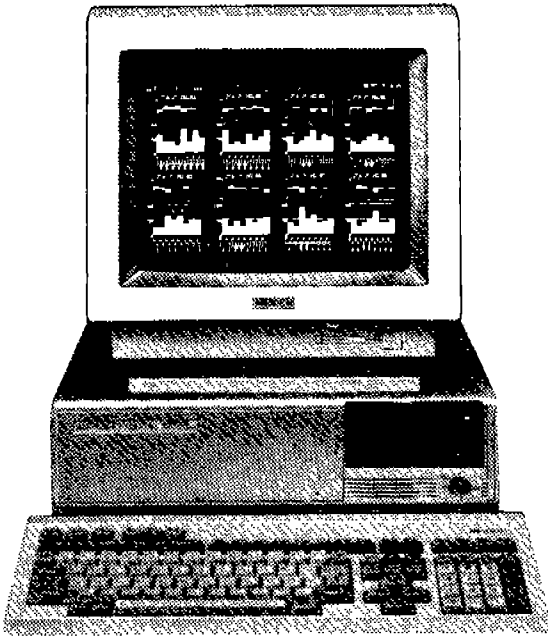


〈실무사례를 응용한〉

# IBM 퍼스널 컴퓨터 에 의한 컴퓨터 강좌 (1)



이근철

대한전기협회지 편수위원·공학박사

## 1. 프로그램의 논리적 흐름의 이해

컴퓨터 교육이라고 하면 일반적으로 컴퓨터 언어 (Computer Language) 만을 공부하는 것으로 생각하는 사람이 많다.

그러나 실무에 근무한 경험이 있는 경력자들이나 연구자들의 말에 의하면 컴퓨터 언어를 공부하는 것도 중요하지만 문제의 분석능력과 논리적인 흐름을 이해하고 활용하는 것이 더욱 더 중요하다고 말하고 있다.

분석능력이란 넓은 의미와 좁은 의미의 두가지로 나누어 생각할 수 있다.

전자는 전산화할 프로젝트를 작은 단위로 각기 세분한 다음 이를 논리적인 흐름에 맞도록 연결한 후 컴퓨터에 적용하여 원하는 결과를 얻기까지의 순서 및 명령군 등을 체계적으로 정리하는 기술을 말하며, 후자는 커다란 프로젝트의 내용 중 세분된 부분을 프로그래밍 하는 경우 입출력 설계, 파일 설계, 데이터 구조설계 등을 통하여 준비단계에서 최종단계까지를 분석하는

기술을 말한다.

한편 분석능력을 습득하려는 경우 넓은 의미의 경우라면 교육기관에서 기본적인 사항을 공부한 후 실무에서 실제 프로젝트를 처리해 가며 능력을 키워 나가야 하나, 좁은 의미의 분석능력인 경우에는 여러 종류의 컴퓨터를 교육기관이나 직장 등에서 습득하는 과정에서 자연스럽게 배양된다.

교육기관에서 여러가지 종류의 컴퓨터 언어를 교육하는 것은 언어습득의 필요성도 중요하지만 다수의 컴퓨터 언어를 교육하는 동안 적용되는 여러가지 예제의 풀이를 통하여 분석능력 (좁은 의미) 과 논리적인 흐름을 교육시키는 데 보다 큰 목적을 두기 때문이다.

논리적인 흐름이란 물(水)이 흐르는 것처럼 프로젝트나 프로그램의 처리순서를 결정해 가는 것으로서 이것을 이해하고 활용하는 것은 용이한 일이 아니다.

그러나 프로그램을 작성하기 전에 문제의 논리적인 흐름을 완전히 파악하여야 하기 때문에

물을 이용하여 논리적인 흐름을 설명하기로 한다.

예를 들면 대형 건물에 있어서 급수계통(給水系統)의 배관을 보면 옥상의 대형 저수조에는 각종으로 물을 보내는 큰 수도관이 연결될 것이며 이 관에는 총별로 급수하는 송수관이 이어질 것이고, 이 송수관의 이음부(결합부) 위에는 급수 여부를 결정하는 개폐기가 부착될 것이다.

이처럼 연결되어 가는 송수관은 세면기 위나 화장실까지 연결되어 어느 위치에서나 불편없이 사용할 수 있도록 설계되어야 할 것이다.

