1. C언어의 시작

♲ 프로그래밍 언어란?

컴퓨터의 동작을 위해 필요한 프로그램을 작성하기 위해 고안된 도구를 프로그래밍 언어라 한다.
프로그래밍 언어는 컴퓨터가 수행할 연산을 상세히 기술하기 위해 사용되며 그것을 가능하게 해주는 도구이다. 현재에는 아주 많은 종류의 프로그래밍 언어가 존재하며 프로그래밍 언어는 그 특성에 따라 고급언어와 저급언어로 구분된다.

저급언어 : 컴퓨터가 이해하고 실행하기 좋은 구조를 가지고 있는 언어이다. 저급언어는 빠른 처리속도와 작은 용량을 가지고 있지만 사람이 이해하고 배우기 어려워 다양한 명령을 내리기에는 적합 하지 않다. 따라서 저급언어는 빠른 처리 속도를 가지거나 용량이 작아야 하지만 다양한 기능이 필요 없는 프로그램을 작성할 경우 사용된다. 대표적인 저급언어로 기계어와 어셈블리어가 있다.

고급언어 : 사람이 쉽게 다양한 명령을 내리기 좋은 구조를 가지고 있는 언어이다. 고급언어는 가독성이 높고 다루기가 간단하여 프로그래머가 다양하고 복잡한 프로그램을 만들어내는 것이 용이하다는 장점이 있다. 따라서 대부분의 응용프로그램은 고급언어로 작성된다. 고급언어는 컴파일러나 인터프리터에 의해 저급언어로 번역되어 실행되며 대표적인 고급언어로는 C 언어, JAVA, BASIC 등이 있다.

♲ C 언어란?

벨 연구소에서 1971년경부터 리치(D.M.Ritchie) 등에 의해서 설계 개발된 시스템 기술용의 프로그래밍 언어이다. UNIX 오퍼레이팅 시스템의 기술에 사용할 것을 목적으로 설계한 언어로 UNIX OS의 대부분이 이 언어로 개발되었다. UNIX의 90% 이상이 C언어로 개발되었으며 어셈블리어의 도움 없이도 빠르고 성능 좋은 시스템의 개발을 가능하게 해주는 언어이다.

♲ C언어의 특징

1. 시스템 간 호환 및 이식성이 좋다.

이식성이라는 말은 어느 한 컴퓨터 시스템에서만 사용되는 것이 아니라 다양한 운영체제와 다양한 하드웨어에서도 큰 수정 없이 사용할 수 있다는 것이다. C프로그램이 약간의 수정만 거치게 되면 도스나 윈도우에서도 사용할 수 있다.

2. 고급언어와 저급언어의 장점을 모두 가지고 있다.

C언어는 사용자 중심의 고급언어(High-Level Language)이면서 하드웨어를 직접 제어할 수 있는 저급언어(Low-Level Language)의 특징을 모두 가지고 있다.

3. 비트 및 증감연산자 등 풍부한 저급언어 연산자와 다양한 제어구조를 지원한다.

직접적인 비트연산을 할 수 있는 다양한 연산자를 제공하며 if-else, switch, whie, do-whie, for, break 등과 같은 제어문을 제공하여 프로그램을 좀 더 구조적으로 만들 수 있다.

4. 함수의 집합으로 구성된 함수형 언어이며 구조적 프로그래밍 언어임으로 모듈식 구현이 용이하다.

함수라는 것은 '어떤 기능을 수행하는 프로그램 단위'이다. C언어에는 기본적인 데이터 타입, 구조체, 유니온 그리고 포인터 값을 반환하는 다양한 함수들이 있으며 이것들을 활용하여 프로그래밍을 할 수 있다.(C언어에서 제공하는 함수는 수천개가 넘기 때문에 어떤 곳에 어떠한 함수를 사용하느냐가 중요한 문제가 될 수 있다.)

5.C언어는 프로젝트 파일 기능을 제공한다. (분할 컴파일)

큰 덩어리의 프로그램을 한 파일에서 한꺼번에 컴파일하는 것이 아니라 모듈별로 각각 분리하여 저장하여 컴파일을 거쳐서 최종적으로는 하나의 실행파일을 만들 수 있다.

6.다른 고급언어에 비해 실행파일의 크기가 작고 속도가 빠르다.

♲ VISUAL C++ 6.0

1. VISUAL C++ 6.00이란?

MicroSoft사에서 개발한 Visual Studio 6.0안에 포함된 C언어 개발 패키지이며 윈도우 xp 버전에서까지 사용가능하다.

