우리말 식 프로그래밍 교육용 언어 PELIHAN의 설계 및 구현

*김 명 렬, **박 영 진
*전북대학교, **(주)한국정보시스템

Design and Implementation of PELIHAN : Programming Educational Language In HANgul

*Myoung Lyoul Kim, **Young Jin Park

*Chonbuk National University, **Korea Information System Corporation

요 약

후치 표기 내지 한국어와 유사한 구문과 LISP의 함수와 같은 모듈로 이루어지는 초중고교생의 프로그래밍 교육용 언어 PELIHAN을 설계하고, PELIHAN 전용의 구문지향적 지능형 편집기와 인터프리터를 내장한 통합 환경을 구비한 언어 시스템의 프로토타입을 구현하였다.

I. 서 론

한글 및 한국어 정보 처리에 관한 지금까지의 연구는 주로 이들 정보 자체의 연구에 집중되고 있다. 다시 말하면 한글 문자 및 한국어의 음성 인식과 한국어의 기계번역에 치중되고 있다.

한편 컴퓨터 교육에 관한 연구는 컴퓨터 자체에 관한 것보다 컴퓨터를 이용하는 CAI 코스웨어 제작이나 저작용 언어와 도구의 개발에만 치중하고 있다.

본 연구는 한국어의 이해와 컴퓨터에 관한 교육의 공통점을 구비한 우리말 식 프로그래밍 교육용 언어의 개발을 위한 것이다.

종래의 유사한 연구는 주로 기존의 프로그래밍 언어에서 키워드만 한글로 대체하는 것이 전부였으므로[1] 사용자들에게 아무런 관심도 불러 일으킬 수가 없었을 뿐만 아니라 이러한 연구는 우리말 식 언어 처리기 개발 등에 별다른 기술 축적도 하지 못하였다. 최근의 연구는 BASIC의 키워드및 변수명까지 한글로 개발된 것이 상품화되어 있다[2]. 본 연구는 개인용 컴퓨터를 사용하는 초중고교생의 컴퓨터 프로그래밍 교육을 위한 우리말 식으로 된 프로그래밍 언어 시스템의 프로토타입을 설계·구현한 것이다[3][4].

이는 국적있는 교육, 창조성을 계발하는 교육, 시대적으로 중요한 컴퓨터 교육 중 본질적인 프로그래밍 교육이라는 교육적 측면과 시스템 소프트웨어 개발의 기술 축적이라는 측면에서도 시기적절하고 가치있는 연구라고 사료된다.

Ⅱ. 기존 교육용 프로그래밍 언어의 비판과 새로운 언어의 필요성

1. BASIC에 대한 편견

(1) BASIC은 1970년대초 마이크로컴퓨터의 시작품으로 불가피하게 채용된 것일뿐 좋은 언어이기 때문에 채용된 것이 아니다. 또, BASIC이 확산된 것은 마이크로컴퓨터의 보급 덕분이지 언어 자체가 좋기 때문이 아니다. 이는 C 언어가 UNIX와 더불어 인기있는 언어로 된 것과 다르다. BASIC이 무비판적으로 확산된 것은 우리에게 적합한 교육용 언어가 없기 때문이었다.

(2) BASIC이 계속 확산된 이유는 디스크 드라이브나 그래픽 기능을 구비한 단말기 등의 개발 기술이 부족하며 고가이기 때문에 LOGO 같은 좋은 언어들이 빛을 보지 못하였으며 컴퓨터 산업의 영리추구에 밀려 컴퓨터에 관한 교육은 등한시 되어 왔기 때문이다[5][6].

(3) BASIC은 좋은 언어가 아니다. 사람들은 쉬운 것을 좋은 것으로 잘못 생각하여 빠져들어 버린다. 이는 전산학에 대한 이해가 부족하기 때문이다. 그리고 간단한 문제는 어떤 언어로도 프로그래밍하기 쉬운 것을 모르고 BASIC으로 모든 것이 잘 해결되는 줄로 착각하고 있다.

