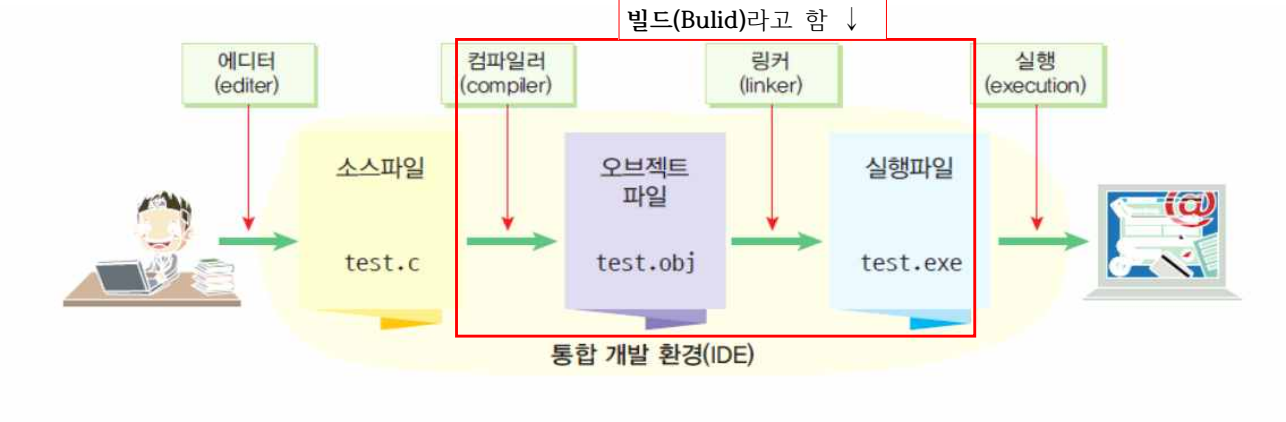
1. 프로그래밍이란 프로그래밍 언어를 이용해서 프로그램을 만드는 것

2. 프로그래밍 언어란?

- 컴퓨터에서 일을 수행하는 프로그램을 작성하기 위해 사용하는 기호 체계를 의미
- 즉, 사람과 컴퓨터가 이해할 수 있는 약속된 형태의 언어를 의미한다.
- 우리가 배울 C언어도 프로그래밍 언어 중 하나이다.

3. 프로그램이 만들어지는 과정

비주얼스튜디오에서는
빌드(Bulid)라고 함 ↓



(+) **소스파일**이란? 프로그래밍 언어로 작성된 프로그램

(+) **컴파일러란?** 프로그래밍 언어로 작성된 프로그램을 컴퓨터가 이해할 수 있는 **기계어**로 번역하는 역할을 **컴파일러**라고 하고 이러한 작업을 **컴파일** 한다고 함

(+) **기계어란?** 컴퓨터가 이해할 수 있는 0과 1(이진수)로 구성된 형태의 언어

(+) **링커란?** 오브젝트 파일과 표준 라이브러리 함수를 연결하여 하나의 파일로 통합하는 역할을 수행

(※표준 라이브러리 함수란 C언어에서 기본적으로 제공하는 함수들의 모임)

(+) **통합 개발 환경이란?** 프로그램을 개발하는 과정에서 에디터, 컴파일러, 링커 등의 각 단계가 모두 하나의 프로그램 속에 통합되어 있는 형태

(※C언어 통합 개발 환경에는 비주얼 스튜디오, Dev-C++, 코드블록 등이 있음)

