C02 학번:2019202052 이름: 김호성

P.119쪽 연습문제

#include<stdio.h>

int main()

{

char abc; //문자입력 받을 변수 abc선언

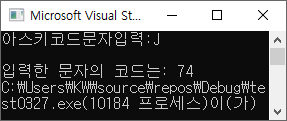
printf("아스키코드문자입력:");

scanf("%c", &abc); //scanf에는 줄바꿈들어가면 안된다.

printf("\n입력한 문자의 코드는: %d", abc);

return 0;

}



p.122쪽 예제

#include <stdio.h>

int main()

{

printf("literal int size: %d \n", sizeof(7));

// int의 데이터크기를 묻는 것이다. int의 범위안에 있는 어떤 숫자를 입력하든 4가 출력된다.

printf("literal double size: %d \n", sizeof(7.14));

// double의 데이터크기를 묻는 것이다. double의 범위안에 있는 어떤 숫자를 입력하든 8이 출력된다.

printf("literal char size: %d \n", sizeof('A'));

// char의 데이터크기를 묻는 것이다. char의 범위안에 있는 어떤 숫자를 입력하든 4이 출력된다.

printf("literal float size %d \n", sizeof(float));

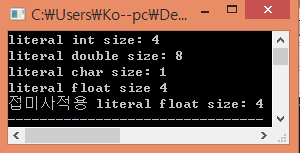
// float의 데이터크기를 묻는 것이다. floatdl라는 자료형을 입력해도 바이트 수가 표현된다는 것을 보여주기 위해 float을 사용했다.

printf("접미사적용 literal float size: %f ", sizeof(3.14f));

//접미사를 적용한 literal상수

return 0;

}



p.126쪽 예제

#include <stdio.h>

int main()

{

double num1 = 245;

int num2 = 3.1415;

int num3 = 129;

char ch = num3;

//char형에서 바로 129를 입력하면 char형의 범위를 넘는 수 이므로 경고가 뜬다.

//상수: 변하지 않는 변수(메모리위치), 리터럴: 변수의 값이 변하지 않는 데이터(메모리 위치 안의 데이터)

printf("정수 245를 실수로: %f \n", num1);

//형변환을 발생하지 않게 하려면 %d가 맞다.

printf("정수 245를 실수로: %lf \n", num1);

//%f와 %lf의 값의 차이를 보기위해 작성했다.

printf("실수 3.1415를 정수로: %d \n", num2);

//소수점아래부분은 삭제된다.

printf("큰정수 129를 작은정수로: %d \n", ch);

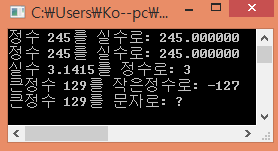
//char형을 넘는 숫자이므로 값이 달라진다.

printf("큰정수 129를 문자로: %c \n", ch);

//아스키코드에는 음수가 없다. 즉 값이 지정되어 있지 않으므로 ?가 출력된다.

return 0;

}



p.129쪽 예제

#include <stdio.h>

int main()

{

int num1, num2;

double divResult;

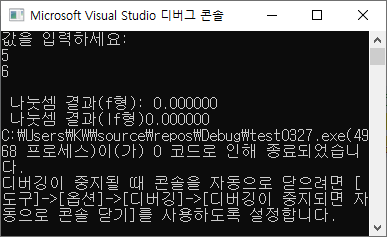
printf("값을 입력하세요: ");

scanf("%d %d", &num1, &num2); //숫자입력받는 공간

divResult = num1 / num2; //계산실행

printf("\n 나눗셈 결과(f형): %f\n 나눗셈 결과(lf형)%lf", divResult, divResult);

return 0;

}

5.용어풀이

5-1 ISO/IEC 646

ISO/IEC 646는 7비트의 문자 코드를 규정하는 국제 표준화 기구(ISO) 표준이며, 이것을 바탕으로 각 나라 언어판의 문자 코드가 규정된다. 미국의 규격인 ASCII를 국제화한 체제이다. 라틴 문자와 숫자를 중심으로 하는 공통 부분과, 각 나라 규격으로 자유롭게 할당할 수 있는 부분(ASCII에서는 기호에 해당)으로 나뉘어 있어 유럽 각 나라에서는 이 부분에 발음 구별 기호를 더해 문자를 할당하여 자국어의 문장을 인코딩할 수 있게 되어 있다. 현재는 유럽에서는 8비트 코드인 ISO/IEC 8859가 주류이기에, 이 규격은 그다지 많이 사용되지 않았다.

5-2 IBM 코드 페이지

코드 페이지(code page)는 특정한 문자 인코딩 테이블을 위해 쓰이는 전통적인 IBM 용어이다. 문자 인코딩 테이블은 0부터 255까지의 정수를 표현하는 단일 옥텟(octet, 바이트)이라고 불리는 일련의 비트들이 특정한 문자와 결합하여 도표화(mapping)한 것이다. IBM과 마이크로소프트는 코드 페이지를 문자열 집합(charset)에 자주 할당한다.(우리나라는 949번이다.)

5-3 ISO 8859

ISO/IEC 8859는 컴퓨터에서 8비트로 문자를 나타내기 위한 ISO와 IEC의 공동 표준이다. ISO/IEC 8859-1, ISO/IEC 8859-2 등의 세부 표준이 제안되어 있다. ASCII에 포함된 95자의 인쇄 영역은 현대 영어로 정보를 표현하는 데 충분하다. 하지만 로마 문자를 쓰는 다른 언어에는 ASCII로 처리할 수 없는 추가적인 기호를 포함하고 있다. 예를 들면 ß(독일어), ñ(에스파냐어), å(스웨덴어와 북유럽 언어) 등이다. ISO 8859는 8비트로 구성된 바이트의 8번째 비트를 이용하여 128자의 추가적인 문자를 할당하여 이 문제를 처리하기 위해 제안되었다. 하지만 한 종류의 문자 집합만으로는 필요로 하는 모든 문자를 처리할 수 없어 여러 가지 매핑이 개발되었다.