(4) BASIC은 교육용 언어가 아니다. BASIC은 그 당시 Darthmouth 대학교 미술과 학생들의 프로그래밍 교육과 시분할 시스템의 구축을 위하여 개발한 것이며, 오늘날의 입장에서 볼 때 명실상부한 교육용 언어라고 할 수 없다[7]. 또, Pascal은 전산학 전공인 학생들의 프로그래밍 교육을 위하여 만든 언어이다.

2. BASIC의 결점

(1) BASIC은 모듈 개념이 없다. 이는 기본적인 전산 개념인 분할정복(divide and conquer)이라는 문제 해결도 모르게 한다. 단일 모듈로 되어있어서 모든 변수가 전역변수이고 문 번호들이 독립적이 아니기 때문에 GOSUB는 GOTO를 이용한 것 이다. 그러므로 BASIC에는 부프로그램이 존재하지 않는다. 따라서, 모듈화 프로그래밍이나 하향식 설계에 아무런 도움을 주지 못한다. 그래서 문제 해결을 위한 접근 방식, 프로그램의 작성 및 표현 방식이 조잡하고 잔 기교를 쓰는 바 건전하지 못한 프로그래밍 방식을 고착화 시킨다. 이는 프로그래밍 언어가 간단하지만 강력하지 못한 증거이다. 즉, 사고와 표현의 도구로써 부적절하여 창조성 계발을 저해하는 측면이 있으므로 교육용으로 적합하지 않다는 것이 인지심리학자들의 공통된 주장이다[8][9].

(2) BASIC은 FORTRAN을 단순화 시킨 것이므로 사무자료 처리나 지식자료 처리에 필요한 기능은 전혀 없으며 재귀(recursion)의 기능이 없으므로 재귀적 사고나 표현이 불가능하다. 또 구조적 제어구문이 빈약하여 심하게 반복과 goto문에 의존하므로 프로그램을 읽기 어렵게 만든다.

(3) 오늘날 실무용 프로그램을 BASIC으로 작성하는 일은 드물다. 그렇다면 BASIC은 교육용으로만 쓸 수 밖에 없다. 교육은 현실적 응용 이외에 보다 차원높은 지식과 기술을 습득하기 위한 전이에 있다고 볼 때 BASIC은 결코 이러한 기능을 구비하고 있는

것이 아니다. BASIC은 유연성, 강력성, 확장성은 전혀 없으며 기존 프로그래밍 언어와는 달리 미래의 프로그래밍 언어개념에 관련 시킬만한 중요 개념이 없다[10][11].

X = X + 1 나 IF X = 1 THEN...에서 처럼 = 는 일관성이 없을 뿐만 아니라 등호를 치환으로 사용하여 초보자에게 혼란을 야기 시키며, 고정양식의 프로그램 작성은 자유성을 억압한다.

(4) BASIC이나 Pascal로서 그래픽을 하려면 이들 언어를 거의 마스터 한 후에야 가능하다. 더우기 좌표에 의한 그래픽을 하므로 국민학생 및 중학생들에게는 난해하다. 이런 점은 교육적인 흥미나 동기 유발의 요소가 되지 못한다.

3. 새로운 프로그래밍 언어의 필요성

(1) 현실 사회의 적응을 위하여 각종 OA용 패키지의 사용 방법과 컴퓨터 문맹퇴치를 강조하고 있지만 학교 교육에서 교육적으로 중요한 것은 프로그래밍 교육을 통한 창조성 계발이다. MIT에서 Scheme을, Yale 대학에서 T 언어를 가르치는 이유는 여기에 있다. 컴퓨터 교육 중 실제적이고 본질적인 것은 프로그래밍이고 이는 프로그래밍 언어 없이는 불가능하다.