4. C언어의 모든 것

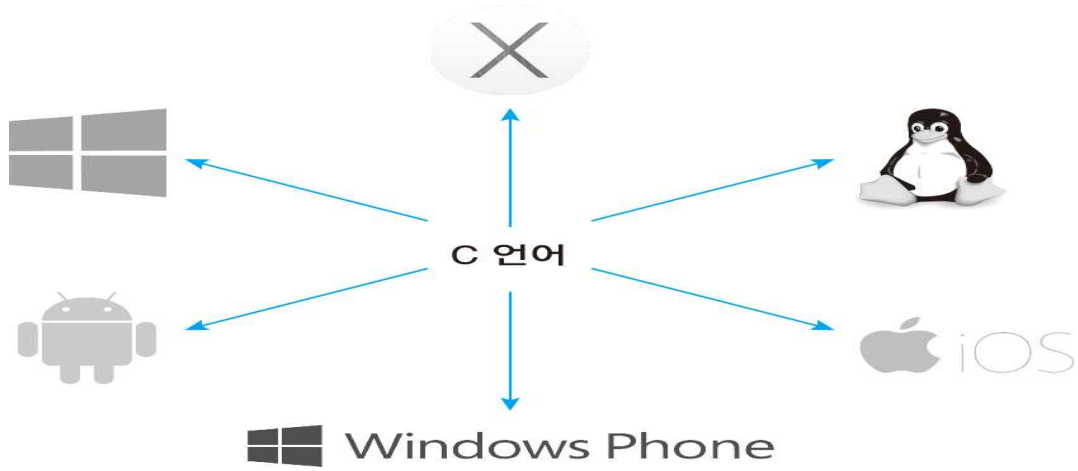
BCPL → B 언어 → C 언어 → C++
Java
C#
Objective-C
PHP
Python
Ruby
⋮

- C언어는 1972년 BCPL언어를 고쳐 B언어(Bell연구소에서 B를 따서 지음)를 개발했었는데 데니스리치가 B언어를 개선하여 C언어가 탄생함

■ C언어를 공부해야 하는 이유

- 이후의 프로그래밍 언어에 영향을 주었기 때문에 다른 언어를 학습하기 쉬워짐
- C언어를 배움으로써 컴퓨터의 기본 동작원리를 이해하게 된다.
- 프로그래밍 언어에 대한 문법을 이해할 수 있다.
- 표현 능력을 향상시킬 수 있다.

(+) https://www.youtube.com/watch?v=pYYnycoVmB8



C언어는 우리가 알고있는 운영체제, 데이터베이스, 여러 가전 제품 등에 많이 사용됨

■ C언어의 장점

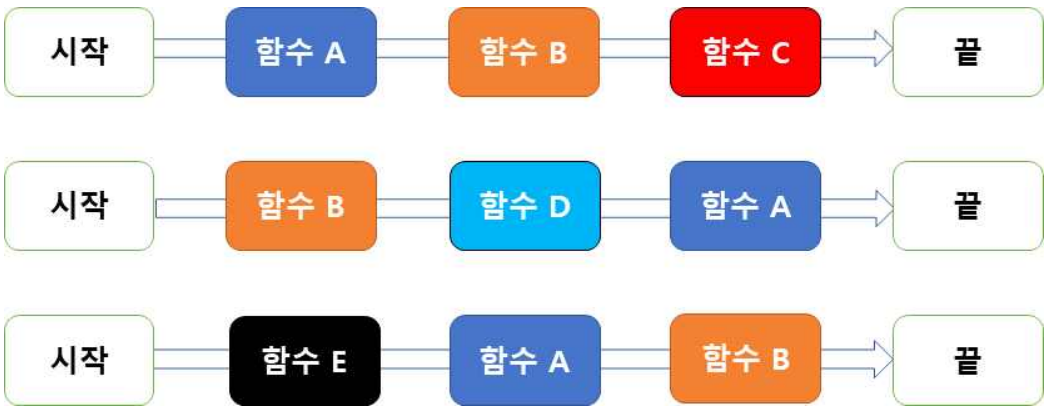
- 낮은 레벨에서 어떻게 동작하는지에 대해서 생각할 수 있게 된다.
- 메모리 관리에 대한 개념이 생긴다.
- 성능에 대해 생각하게 된다.
- 문자열을 관리하는 방법에 대해서 생각한다.
- 메모리상에 데이터들이 어떻게 배치되는지 생각할 수 있게 된다.
- 쉽게 익숙해질 수 있다.