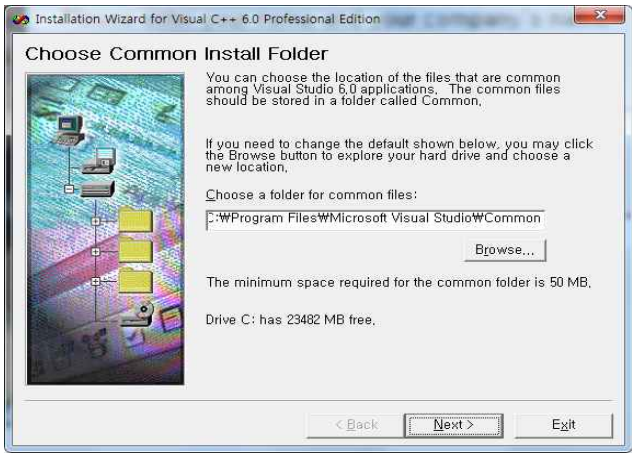
2. VISUAL C++ 6.0의 설치 방법

1) 설치안내

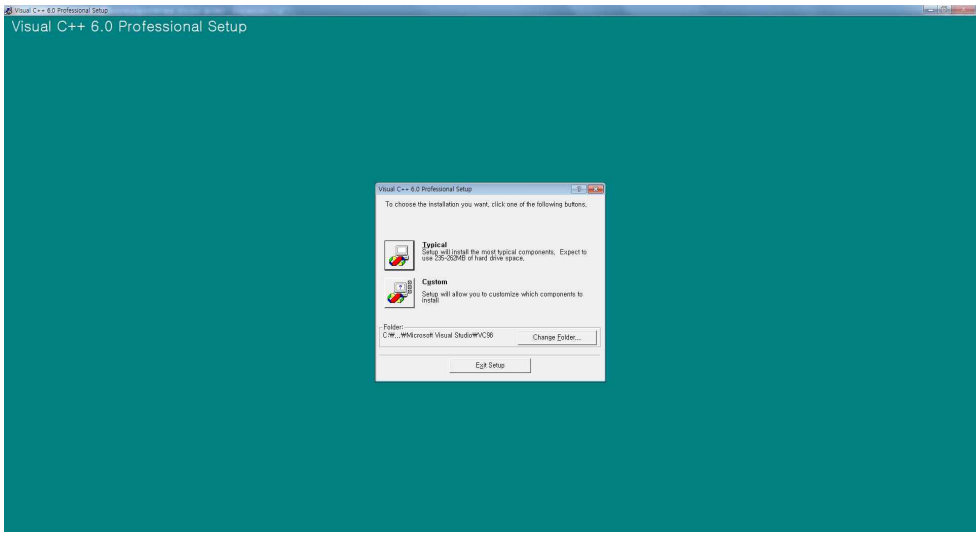
2) 라이센스 동의

3) 시리얼코드 및 이름입력

4) 설치폴더 선택

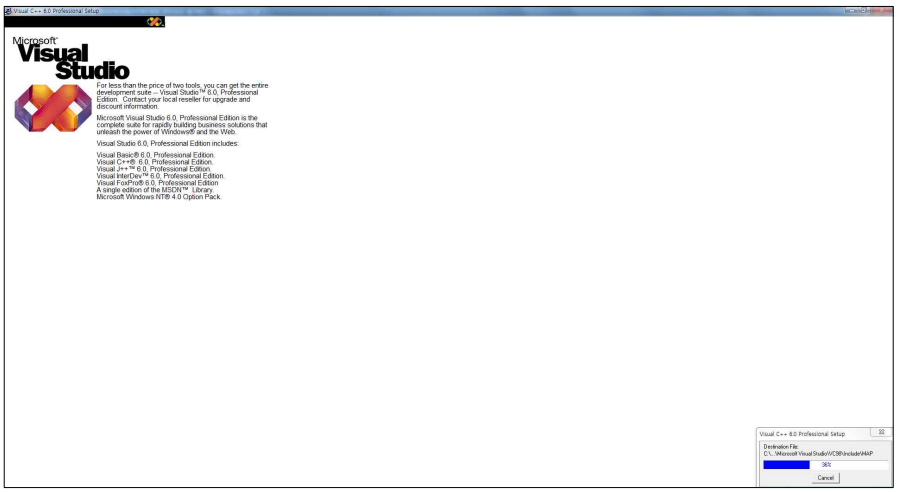


5) 설치형태 선택



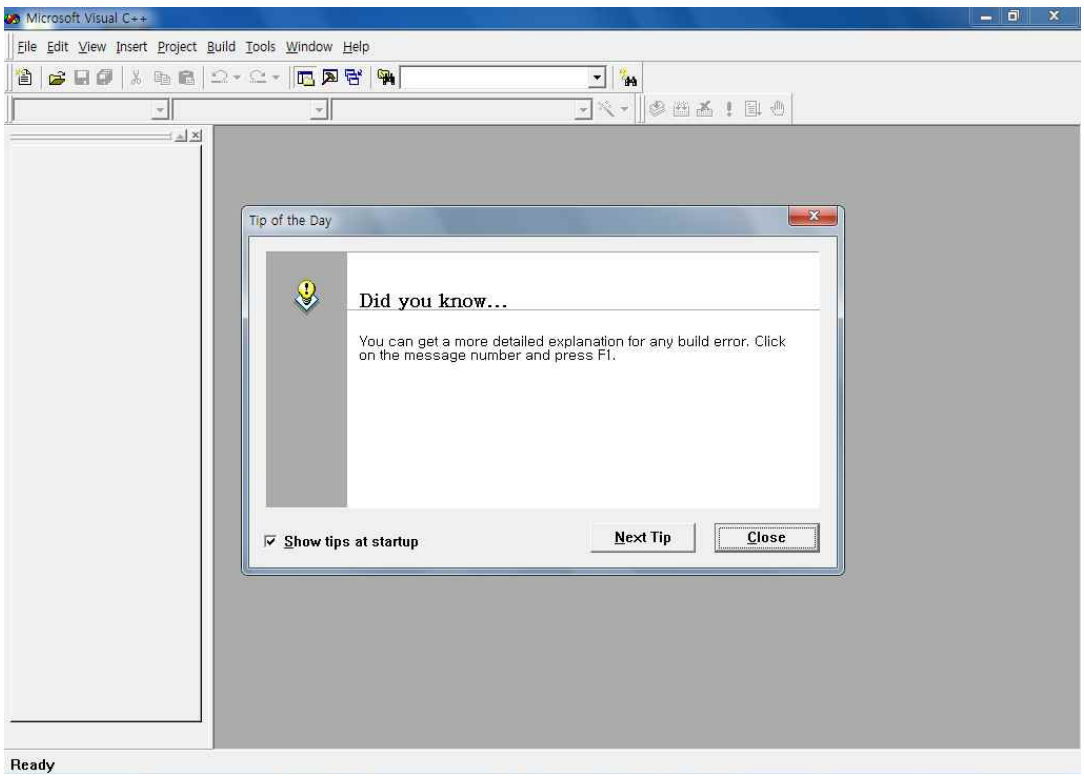
6) 환경변수 등록 여부 선택

7) 설치화면

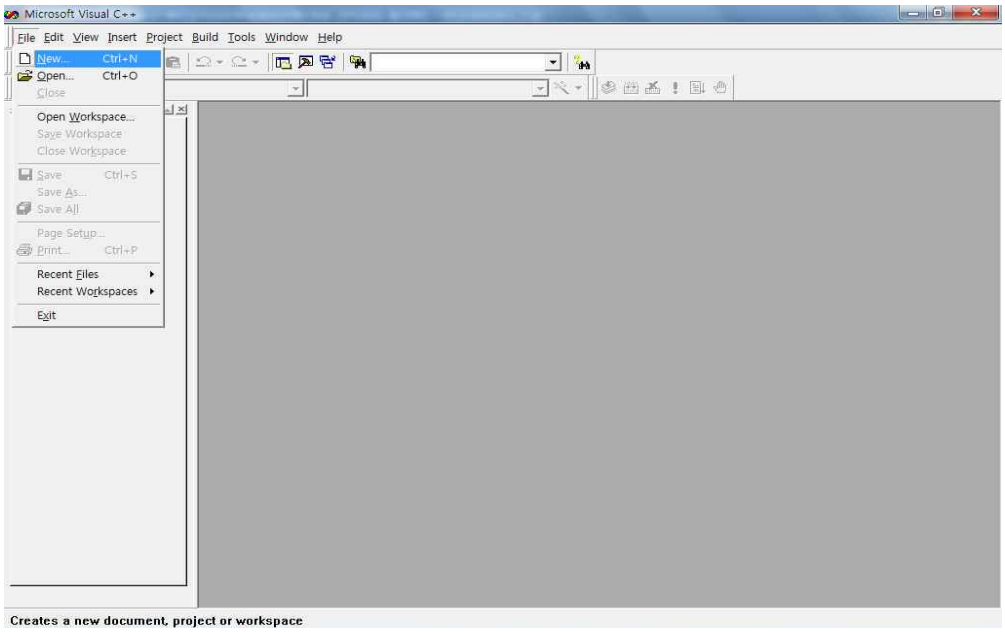


3. VISUAL C++ 6.0의 기본적인 사용 방법