(2) Sapir - Whorf의 언어 이론은 「언어는 우리의 행동과 사고의 양식을 결정하고 주조한다」고 하였다. 언어에 대한 현대 철학의 입장처럼 사고는 언어의 필요한 조건이지만 언어는 고차원적 사고를 위한 매체인 것이다. 결론적으로 언어는 사고를 표현하는 수단일 뿐만 아니라 무엇보다도 사고를 조작하는 능력을 제공하는 도구인 것이다[12]. 적어도 서구의 많은 국가들이 자국어로된 BASIC을 교육함은 그들의 언어가 영어에 유사하기 때문이다. 이제 우리는 우리의 사고 방식대로 표현할 수 있는 언어가 필요하다. 비효율적이거나 불편하거나 난해하지 않다면 BASIC에서 탈피하여 우리말 식 교육용 언어를 채택하여 컴퓨터 교육에 초석이 되어야 한다. 따라서, 기성 전산인은 고정관념에 사로잡힌 시각과 편견을 버리고 더우기 발전의 저해 요소인 안일에서 탈피하여 우리 환경에 맞는 컴퓨터 문화를 개발해야 할 것이다.

(3) 오늘날에는 컴퓨터에 대한 이해의 수준이 높아졌으며 값싼 디스크 드라이브, 풍부한 메모리, 다방면으로 프로그래밍 환경이 매우 좋아졌다. BASIC이 시스템에 내장되는 시대는 이미 과거가 되어 버렸기 때문에 좋은 언어가 없어서 안타까울 뿐이므로 결코 과거처럼 어떤 언어에 고착되어서는 안 된다. 올바른 컴퓨터 교육의 정착을 위해서는 새로운 프로그래밍 언어의 개발, 보급, 활용은 필수 불가결한 것이다[13][14].

Ⅲ. PELIHAN의 정의

1. PELIHAN의 설계 목표

(1) 우리말 식 언어 ... 구문이 한국어 구문과 유사하여 프로그램 작성이 용이하고 자연스러울 것.
(2) 간명한 언어 ..... 구문이 간명하여 프로그램을 읽어 이해하기 쉬울 것.
(3) 쉬운 언어 ..... 문법과 의미를 배우기가 쉽고 구현하기 용이할 것.
(4) 흥미로운 언어 .... 멀티미디어 시대에 부응하여 turtle 그래픽과 sound 기능을 구비한 시청각적인 언어일 것.
(5) 실용적 언어 ..... 초중고교생의 어떤 학습내용도 프로그래밍 할 수 있도록 풍부한 자료 구조와 다양한 라이브러리를 구비할 것.
(6) 새로운 언어 ..... 프로래밍언어 및 소프트웨어 공학 측면에서 현대적일 것.

2. PELIHAN의 사양

(1) PELIHAN으로 프로그램구조는 다음과 같다.

〈프로그램〉 ::= 〈정의〉+

〈정의〉 ::= 〈정의명〉(〈형식인수나열〉) # 〈라벨부착가능문〉+ #

〈라벨부착가능문〉 ::= [〈라벨〉]〈문〉, 〈라벨〉 ::= 〈〈명칭〉〉

(2) PELIHAN의 문은 공백으로 분리되며 각 문의 구문은 다음과 같다.

〈문〉 ::= 〈치환문〉¦〈선택문〉¦〈조건반복문〉¦〈요소반복문〉¦〈정의호출문〉¦〈입력문〉
¦〈출력문〉¦〈보조제어문〉¦〈선언문〉

〈치환문〉 ::= 〈1값〉 〈- 〈식〉

〈선택문〉 ::= (〈조건〉) 이면 { 〈라벨부착가능문〉+ }
[아니면 { 〈라벨부착가능문〉+ }]

〈조건반복문〉 ::= (〈조건〉) 반복 { 〈라벨부착가능문〉+ }

〈요소반복문〉 ::= (〈변수〉 〈- 〈초기치〉..〈최종치〉,〈증감치〉) 반복{ 〈라벨부착가능문〉+ }

〈정의호출문〉 ::= 〈정의명〉(〈형식인수나열〉) ¦(〈형식인수나열〉)〈정의명〉

〈입력문〉 ::= ([화일:] 〈1값 나열〉 [:입력양식])읽어라

〈출력문〉 ::= ([화일:] 〈식의나열〉 [:출력양식])써라

〈보조제어문〉 ::= (〈라벨〉)가라 ¦ ([〈식〉])반환

〈선언문〉 ::= 〈배열선언문〉 ¦ 〈구조선언문〉

〈배열선언문〉 ::= 〈배열명〉(〈형〉:〈모양〉)