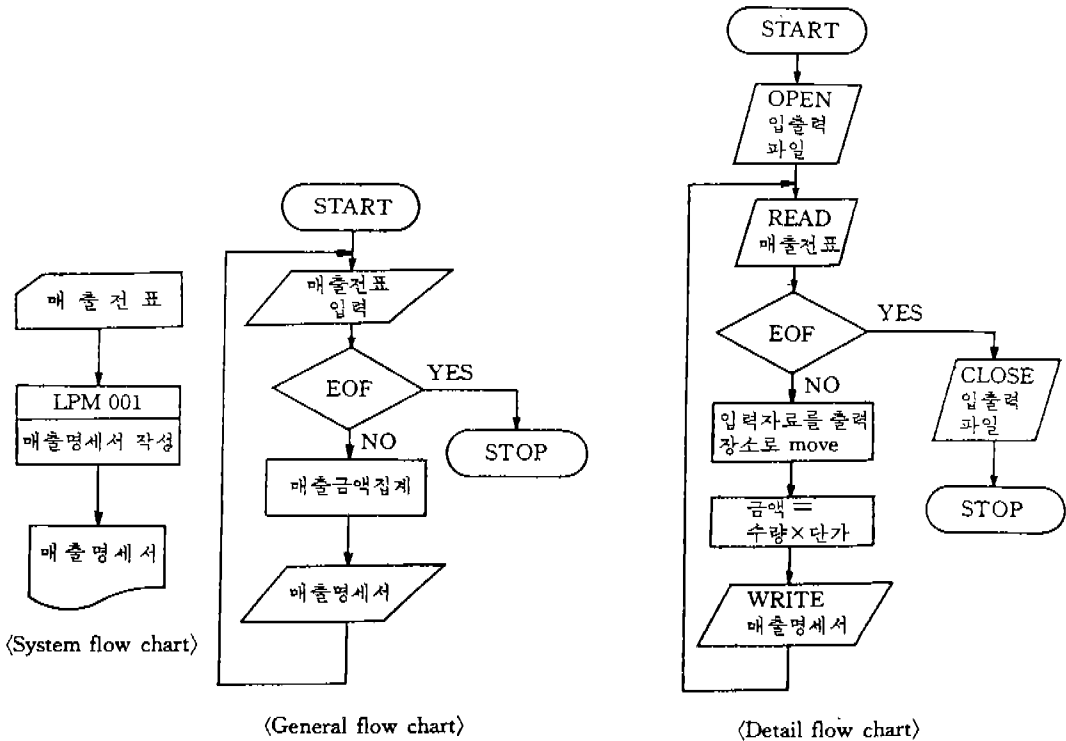
그러나 완벽한 설비가 갖추어져 있다 하더라도 원하는 층, 혹은 특정 부분에만 급수를 하려 할 경우에는 어느 개폐기를 열고 어느 개폐기를 닫아 주어야 하는지를 결정하여야 하는 문제가 발생된다.

이러한 문제를 해결하는 방법과 프로그램의 논리적인 흐름은 공통점을 가지고 있다.

즉, 물의 흐름이 막힘없이 원하는 위치의 수도꼭지에서 물이 나오도록 하는 급수계획과 입력된 데이터를 프로그램 속에 포함된 명령들 중 어떤 명령을 처리하여 원하는 결과를 얻어 낼 수 있는가를 개폐기의 개폐원리를 응용하여 프로그램을 작성하는 것이 논리적인 흐름의 기본인 것이다.

이러한 논리적인 흐름을 눈으로 확인해 가며 문제를 처리해 나가도록 정해진 도형을 이용하여 프로그램의 처리순서를 일목요연하게 나타낸 도표를 Flow Chart (순서도, 흐름도, 유통도) 라고 하는데, 프로그램이 간단할 때에는 거의 필요하지 않으나 복잡한 프로그램을 작성할 때는 순서도를 잘 작성해야만 효율적으로 프로그래밍을 할 수 있는 것이다.

다시 말하면 순서도의 역할은 프로그램 코딩의 자료가 되며, 타인에게 전달이 용이하고 문



〈그림 1〉 순서도의 종류

제처리과정과 논리적인 체계를 쉽게 파악할 수 있을 뿐만 아니라 에러가 발생할 때 수정이 용이하다는 것이다.

표 1은 순서도작성시 가장 많이 사용되는 도형을 예시한 것이다.