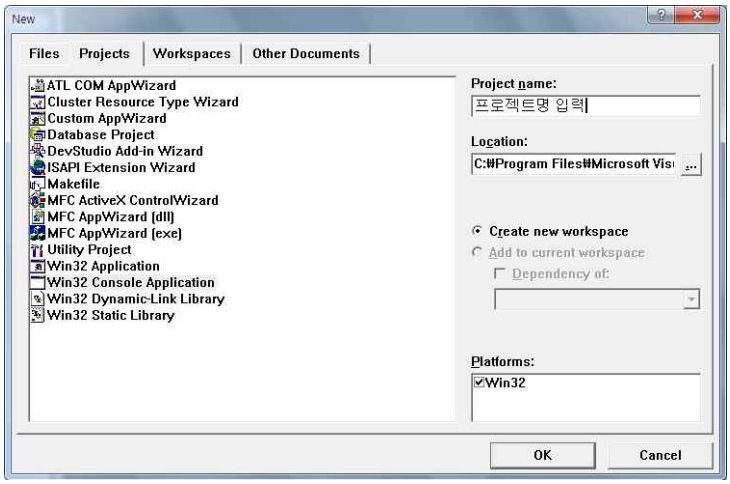
1) 윈도우의 "시작" 버튼을 클릭하여 프로그램 목록에 있는 "MicroSoft Visual C++ 6.0"을 실행 시킨다.



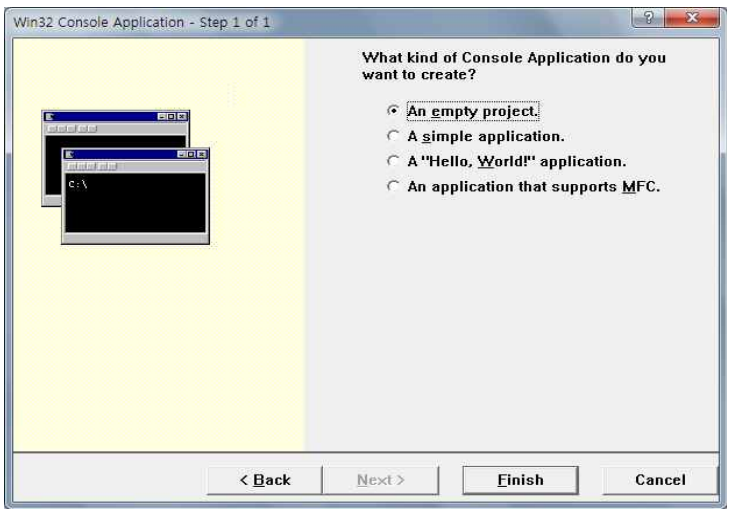
2) 새로운 프로젝트 생성 방법 1 - File 메뉴의 New를 선택한다.



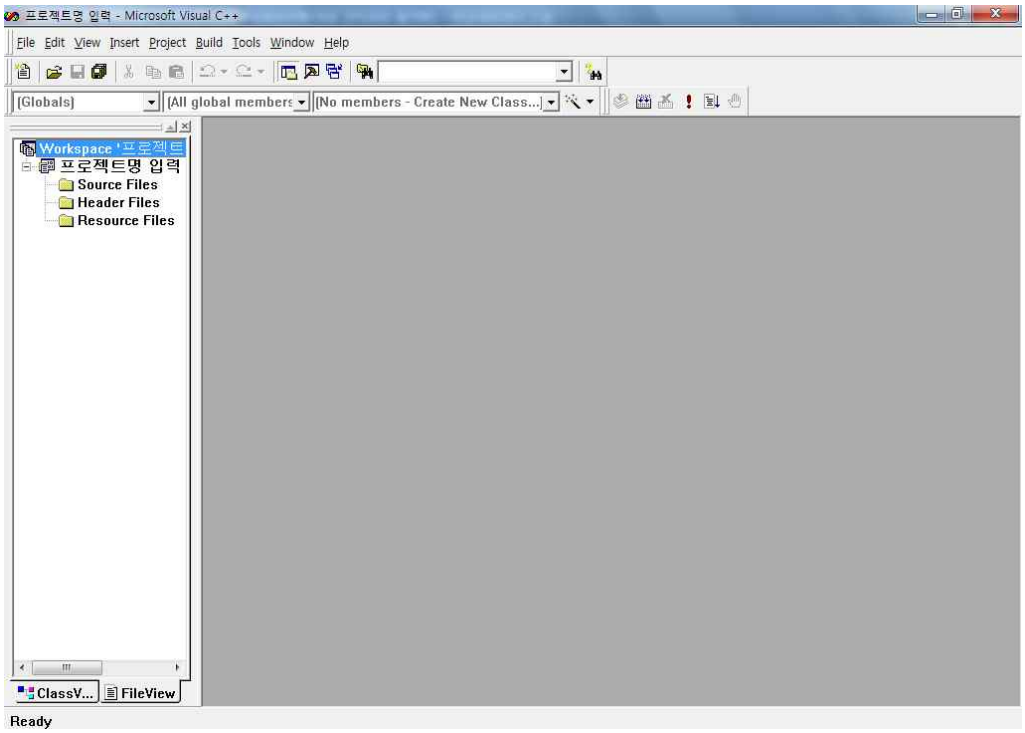
2) 새로운 프로젝트 생성 방법 2 - Project 탭에서 "Win32 Console Application"을 선택하고 Project name을 입력 한 뒤 Location을 선택한다.