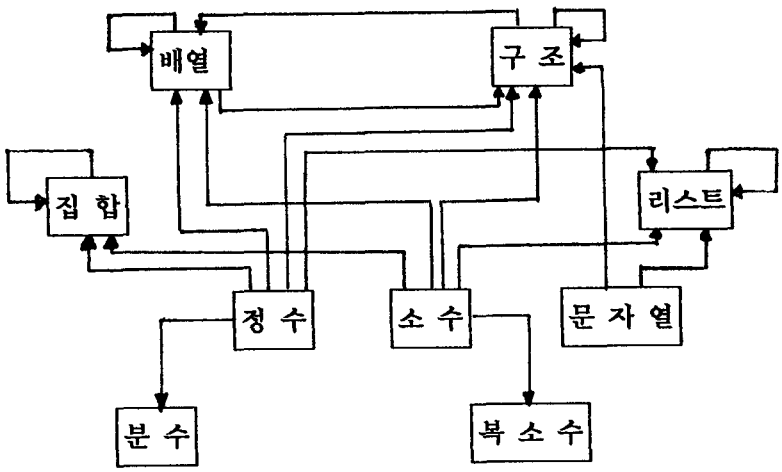
〈구조선언문〉 ::= 〈구조명〉(〈그룹또는 개별항목〉+)

〈그룹또는 개별항목〉 ::= 〈그룹항목〉 ¦ 〈개별항목〉

〈그룹항목〉 ::= 〈계층번호〉〈그룹명〉
〈그룹 또는 개별항목〉+

〈개별항목〉 ::= 〈계층번호〉〈항목명〉:〈형〉

(3) PELIHAN의 자료형은 다음과 같이 구조적으로 도시할 수 있다.



3. PELIHAN의 특징

(1) 우리말 식으로 작성할 수 있으므로 프로그램의 작성과 이해가 용이하다.

(2) 하나의 〈정의〉는 모듈이므로 모듈화 프로그래밍이나 하향식 설계, 절차 지향 프로그래밍을 할 수 있다.

(3) f(x)와 같은 정의 호출이 가능하므로 함수형 프로그래밍을 할 수 있으며, (x)f와 같은 정의 호출이 가능하므로 객체지향 기법을 이해할 수 있다[15].

(4) 재귀적 표현이 가능하며, turtle 그래픽이 용이하므로 교육적 동기 유발이 보다 쉽다[16].

(5) 구문과 시스템의 운용 방법은 일관성을 있게 하였다.

(6) BASIC처럼 쉽지만 Pascal처럼 강력하다.

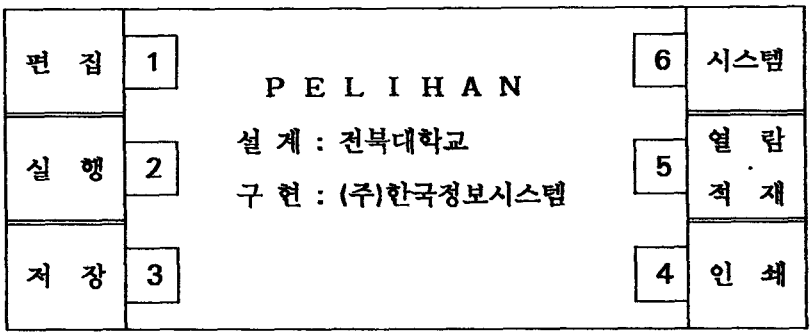
Ⅳ. PELIHAN의 언어시스템

1. 프로그래밍 작성 환경

(1) 통합 환경의 구축

① 객체지향 기법을 구현한 프로그래밍 환경은 윈도우 방식에 따라 프로그래밍을 구축한다. 이는 프로그램 작성과 실행을 간편하고 용이하고 대화식으로 수행하는 점에서 매우 좋다. 오늘날의 상업용 윈도우에 가까운 환경의 구축은 실제로 어려울 뿐만 아니라 초심자에게는 어려우므로 최소한의 최상위 레벨의 설정과 하나의 하위 단계만으로 간단한 메뉴를 구성하여 사용하기 용이하게 하였다.