## 2. 순서도의 종류 및 기본형태

순서도는 자료의 흐름을 중심으로 입력자료로부터 최종 결과를 얻을 때까지 시스템 전체 및 어느 일부분을 종합적으로 표시하여 나타낸 시스템 순서도(System Flow Chart)와 시스템 전체의 작업중에서 전자계산기로 처리되는 부분의 순서도인 프로그램 순서도(Program Flow Chart)로 크게 나누어진다.

프로그램 순서도는 프로그램 전체의 흐름을 개괄적으로 표시하는 제너럴 순서도(General Flow Chart)와 세부적인 흐름을 자세히 표시하는 상세순서도(Detail Flow Chart)로 분류되는데, 이것은 프로그램 작성시 데이터의 검토나 프로그램의 기록 또는 코딩 상에 필요한 준

비를 위하여 사용된다.

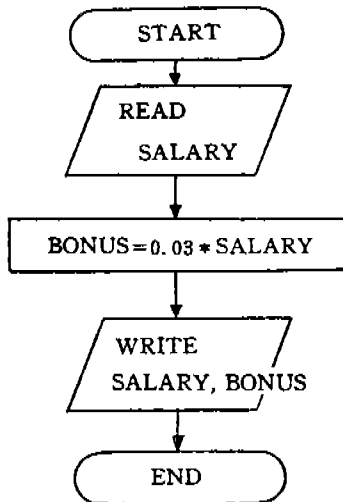
그림1은 순서도의 종류를 표시한 것이다. 다음에 순서도의 기본형태를 보면 요즘 널리 사용되고 있는 구조화 프로그래밍(Structured Programing)에서 직선형, 분기형 및 반복형 세 가지로 나눈다.

직선형은 예를 들면 1매의 카드에서 봉급액을 읽어 들여 보너스(보너스는 봉급액의 3%)를 구하여 봉급액과 보너스를 인쇄하고 끝마치기로 한다면 그림 2와 같은 순서도가 작성된다.

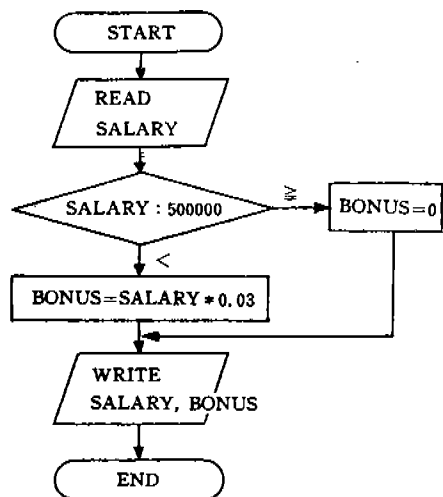
그런데 이 순서도는 어떤 조건에 따라서 분기되거나 반복되는 과정이 없이 단순하게 한번 처리하고 끝나는 순서도로서 이와 같은 형태의 순서도는 실무에서는 거의 사용되지 않는다.

분기형 순서도는 프로그램 작성시에 나타나는 어떤 조건을 비교해서 조건이 참과 거짓으로 나누어지는 형태로서, 분기조건은 두 값의 등가 여부 또는 대소관계로 표시된다.

그리고 분기로서 두 방향으로 나누어진 형태는 조건에 속한 명령을 수행한 후에 반드시 한 곳으로 모여야 한다(그림 3 참조).



〈그림 2〉 직선형 순서도



〈그림 3〉 분기형 순서도

〈표 1〉 순서도에 사용되는 기호


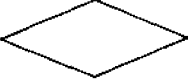

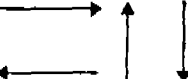



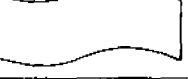





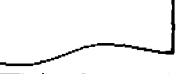
기 호	의 미	설 명
	단말 (Terminal)	시작와종료를 나타낸다
	비교판단 (Compare & Decision)	비교, 판단, 결정을 나타낸다
	처리 (Processing)	여러가지 처리를 나타낸다.
	흐름선 (Flow Direction)	작업의 흐름을 나타낸다
	결합자 (Connector)	순서도 작업중에 중단된 부분을 연결하는 기호(동일 페이지 내에서 사용)
	입, 출력 (Input/ Output)	일반적인 입·출력을 나타낸다
	종이카아드 (Punch Card)	종이카아드를 매체로하는 입 출력 기능을 표시한다.
	종이테이프 (Paper Tape)	종이테이프를 매체로하는 입·출력 기능을 표시한다.
	자기테이프 (Magnetic Tape)	자기테이프를 매체로하는 입·출력 기능을 표시한다.
	디스플레이 (Display)	처리중에 사람에게 상황을 알리기 위해 시각기(Plotter, Console, Line Printer)에 의한 출력을 나타낸다.
	정의된 처리 (Predefined Process)	서브루틴과 같이 다른 곳에서 이미 정의된 명령이나 명령군들을 처리하는 과정을 나타낸다.
	준비 (Preparation)	작업의 처리에 앞서 필요한 사항등을 결정하는 작업을 나타낸다(초기치설정, 배열 선언등)
	수작업 (Manual Operation)	off-line로 수행되는 여러가지 수작업을 표시한다.
	서류 (Documentation)	여러가지 보고서 및 서류를 나타낸다.