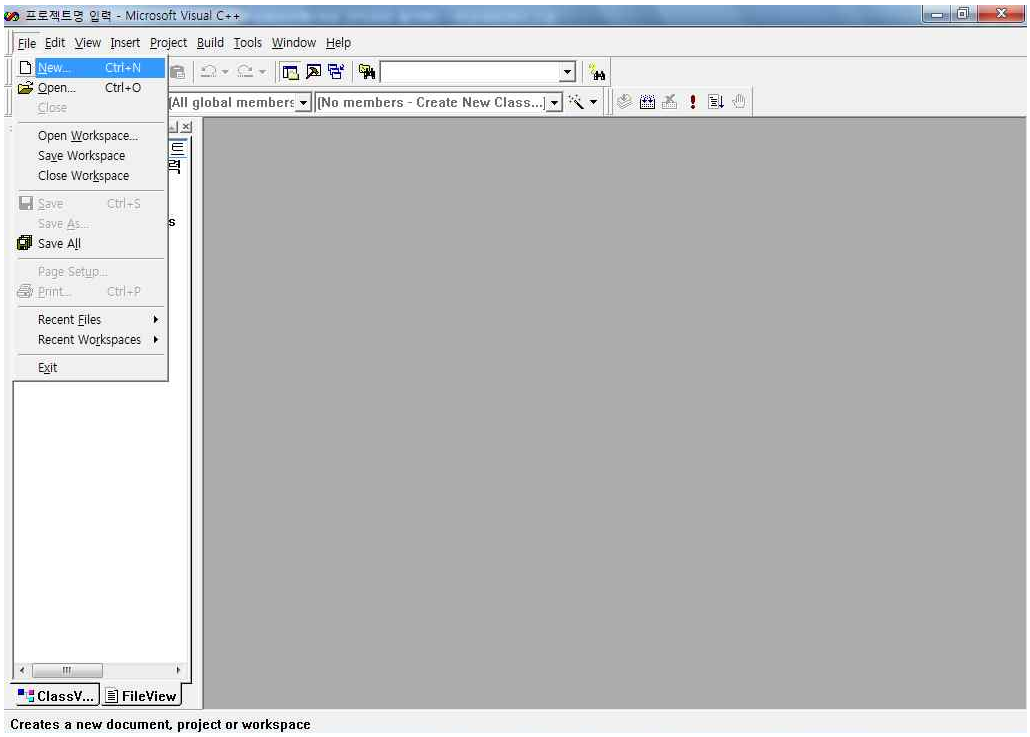
2) 새로운 프로젝트 생성 방법 3 - An empty project를 선택한다.



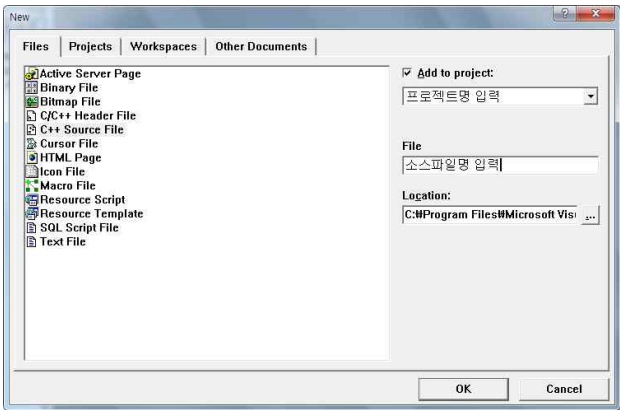
2) 새로운 프로젝트 생성 방법 4 - 화면 좌측 Work space창에 빈 Project가 생성된 것을 확인한다.



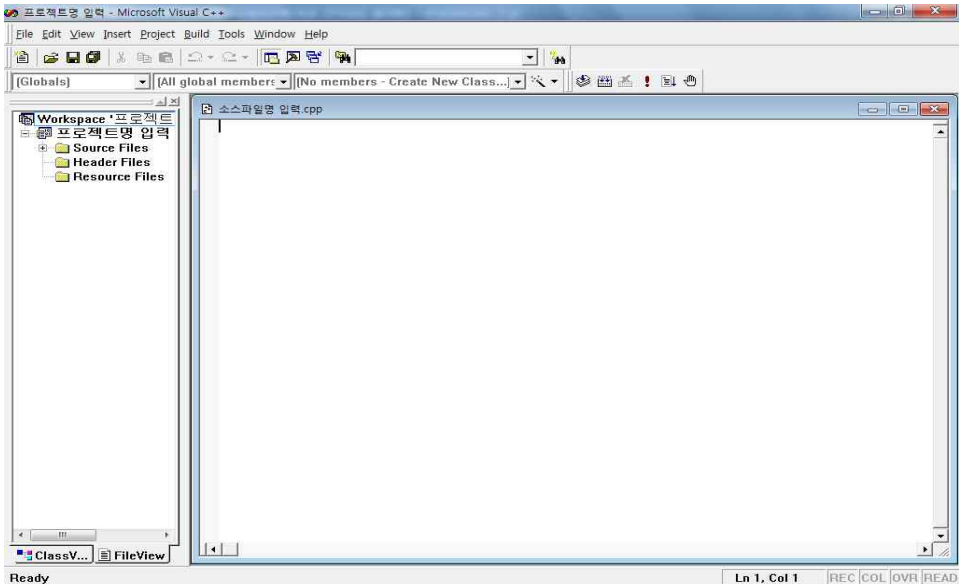
2) 새로운 프로젝트 생성 방법 5 - 화면 좌측 Work space창에 빈 Project가 생성된 것을 확인한다.