② PELIHAN의 통합환경은 상태천이도를 반영한 것으로 그 구성은 다음 그림과 같다.



그 림 2. PELIHAN의 통합환경

③ 사용자는 초중고교생이므로, 또 인터프리터의 구현을 용이하게하기 위하여 메모리에는 하나의 화일만 존재하고(작성, 적재) 실행될 수 있도록 한다.

(2) PELIHAN의 편집기

① 사용하기 용이하고 친밀한 대화적 프로그래밍 환경의 구축을 위하여 PELIHAN 전용의 편집기를 통합환경에 포함시킨다[17].

② 이 편집기는 구문지향적(syntax directed)이며 구조적(structual) 또는 지능형(inteligent) 편집기로서 그림 3과 같이 프로그램의 작성과 수정을 용이하게하며 인터프리터의 효율적인 해석실행을 도와주는 혼합형 모드로서 그 특성은 다음과 같다.[18][19].

③ PELIHAN의 구문에 익숙치 못한 사용자에게 구문을 미리 보여주어 구문오류를 방지한다.

④ PASS1의 역할로서 해석목을 생성하여 실행속도를 빠르게 한다.

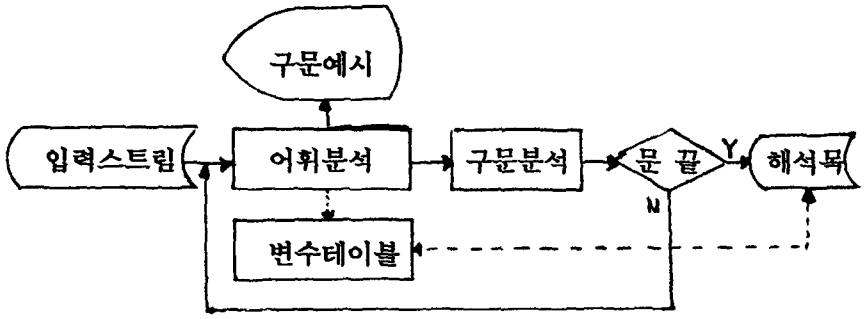


그림 3. PELIHAN의 혼합형 편집기

2. 프로그램 실행환경

(1) PELIHAN의 인터프리터

① 대화식 및 교육용 컴퓨터에는 대개 인터프리터 방식이 좋으므로 PELIHAN의 언어처리계도 인터프리터로 구현한다[22][23].

② 언어 확장과 이식성을 위하여 C 언어로 구현한다[20][21].

③ 인터프리터는 그림 4와 같이 PASS2의 역할로서 해석목을 실행한다.

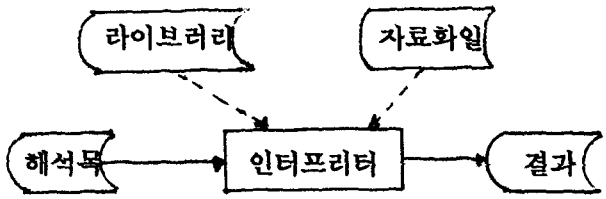


그림 4. PELIHAN의 인터프리터

(2) 실행 방식

① 프로그램은 <정의>의 집합이므로 인수를 전달하는 방식으로 한다.

② PROLOG와 다른 점은 하나의 문으로만 질문하도록 한다. 즉,

<질문> ::= ? <문>↵

③ 위 ②의 경우 프로그램과 무관한 인수 없는 <문>인 경우에는 인터프리터 언어의 즉시형 모드처럼 실행한다.