그림 3은 순서도를 작성하는 조건중에서 봉급액이 50만원 이상인 사람의 경우에는 보너스 액수를 제로로 처리한다는 점이 그림 2와 다르다.

그림 2를 작성하는 카드에서는 봉급액(Salary)을 읽어서 봉급액이 50만원 이상인 사람은 보너스가 없고 50만원 미만인 사람은 봉급액의 3%를 보너스로 지급한다고 할 때 이상의 조건을 처리하고 개인별 봉급액과 보너스액을 출력한다.

그리고 카드는 여러 판(板)인데 봉급액이 제로이면 끝내고, 끝날 때는 봉급액과 보너스의 전체 합계를 인쇄하는데, 이와 같은 내용을 조건에 따라서 순서도를 작성하면 그림 4와 같이 된다.

그림 4는 동일한 내용을 반복 처리하다가 봉급액의 값이 제로가 되어 끝나게 되는데, 이와 같이 순서도의 일정한 부분을 주어진 조건이 만족될 때까지 반복하여 수행하는 형태의 순서도를 반복형순서도(또는 루프형 순서도)라고 한다.

순서도에서 일정한 부분이 반복되어 처리되는 것을 일반적으로 루프(Loop)라고 하는데, 이 루프가 끝나게 되는 조건은 특정한 값이 만족되거나 일정한 회수만큼 반복하고 끝나치게 되는 경우 등 두가지가 있다.

그러나 이상에서 살펴 본 3가지 기본형이 그대로 실무에서 사용된다고 볼 수 없다. 이유는 실무에서는 대상업무가 복잡하고 그 규모가 크기 때문에 이들 기본형태의 원리를 이용한 복합 형태의 순서도가 작성되고 있다.

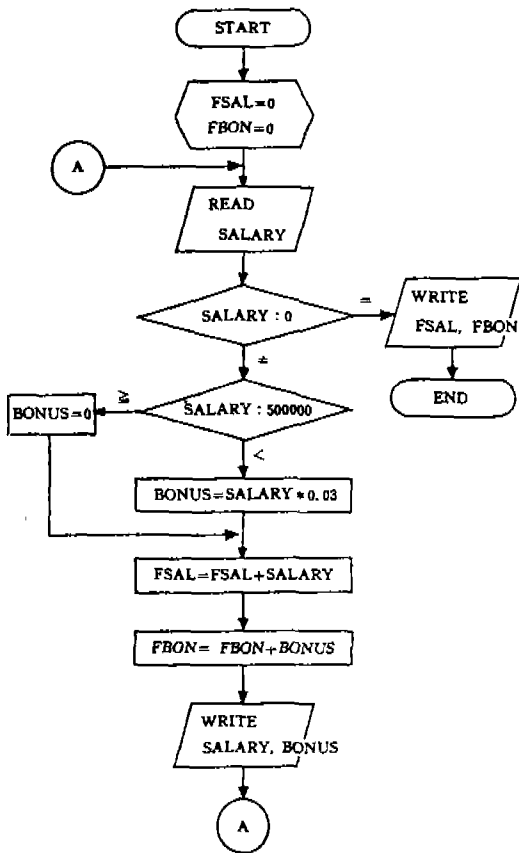
### 3. 프로그래밍 기초

프로그래밍(Programming)은 컴퓨터가 어떠한 업무를 처리, 계산하는 순서를 지시하는 프로그램을 작성하는 것을 말한다.