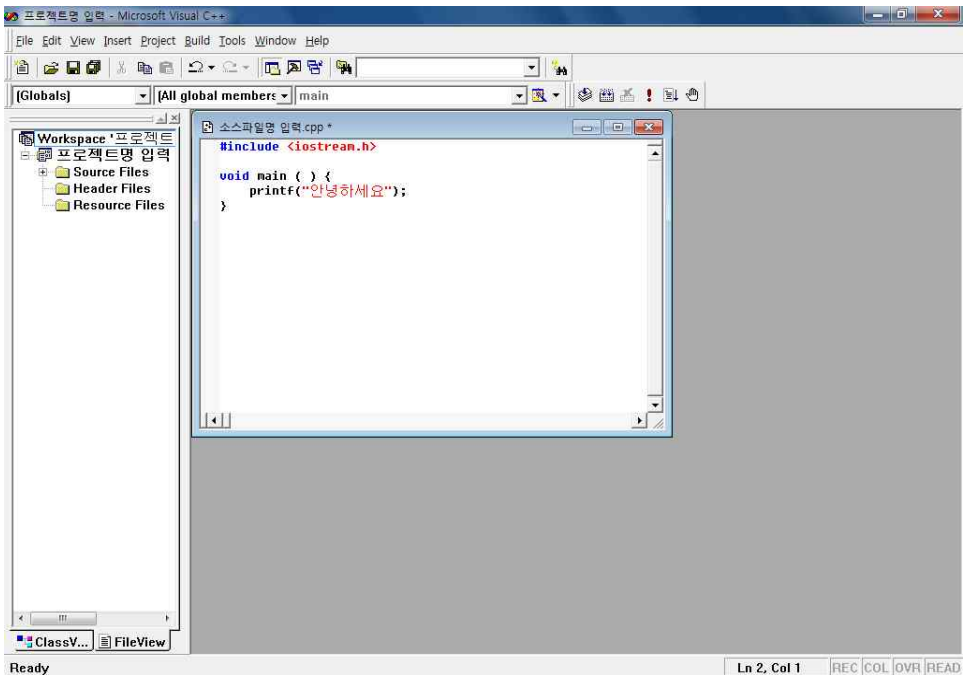
2) 새로운 프로젝트 생성 방법 6 - Files탭에서 "C++ Source File"을 선택한 뒤 File명과 Location을 선택한다.



2) 새로운 프로젝트 생성 방법 7 - 화면 중앙의 work창에 생성된 소스파일에 내용을 입력하여 프로그램을 작성한다.



3) 프로그램 작성 방법 - 소스파일 안에 원하는 코드를 작성한다.



4) 컴파일 및 실행 방법 1 - 메뉴를 이용한 컴파일 및 실행 방법 : Build 메뉴에서 Compile -> Build -> Execute를 차례로 선택해 실행 한다.

4) 컴파일 및 실행 방법 2 - 단축 아이콘 바에서 Compile -> Build -> Execute를 차례로 선택해 실행 한다.

 :단축아이콘 바의 모양

 : Compile 버튼  : Build버튼  : Execute 버튼

 : Go 버튼  : Insert/Remove Breakpoint

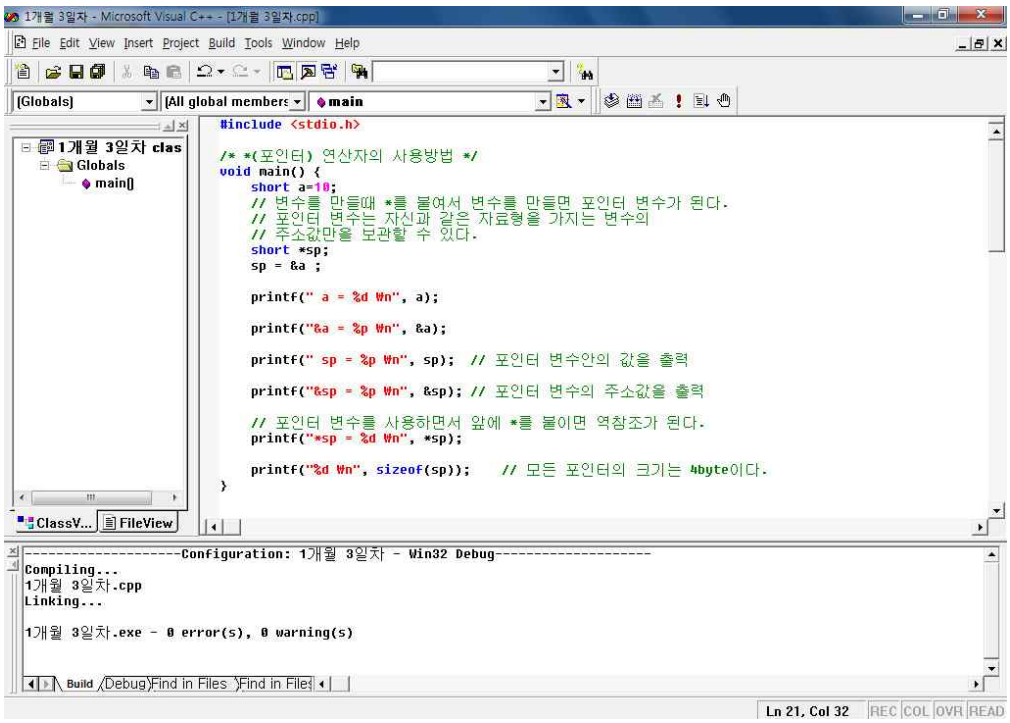
4) 컴파일 및 실행 방법 3 - Compile (Ctrl + F7) -> Build (F7) -> Execute (Ctrl + F5) 를 위한 단축키를 순서대로 누른다.