V. 결 론

참신하고 주체성 있는, 창조성을 계발하는 컴퓨터 교육과 시스템 소프트웨어 개발 기술의 축적이라는 측면에서 우리의 실정에 적절한 미래지향적인 프로그래밍 교육용 언어의 개발은 시급하고 필요하다고 생각되어 이제 모두가 기존 언어에 대한 냉철한 반성과 아울러 새롭고 올바른 전산교육의 확립을 위하여 PELIHAN의 보완과 보급에 앞장서야 할 것이다. 이는 국가적 차원의 지원(교육과정 반영 및 언어 시스템의 완벽한 구현)없이는 어려운 일이라고 생각된다.

앞으로 PELIHAN은 논리개념, 병행성 및 객체지향 기법의 설비를 언어 확장과 더불어 디버거 등을 구비한 완전한 언어 시스템으로 발전되어야 할 것이다.

<참고 문헌>

[1] 공업진흥청 : 정보처리를 위한 중장기 계획에 관한 연구; 공업진흥청, pp. 294-295, 1987.

[2] 안철우 : 한 베이직 인터프리터

[3] 김명렬 : "컴퓨터 교육용 프로그래밍 언어 CELL의 설계에 관한 연구"; 홍익대 박사 학위 논문, 1989.8.

[4] 원유헌, 김명렬, 이태욱 : " 컴퓨터 교육용 프로그램이 언어 CELL의 설계에 관한 연구"; 한국정보과학회 논문지, 제16권 4호, pp. 350-361, 1989.

[5] William J. Bramble, Emanuel J. Mason : Computers in Schools; McGraw - Hill Book Company, 1985.

[6] William E. Schall, Lowell Leake, Jr., Donald R. Whitaker : Computer Education : Literacy and Beyond, 2nd. ed.; Brooks/Cole Publishing Company, 1989.

[7] Eileen Scanlon, Tim O'Shea : Educational Computing; John Wikey & Sons, 1987.

[8] Seymour Papert : Mindstorms : Children, Computers , and Powerful Ideas; Basic Books, 1980.

[9] Masoud Yazdani : New Horizontal in Educational Computering; Ellis Horwood Limited, 1984.

[10] 원유헌 : 프로그래밍 언어론 : 정익사, p. 31, 1991.

[11] Fred A. Masterson : "Language for Students", Byte, Vol. 6, pp. 223-238, 1984.

[12] Howard Levine, Howard Rheingold : The Cognitive Connection; Prentice Hall Press, p. 1, 1987.

[13] 이태욱 : "컴퓨터 교육학의 정립"; 한국교원대학교, 교원교육, 제2권 1호, 1986.

[14] 한국교원대학교 : 학교 컴퓨터 교육; 한국교원대학교, p. 53, 1992.

[15] David Hu : Object-Orinted Environment in C++; Baldur Systems Corp., pp. 27-29, 1990.

[16] 김명렬 : LOGO 프로그래밍; 산학사, 1984.

[17] L. Allison : "Syntax Directed Program Editing"; Software-Practice and Experience, Vol. 13, pp. 453-465, 1983.

[18] P. Degano, S. Mannucci, B. Mojana : "Efficient Incremental Parsing for Syntax-Directed Editions"; ACM TOPLAS, Vol. 10, NO. 3, pp. 345-373, 1988.

[19] T. Teitelbaum, T. Reps : "The Cornell Program Synthesizer: A Syntax-Directed Programming Environment", Com. ACM, Vol. 24, No. 9, pp. 563-573, 1981.

[20] 김명렬 : C 언어 프로그래밍; 산학사, 1986.

[21] Allen I. Holub : Compiler Design in C; Prentice-Hall International, Inc., 1990.

[22] Brown P. J. : Writing Interactive Compilers and Interpreters; John Willey & Sons, 1979.

[23] Samuel N. Kamin : Programming Language : An Interpreter-Based Approach; Addison Wesley Publishing Company, Inc., 1990.