4.3 형변환

개요

C에는 int형, float형, char형을 포함한 여러 개의 자료형이 있습니다. 가끔 변수를 한 자료형에서 다른 자료형으로 변환시켜줘야 할 때가 있는데, C는 **형변환(Typecasting)**을 통해 이것을 해결합니다. 형변환할 때 정밀도(표현 범위)가 더 높은 자료형으로 바꿀 경우 값에 오차가 발생할 수 있다는 점을 유의해야 합니다.

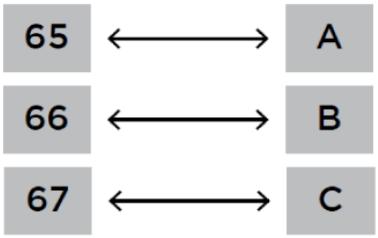
핵심개념

* 형변환
* 명시적 형변환
* 암묵적 형변환

char형과 int형 사이의 형변환

```
1  #include <stdio.h>
2
3  int main(void)
4  {
5      int x = 65;
6      printf("%d\n", x); // 65
7      printf("%c\n", (char) x); // A
6  }
```

▲ 〈코드1〉



ASCII 표준은 각 문자에 그 문자를 식별할 수 있는 고유 숫자를 부여했습니다. 대문자 A는 65, 대문자 B는 66과 같이 숫자가 할당되어 있습니다. 그래서 **형변환**을 통해 int형과 char형을 서로 변환할 수 있습니다.

〈코드 1〉의 5행을 보시면 x라는 int형 변수가 65의 값을 가지고 있습니다. 화면에 그 변수를 출력하기 위해 6행처럼 쓰면, 65과 출력됩니다. 7행에는 int형이 아닌 char형을 위한 형식문자 %c가 있습니다. 6행과 마찬가지로 x의 값을 출력하지만 **변수 이름 앞에 (char) 이 작성**되어 있어 char형으로 변환됩니다. **이미 존재하는 변수 앞에 새로운 자료형을 넣어 다른 자료형으로 바꿔주는 것**을 **명시적 형변환**이라고 합니다. 우리가 직접 자료형을 변환하라는 지시를 내리는 것이죠. 이 상황에서 명시적 형변환은 사람들이 코드를 보고 이해하기 쉽기 때문에 더 좋은 방식이라고 할 수 있습니다.

사실 char형의 형식문자를 썼기 때문에 컴파일러는 문자가 전달될 것이라고 여겨, 7행의 (char)을 빼도 코드는 여전히 ASCII 65값이 나타내는 문자 A를 출력할 것입니다. ASCII 표준에 해당하는 정수를 %c에 전달하면 컴파일러는 그 값을 자동으로 문자로 해석하기 때문입니다. 이것을 **암묵적 형변환**이라고 합니다.

int형과 float형 사이의 형변환

```
1  #include <stdio.h>
2
3  int main(void)
4  {
5      int a = 28;
6      float b = a / 5;
7      float c = (float) a / 5;
8
9      float d = 28.523;
10     int e = d;
11 }
```

▲ 〈코드 2〉

형변환은 부동 소수점 수와 정수의 변환에 아주 유용하게 쓰입니다. 〈코드 2〉를 보면, 6행에서 28에 5를 나눈 값인 5.6을 b에 저장되어야 하는데, 실제로는 b의 값이 5.0이 됩니다. 이것은 컴파일러가 정수 두 개를 나누었기 때문에 그 결과도 정수로 나온 것입니다. 이 문제를 해결하기 위해 7행과 같이 a를 float형으로 **명시적 형변환**하면 그 결과값으로 c는 5.6이 됩니다.

암묵적 형변환도 int형과 float형 변수들을 다룰 때 유용하게 쓰입니다. 정수는 소수점 뒤의 숫자들에 대한 정보를 저장할 수 없기 때문에 float형인 값을 int형으로 변환하면 부동 소수점 수의 소수점 뒤를 버린 값을 쉽게 저장할 수 있습니다. 10행을 보면 int형인 변수에 부동 소수점 값을 넣으려고 했을 때, d는 묵시적으로 int형으로 형변환되며, 이 과정에서 소수점 뒤의 값을 버리게 됩니다. 결과적으로 e의 값은 28입니다.