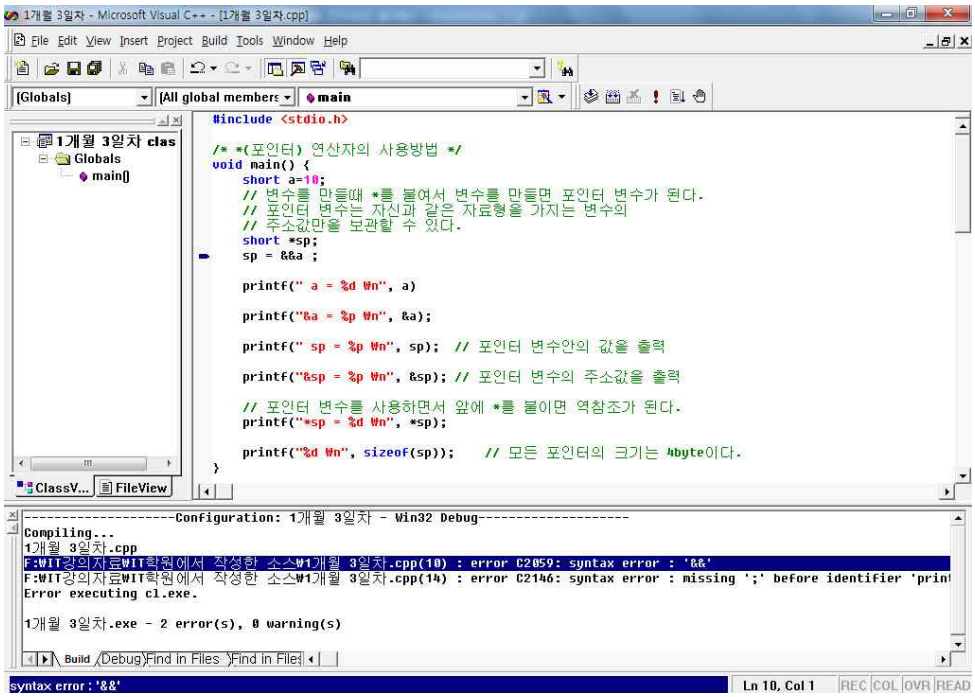
단축키 F5 : Compile, Build, Execute가 한 번에 수행되지만 실행된 프로그램이 즉시 종료된다.

단축키 Ctrl + F5 : Compile, Build, Execute가 한 번에 수행되지만 실행된 프로그램이 키를 누를 때까지 종료되지 않는다.

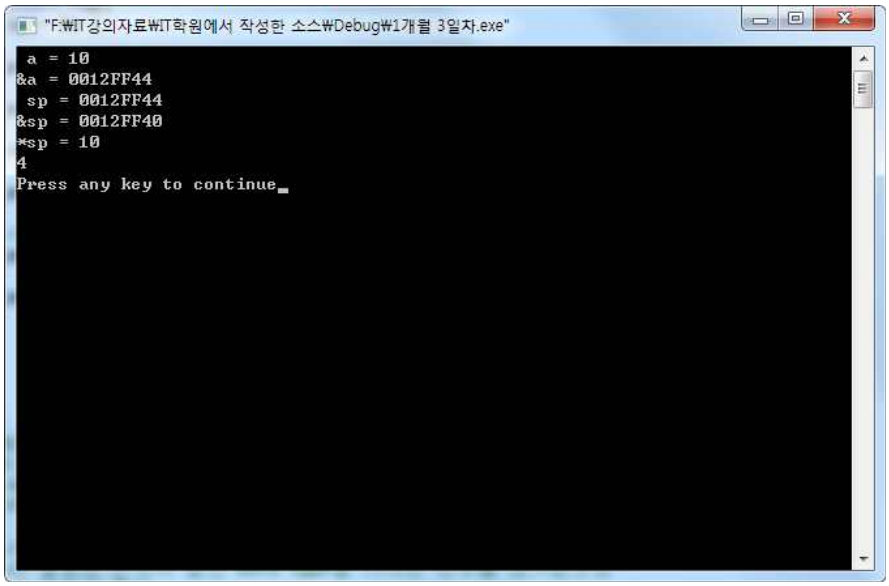
5) 에러 확인 방법 - 화면 하단의 Output 창에 나오는 error 와 warning을 확인한다.

error 와 warning이 발생한 곳의 위치는 메시지를 더블클릭하거나 단축키 F4를 이용해 확인한다.

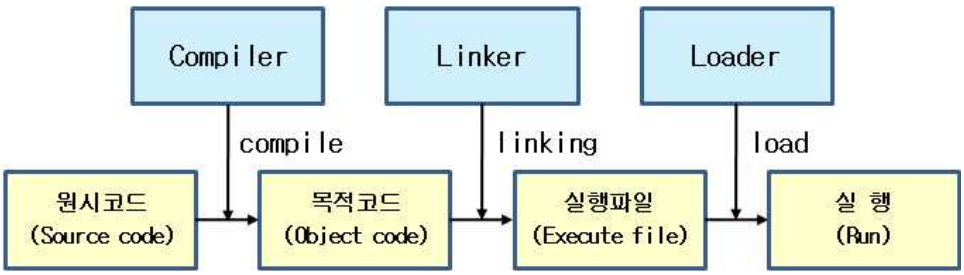




6) 프로그램이 실행된 모양



3. Compile 및 실행과정



Compile : 사용자가 만든 소스코드를 컴퓨터가 이해하고 실행 할 수 있는 목적코드(object code)로 변환하는 작업이다. 소스코드가 기계어나 어셈블러로 변환된다.

Linking : 생성된 object code를 실행시키기 위해 부가적인 파일들을 이어주는 연결고리가 생성되는 것을 말한다.

Execute : 생성된 실행파일을 메모리에 로드에 실행 하는 것을 뜻한다.

