60200307 우연정

1. 컴퓨터와 IO Device가 어떻게 통신하는가?
2. 제어용 문자(Control Data)

1. 컴퓨터와 IO Device가 어떻게 통신하는가?

키보드에서 무언가를 받아들이면 CPU로 들어오는데, CPU로 들어온 것이 다시 외부 디바이스로 나가게 된다. 키보드와 CPU는 완전히 다른 장치이다. 독립적인 장치 사이에서 정보가 소통되기 위해서는 언어가 정의되어야 한다.
즉 디바이스와 디바이스 사이에서는 표준화된 코드가 필요하다. 코드, 문법, 의미가 표준화되어야 한다. 아스키코드는 그중에서도 아주 간단한 언어이다. 언어가 만들어지려면 문법과 의미가 있어야 하고, 그 문법을 구성하는 문자가 필요하다. 코드와 문법과 의미는 다른 것이다. 예를 들어, 한글은 글자이다. '한국말'은 꼭 한글이 아니라 알파벳으로도 표현할 수 있다.

OS에서 키보드는 실제 키보드가 아스키코드를 만드는 것이 아니고, 키보드 키를 치면 아스키로 가지 않고 고유의 값인 키보드의 넘버, 키값으로 간다. 이는 아스키가 아니며, 키보드를 만드는 업체가 윈도우와 합의를 해서 윈도우에 몇 번째 키인지를 약속한 것뿐이다. 이를 받으면 OS가 디바이스 드라이브를 통해서 아스키코드나 유니코드로 바꾼다. 그러나 꼭 아스키로 바꾸는 것은 아니고, 나라마다 그 나라의 언어로 바꾼다. 실제로 수많은 언어 set을 키보드가 가지고 있을 수는 없다. 키보드를 치면 OS가 기본적으로 아스키로 보고, 한영 키를 누르면 한글이나 영어로 바뀌는 것이다.

System.in.read로 숫자 3을 입력하고, int로 저장하여 출력하면 51로 출력된다. 컴퓨터 내부에서는 51이 숫자인지 문자인지 알 수 없고, 프로그래머가 그 타입을 정하는 것에 따른다. 입력한 그대로 3을 출력하고 싶으면 charactor로 받아야 한다.
우리가 3을 입력하면 아스키코드로 만들어지고 그 숫자들을 일정한 단위로 끊어서 경우의 수를 만든다. 그 경우의 수를 가지고 코드나 문자나 숫자 등의 데이터 타입을 정의해서 어떤 데이터 타입이냐에 따라서 똑같은 값이 들어가도 다른 데이터 타입이 되는 것이다.

2. 제어용 문자(Control Data)

숫자 234를 입력했을 때, 2가 200인지 20인지 2인지 알기 위해서는 '끝났다'라는 표시가 필요하다. 이를 제어용 문자라고 한다. 제어용 문자와 내용으로 보는 문자로 구분되어 있어야 한다. 234를 입력하고 엔터를 치고 나서 키보드가 실제로 읽기 시작한다. 엔터를 치면 그제야 OS가 읽는다. 엔터를 치지 않으면 키보드의 메모리에 저장된다. 엔터도 아스키코드이기 때문에 엔터가 들어온 것을 알 수 있다. 엔터를 아스키코드로 출력하면 13, 10이 나온다. 13은 carrage return(Hex로 0D), 10은 line feed(커서 위치를 다음 줄 앞으로 내림, Hex로 0A)이다. 그래서 234를 입력하면 키보드가 32, 33, 34, 0D, 0A로 받는다. 반복한다면, 숫자를 읽기 위해서는 0D가 오기 전까지만 읽어야 한다.

아스키코드에서도 시각적인 문자(내용으로 보는 문자)는 Hex 20부터이다. 19 이하는 모두 제어용 문자이다. 사람들은 끝나는 문자를 보이는 문자로 쓴다. (스페이스, 마침표 등) 그러나 스페이스나 마침표 등은 컴퓨터가 제어용 문자인지 내용 문자인지 구별할 방법이 없다는 문제가 있다. 스페이스의 경우는 하얀 바탕을 칠한 문자나 마찬가지이기 때문이다.

컴퓨터에서 모호성이 있으면 안 되기 때문에 제어용 문자가 나온 것이다. 대표적인 제어용 문자가 엔터이다. 입력을 받을 때는 엔터를 친다. 제어용으로 제일 많이 쓰는 마침표는 null이다. 파일이나 통신이 끝났다는 것도 대부분 null을 쓴다. String이 끝나면 null을 집어넣는다.