코드 설계 평가 방법을 제안하였다. 본 평가방법은 순서도 없이 코드 결과물만 수집하는 SW교육 강좌에서 활용도가 높다. 또한, 입력 정보 종류의 수와 출력 정보 종류의 수를 보다 더 큰 값으로 다양하게 적용하여 코드 설계 역량을 종합적으로 평가할 수 있는 확장성을 갖는다. 또한 분석결과로부터, 평가 문제에서 지정한 입력과 출력 정보 종류 개수가 응답자 답안의 코드 구조에 영향을 준다는 것을 확인하였다. 추후 연구에서는 더 큰 규모의 응답자 답안을 수집한 후 머신러닝 모델을 이용하여 평가 문제에서 지정한 입력과 출력 정보 종류 개수가 응답자 답안의 코드 구조에 미치는 영향을 분석한다. 이를 통해, 유효한 예측 및 분류 모델을 만들어 학생들의 코드 설계 역량 교육에 적용한다.

참고문헌

[1] S. O. Yang, “Necessity of computational thinking,” *Korea Information Processing Society Review*, vol. 24, no. 2, pp. 4-12, March 2017.

[2] J. M. Wing, “Computational thinking,” *Communications of the ACM*, vol. 49, no. 3, pp. 33-35, March 2006.

[3] J. M. Wing, “Computational thinking and thinking about computing,” *Philosophical Transactions of the Royal So Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1881, pp. 3717-3725, October 2008.

[4] W. S. Sohn, “A method for measuring and evaluating for block-based programming code,” *Journal of the Korean Association of Information Education*, vol. 20, no. 3, pp. 293-302, June 2016.

[5] J. K. Kim, “Development of rubric for assessing computational thinking concepts and programming ability,” *Journal of Korean Association of Computer Education*, vol. 20, no. 6, pp. 27-36, November 2017.

[6] K. Hur, “A study on quantization of a code evaluation method through visualization of computational thinking,” *Journal of Practical Engineering Education*, vol. 13, no. 1, pp. 199-206, April 2021.

[7] K. Hur, “Educational method of computational thinking processes using physical teaching devices,” *Journal of Practical Engineering Education*, vol. 10, no. 1, pp. 35-39, June 2018.

[8] K. Hur, “An education method of computational thinking using microbit in a java-based SW lecture for non-major undergraduates,” *Journal of Practical Engineering Education*, vol. 11, no. 2, pp. 167-174, December 2019.

[9] K. Hur, “An education method of Java SW designs for IoT wireless device control using microbits,” *Journal of Practical Engineering Education*, vol. 12, no. 1, pp. 85-91, June 2020.

허 경 (Kyeong Hur)_종신회원

1998년 : 고려대 전자공학과 학사
2000년 : 고려대 전자공학과 석사
2004년 8월 : 고려대 전자공학과 통신공학박사
2004년 8월 ~ 2005년 8월 : 삼성종합기술원(SAIT) 전문연구원
2005년 9월 ~ 현재 : 경인교대 컴퓨터교육과 교수
〈관심분야〉 네트워크 MAC QoS, IoT, SW교육, AI교육, 데이터과학교육