5. C 프로그램 기본 예제

```c
#include<stdio.h>
int main(void)
{
   printf("Hello World\n");
   return 0;
}
```

■ C 언어의 기본단위인 '함수'의 이해

- "C언어는 함수로 시작해서 함수로 끝난다!"
- C 언어로 프로그램을 작성한다는 것은 함수를 만들고, 만든 함수들의 실행순서를 결정하는 것임
- 정해진 순서에 의해서 진행되는 함수의 호출이 바로 프로그램의 흐름이 되는 것임



■ main 함수의 형태

```c
int main(void) {

    return 0;
}
```

■ 함수의 전체적인 구조

```
반환형  함수이름( 매개변수 ) {
      함수 몸체
  반환
}
```

■ 표준 라이브러리와 printf 함수

```
printf("학번 : 1234, 이름 : 홍길동 \n");
```

- printf 함수는 함수 호출 시 전달되는 문자열을 모니터에 출력하는 기능
- 직접 만들지 않아도 호출이 가능한, 기본적으로 제공되는 함수 => (표준함수)
- 표준함수들의 모임을 '표준 라이브러리'라고 함
- \n은 줄을 바꿀 때 사용하는 줄바꿈 문자(즉, 엔터임)

■ 함수내에 존재하는 문장의 끝에는 세미콜론 문자 ;를 붙여준다.

- 함수 내부에 각 문장의 끝에는 세미콜론 문자 ; 가 붙어 있음.
- C언어는 문장의 끝을 표현하기 위해서 세미콜론을 사용함.

■ 헤더파일 선언의 필요성

```
#include <stdio.h>
```

- 헤더파일이란 외부에서 정의된 소스파일이며 미리 정의된 함수나 변수를 사용하기 위해 만들어짐.
- 위에서 헤더파일은 stdio.h이고 여기에는 printf 함수의 호출에 필요한 정보가 존재함.
- 따라서 이 파일의 정보를 포함하는 헤더파일 선언문이 삽입되어야 함.
- 헤더파일의 선언은 소스파일 맨 앞부분, main 함수 정의 이전에 와야 함.

  (+) stdio는 standard input output의 약자로 표준입출력과 관련된 함수들이 들어있는 파일

■ return은 함수의 종료와 값의 반환이라는 두가지 의미

```
return 0;
```

- 함수를 호출한 영역으로 값을 반환
- 현재 실행 중인 함수의 종료

■ 최종 정리

```c
#include<stdio.h> //헤더파일
int <-반환형 main <-함수이름()
{ //함수 몸체 시작
   printf("hello\n"); //표준 함수 printf
   return 0; //함수 종료를 의미하는 return
} //함수 몸체 끝
```

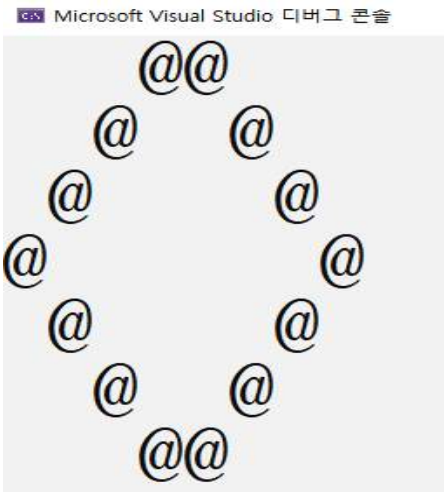
문제 01-1 [예제 변경해보기]

지금까지 공부한 내용을 가지고 간단한 프로그램을 작성해보자. 위의 기본 예제를 변경해서 작성해도 된다. 완전히 이해하지 못했어도 상관없다. 참고로 이렇게 예제를 변경해 보는 것은 많은 공부가 된다.

1. 다음과 같은 형태로 본인의 이름을 출력하는 프로그램을 작성해보자. 단, printf 함수는 한번만 호출해야 한다.(줄바꿈 문자는 \n)

2. 본인의 이름, 학번, 그리고 전화번호를 출력하는 프로그램을 작성해보자. 더불어 이스케이프 시퀀스 \n을 적절히 삽입해서 출력형태를 보기 좋게 다듬기 바란다. 총 3번 printf 함수를 호출해서 문제를 해결하자.

3. 화면에 다음과 같은 그림이 출력될 수 있도록 printf 함수와 이스케이프 시퀀스 \n을 적절히 사용하여 문제를 해결하자.



(+) **이스케이프 시퀀스란?** '\'는 이스케이프 문자라고 하며 화면에는 표시되지 않고 다른 문자와 함께 쓰여 특수한 동작을 수행. 이러한 문자들을 이스케이프 시퀀스라고 한다.

ex) ▸ \n : 줄바꿈 ▸ \t : 탭 기능 ▸ \\ : 화면에 '\'를 표시

2. “주석”을 넣어보자

■ 주석은 선택이 아닌 필수!!