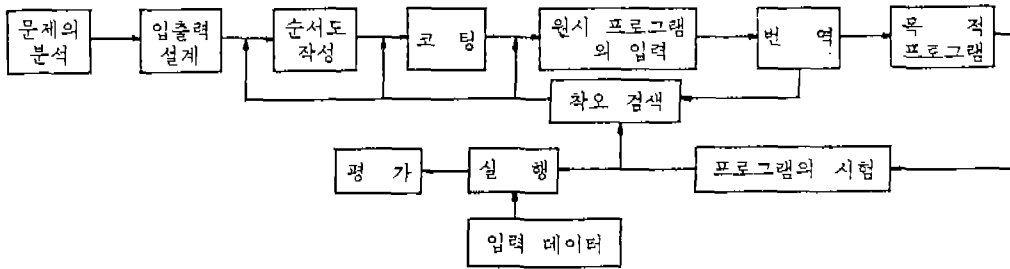
즉, 컴퓨터는 처리능력과 판단능력을 갖고 있기 때문에 어떠한 업무를 처리하고자 할 때는 업무의 내용을 면밀히 분석하고 처리순서를 정하여 신속 정확하게 자료가 처리되도록 하기 위하여 어떤 형태의 데이터를 절차에 따라 어떠한 방법으로 처리하여 결과를 어떻게 출력할 것인가에 대한 구체적인 지시가 필요하다.

따라서 원하는 결과를 얻도록 처리의 절차와 방법을 지시하는 명령의 순서를 프로그램이라고 하며, 프로그램을 작성하는 작업을 프로그래밍이라고 한다.

프로그램에는, 급여관리, 인사관리, 학사관리, 자재관리 프로그램 등과 같이 사용자가 직접 작성하는 응용 프로그램과 오퍼레이팅 시스템(OS)의 이용에 대해서 도입처의 총합적인 생산성의 향상을 생각하면서 이들의 계획과 생성, 유지,



(그림 4) 반복형 순서도



〈그림 5〉 프로그래밍의 순서

확장 및 관리하는 시스템 프로그램이 있다.

프로그래밍은 문제를 해결하기 위한 전자계산기의 명령의 순서적인 나열을 의미하는 것으로서, 프로그래밍의 절차는 다음 순서에 준하여 프로그램을 작성해야 한다.

(1) 처리하고자 하는 문제가 대두되면 이 문제를 컴퓨터로 처리하려고 할 때 그 문제의 특성, 사용할 수 있는 전자계산기의 기억용량, 처리시간, 처리비용, 프로그래머 구성 등을 면밀히 검토한 후에 결정할 것이다.

이때 문제의 구조화가 논리화되지 않으면 전자계산기로 이 업무를 처리할 수 없다는 것으로 판명되어 전산화가 될 수 없을 것이다.

(2) 업무를 처리하는 데 필요한 입력자료와 출력자료가 무엇인지를 구체적으로 명시하고 결정한다.

(3) 업무를 분석한 결과와 입력자료를 기초로 하여 업무의 해결방안을 순서적으로 나타내는 순서도(Flow Chart)를 작성한다.

(4) 프로그램의 순서도가 완성되면 특정한 프로그래밍 언어로 명령문을 기술하여 나간다.

이것을 코딩(Coding)이라고 하며 코딩용지는 프로그래밍 언어에 따라 기술하는 형식이 다르다.

(5) 작성된 원시 프로그램(Source Program)은 사람 중심이므로 컴퓨터가 직접 알아들을 수

없으므로 각종 입력장치를 통하여 컴퓨터에 입력시킨다.

(6) 프로그램 정정작업을 디버깅이라 하며, 이것은 수작업을 하는 동안 에러(Error)가 발생하는 것이 보통이기 때문에 이것을 하나하나 체크해야 한다.

유능한 프로그래머는 에러를 신속히 찾아서 처리할 수 있는 능력을 가져야 한다.

(7) 기계어로 번역된 목적 프로그램은 컴퓨터의 입·출력장치와 같은 여러 가지 장치를 움직이게 하는 시스템 프로그램과 결합되어 데이터를 처리할 수 있는 프로그램이다.

이것을 실행 가능한 프로그램이라고 하며, 오퍼레이팅 시스템(OS)이 맡아주므로 프로그래머는 크게 관여할 필요가 없다.

다만 실행 가능한 프로그램에 데이터를 입력시켰을 때 원하는 답을 얻을 수 있는가를 검사해야 한다.

(8) 프로그램 테스트가 끝나면 데이터를 입력시켜 프로그램을 실행한다.

(9) 완성된 프로그램이라 할지라도 항상 그 프로그램의 효율성과 경제성을 평가하여 보다 능률적인 처리방법이 있을 경우에는 프로그램을 수정하거나 갱신, 추가하여야 한다.

〈다음 호에 계속〉