♲ 프로그램 작성 방법

1. 프로그램의 개발순서

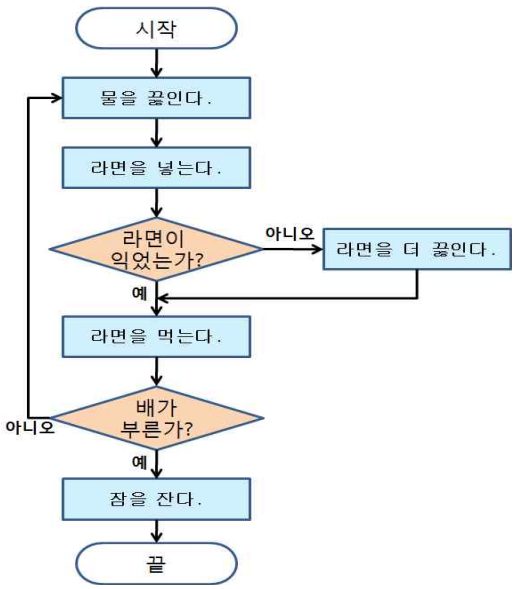
1) 요구사항들을 분석 한다.
2) 만들고자 하는 프로그램의 목적, 개요 등을 정한다.
3) 위의 1)과 2)에서 정해진 목적 및 요구 사항들을 반영해 프로그램의 알고리즘을 결정한다.
4) 프로그램을 구현(코딩) 한다.
5) 테스트 및 디버깅 작업을 한다.
6) 프로그램을 유지보수 한다.

2. 알고리즘

1) 알고리즘 : 어떤 문제를 해결하는 데 있어서 컴퓨터가 행하는 처리 방법과 관련된 일련의 논리적 절차(어떠한 문제의 해결방법을 결정하는 것)를 말함. 즉, 문제를 해결하는 방법 또는 문제를 해결해 가는 과정을 말함. 크게 상향식 설계(작은 단위의 문제를 먼저 결정하고 이를 이용해 큰 단계의 문제를 해결하는 방법)와 하향식 설계(큰 방향을 먼저 정한 뒤 세부방법을 결정해 문제를 해결하는 방법) 그리고 객체지향 설계(작은 단위의 문제 해결을 위한 자료와 처리방법을 묶어서 객체라는 부품을 만들고 이러한 부품을 연결해 문제를 해결하는 방법) 방법이 있다.

2) 알고리즘의 조건
조건1 : 외부에서 0개 이상 입력을 받아 1개 이상 출력을 생성한다.(입력, 출력)
조건2 : 각 단계가 단순해야 하며, 모호하지 않아야 한다.(명확성)
조건3 : 한정된 수의 작업 후에는 반드시 끝나야 한다.(유한성)
조건4 : 모든 명령이 수행 가능해야 한다.(효과성)
조건5 : 알고리즘이 효율적이어야 한다.

3) 알고리즘의 표현방법 1 - 순서도 (Flowchart)
프로그램이 실행되는 순서를 그림으로 표현한 것을 순서도(Flowchart)라 한다. 처리하고자 하는 문제를 분석하고 입·출력 설계를 한 후에, 그 처리 순서와 방법 (처리과정 = 프로그램 논리의 흐름)을 결정한 뒤 일정한 기호를 사용하여 일목요연하게 나타낸 그림이다.
순서도의 예 :



정 의 : 프로그램 논리의 흐름, 즉 처리과정을 다음 그림과 같은 기호를 사용하여 표현한 것.
장 점 : ① **처리 내용에 따라 정해진 기호를 사용하므로 전체의 내용을 쉽게 파악할 수 있다.**
② 프로그램 작성 시 프로그램의 개발에 따른 시행착오를 줄일 수 있다.
③ 완성된 프로그램의 오류 수정에 많은 도움을 준다.

순서도(Flowchart) 기호의 종류 : 순서도를 그릴 때 사용하는 기호는 약속된 것을 사용한다.

기호	명칭	의 미
	단자	프로그램의 시작과 끝을 표시
	처리	계산 등 다양한 자료의 처리를 표시
	자료입력	자료의 입력을 표시한다.
	비교/판단	조건의 충족 여부를 비교하여 판단
	출력	처리된 내용을 출력함을 표시함
	준비	프로그램 시작 시 초기값을 설정
	흐름선	프로그램의 진행방향을 표시
	연결	흐름선을 연결할 때 사용

순서도(Flowchart) 작성 방법 : 일반적으로 순서도는 다음과 같은 방법으로 작성한다.

① 국제 표준화 기구에서 정한 표준 기호를 사용한다.
② 논리적인 흐름의 방향은 위에서 아래로, 왼쪽에서 오른쪽으로 서로 교차되지 않도록 한다.
③ 간단하고 명료하게 작성한다.
④ 처음에는 큰 줄거리만 나열하고, 점차 구체적으로 작성한다. (개략적 설계 ⇒ 상세 설계로 작성)
⑤ 논리적은 흐름이 복잡하고 어려울 때에는 여러 단계로 구분하여 작성한다.
⑥ 순서도 기호 내부에 처리할 내용을 간단히 입력한다.

순서도(Flowchart)의 역할

① 프로그램을 작성하는 기초 자료가 된다.
② 프로그램의 정확성 여부와 오류를 쉽게 판단할 수 있어서 수정이 쉽다.
③ 업무의 흐름을 그림으로 직접 볼 수 있어서 프로그램의 논리적인 절차 및 처리 내용을 쉽게 파악할 수 있다.
④ 일정한 기간이 지난 후에도 프로그램을 이해하는데 도움을 준다.
⑤ 프로그램의 인수, 인계가 쉽다.
⑥ 프로그램의 갱신 및 유지 관리가 쉽다.

순서도(Flowchart)의 기본형태

문제해결을 위한 순서의 흐름 유형은 크게 직선형, 분기형, 반복형으로 구분 문제해결을 위하여 직선형, 분기형, 반복형 중의 세 가지 기본 형태가 조합되어 전체적인 프로그램 순서도로 작성된다.

① 직선형(순차구조) : 조건에 의해서 분기되거나 또는 일정한 내용을 반복 처리함 없이, 위에서 아래로 하나의 명령문씩 단계적으로 실행하여 실행 정지 명령에 도달되는 구조
② 분기형(선택구조) : 주어진 조건의 만족 여부에 따라 실행 내용이나 순서를 서로 달리하고자 할 때 작성되는 순서도 구조
③ 반복형(반복구조) : 특정 조건이 만족되는 동안 어느 부분의 처리 내용을 반복 실행하도록 그려진 구조. 반복 조건을 종료하는 조치가 없으면 무한 반복을 하게 된다.

3) 알고리즘의 표현방법 2 - 자연어를 이용한 서술적 표현

프로그램의 동작 모양을 말로서 표현하는 방법이다.