- 주석(comment)은 프로그램 내에 삽입된 메모
- 컴파일 대상에서 제외됨
- 주석의 유무는 프로그램의 실행결과에 영향을 미치지 않음

■ 블록 단위 주석 (단축키는 드래그 후 ctr + k + c)

```
/* 주석처리 된 문장 */
```

- 아래처럼 두 줄 이상의 주석처리 사용 가능

```
/*
    주석처리 된 문장 1
    주석처리 된 문장 2
    주석처리 된 문장 3
*/
```

■ 행 단위 주석

```
//  주석처리 된 문장 1
//  주석처리 된 문장 2
//  주석처리 된 문장 3
```

■ 기본예제에 주석 추가하기

```
/*
    제목 : 학번, 이름 출력하기
    기능 : 문자열의 출력
    파일이름 : ch01.c
    수정날짜 : 2021.03.05
    작성자 : 홍길동
*/
#include <stdio.h>  // 헤더파일 선언
int main()          // main 함수의 시작
{
    /*
        이 함수 내에서는 하나의 문자열을 출력한다.
        문자열은 모니터로 출력한다.
    */
    printf("학번 : 1234, 이름 : 홍길동 \n"); // 문자열의 출력
    return 0;       // 함수 반환
}   // main 함수의 끝
```

3. printf 함수의 기본적인 이해

■ printf함수를 이용한 정수의 출력과 서식문자

```c
#include <stdio.h>
int main()
{
    printf("프로그래밍 수업 파이팅\n");
    printf("%d\n", 1234);
    printf("%d %d\n", 10, 20);
    return 0;
}
```

- 실행 결과

```
Microsoft Visual Studio 디버그 콘솔
프로그래밍 수업 파이팅
1234
10 20
```

- printf 함수를 이용하여 문자열도, 정수 데이터도 출력이 가능함을 알 수 있음.
- printf("서식문자" , [인수]) 형식으로 이루어짐.
- 위에서 %d 는 **서식문자**라고 함. 출력 형태를 지정하는 용도로 사용

서식문자	출력 내용
%d	정수(int)
%f	실수(float, double)
%c	문자(character)
%s	문자열(string)

- 서식 문자는 '%'와 출력형식을 나타내는 문자(d, f, c, s 등)가 더해져서 구성됨
- 서식 문자 자리에는 뒤에 해당하는 인수가 출력된다.(※**서식문자의 형식과 인수의 형식은 같아야 함**)

■ printf 함수를 이용한 문자열, 정수형, 실수형 출력과 서식 문자의 예시

- 정수형 출력과 서식 문자

입력 형태	출력
printf("정수 출력 %d\n", 10);	Microsoft Visual Studio 디버그 콘솔 정수 출력 10

- 실수형 출력과 서식 문자

입력 형태	출력
printf("실수 출력 %f\n", 1.4);	Microsoft Visual Studio 디버그 콘솔 실수 출력 1.400000

- 단일 문자형 출력과 서식 문자

입력 형태	출력
printf("문자 출력 %c\n", 'a');	Microsoft Visual Studio 디버그 콘솔 문자 출력 a

- 문자열형 출력과 서식 문자

입력 형태	출력
printf("문자열 출력 %s\n", “abcd”);	Microsoft Visual Studio 디버그 콘솔 문자열 출력 abcd

▶ 단일 문자와 문자열 출력시 주의 사항

> 단일 문자는 작은따옴표 ‘ ’를 사용하고, 문자열은 큰따옴표 “ ”를 사용한다.

문제 01-2 [printf 함수의 다양한 활용]

1. 다음의 출력 결과가 보이도록 예제를 작성해보자. 단, 출력되는 숫자들은 서식문자 %d를 이용하여 출력하도록 하자.

```
Microsoft Visual Studio 디버그 콘솔
제 이름은 홍길동입니다.
제 나이는 20살입니다.
제 전화번호는 010-1234-5679입니다.
```

2. 다음의 출력 결과가 보이도록 예제를 작성해보자. 이번에도 역시 출력되는 숫자들은 서식문자 %d를 이용해서 출력하도록 하자.

```
Microsoft Visual Studio 디버그 콘솔
4 * 5 = 20
7 * 9 = 63
```