예) 물을 끓인다.
라면을 넣고 라면이 익을 때 까지 끓인다.
라면을 먹는다.
아직 배가 고프다면 다시 물을 끓이는 과정부터 반복한다.
배가 부르다면 잠을 잔다.

3) 알고리즘의 표현방법 3 - 프로그램을 이용한 구체화 방법
알고리즘을 프로그래밍 언어로 작성해 표현하는 방법이다. 즉시 프로그램으로 구현되어 별도의 코딩이 필요 없지만 해당 프로그래밍 언어를 모르는 사용자는 이해하기가 어렵고 다른 언어로 변환 작업이 어렵다.

3) 알고리즘의 표현방법 4 - 의사코드(수도코드)를 이용한 작성
의사 코드(수도코드)란 프로그래밍언어를 이용한 알고리즘 표현 방법과 자연어를 이용한 표현방법을 섞은 형태의 알고리즘 표현 방법이다. 특정한 언어로 작성된 것이 아님으로 즉시 사용할 수는 없지만 각종 언어로 쉽게 변환 할 수 있으며 프로그래밍 언어에 대한 지식이 많지 않은 사용자도 이해하기가 쉽다는 장점이 있다.
예)

```
hungry ( ) {
    물을 끓이고 라면을 넣는다.
    if(라면이 익었다)
        라면을 먹는다.
    else
        더 끓인 이후 라면을 먹는다.
    잠을 잔다.
}
```

3. 프로그래밍 작업 시 유의사항

1) 프로그래머가 만든 프로그램을 동작 시키는 건 컴퓨터다 따라서 모든 알고리즘 작성 시 컴퓨터의 동작에 맞추어 프로그램을 작성해야 한다.
2) 컴퓨터는 바보다! 받은 명령을 그대로 수행할 뿐이다. 따라서 프로그램이 오동작 한다면 명령을 잘못내린 프로그래머의 잘못이다.
3) 허락하는 최대한으로 주석을 달아둔다. 내가 만든 프로그램도 시간이 지나면 이해하지 못 할 수 있다.
4) 남들이 알아보기 어려운 문법을 사용해 프로그램을 작성하는 건 바보이다. (가장 좋은 코드는 보기 좋고 이해하기 쉬운 코드이다.)
5) 만들기 쉬운 프로그램 보다 고치기 쉬운 프로그램이 좋은 프로그램이다.

3. C언어에서의 프로그램 작성 방법

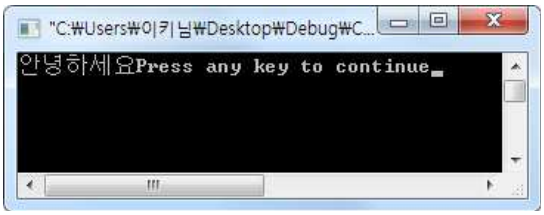
```
#include <stdio.h>                  // 전처리부,프로그램의 헤더파일과 매크로 등을 지정한다.
/* 예제프로그램 */                  // 주석
int x;                              // 전역변수

void func( ) ;                      // 함수 선언부

void main( ) {                      // 프로그램의 본체인 main함수, 프로그램의 시작
    short a=10;                     // 변수선언, 모든 변수는 함수의 본체의 앞에 정의되어야 한다.
    printf(" a = %d ₩n", a);        // 프로그램의 본체, 명령문 등을 입력한다.
    func( );
}                                   // 프로그램의 끝

void func( ) {                      // 함수의 정의부
    printf("난 func함수다!₩n");
}
```

1) C언어의 명령문은 함수의 시작 후 나오는 { 와 } 사이에 기재 한다.
2) 하나의 명령문이 완료되면 반드시 세미콜론(;)을 기재 해야 한다.

4. 간단한 예제 프로그램

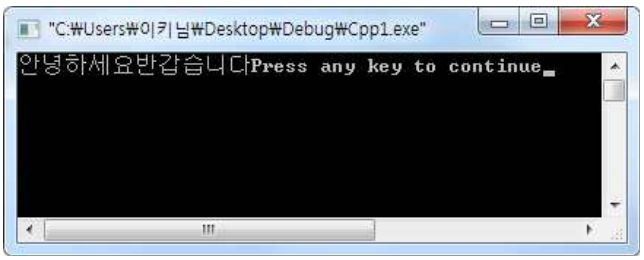
예제 1-1 : 화면에 원하는 문장을 출력하기

```
#include <stdio.h>
void main( ) {
    printf("안녕하세요");
}
```



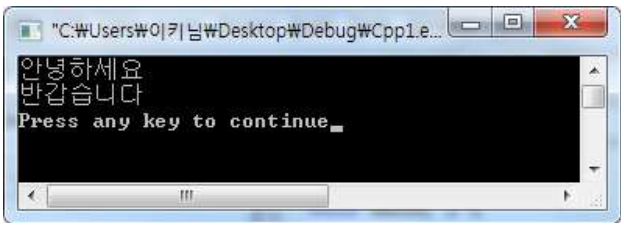
예제 1-2 : 화면에 여러개의 문장을 출력하기

```
#include <stdio.h>
void main( ) {
    printf("안녕하세요");
    printf("반갑습니다");
}
```



예제 1-3 : '₩n' 문자를 출력해 줄을 바꾸는 방법

```
#include <stdio.h>
void main( ) {
    printf("안녕하세요₩n");
    printf("반갑습니다₩n");
}
```



♲ 자료형 (data type)

자료형이란 data의 형태와 특성을 말한다. data를 보관하기 위해서는 해당 data의 모양을 알아야 하며 data를 연산하기 위해서는 해당 data의 특성을 알아야 한다. 이러한 data의 모양과 특성을 자료형(data type)이라 한다.

기본 data type의 형태

data type명		크기(byte)	보관하는 data의 종류	보관가능한 자료의 범위
char	char	1	문 자	-128 ~ 127 (총 256가지)
	signed char			-128 ~ 127 (총 256가지)
	unsigned char			0 ~ 256 (총 256가지)
short	short	2	정 수	-32768 ~ 32767 (총 65536가지)
	unsigned short			0 ~ 65535 (총 65536가지)
int	int	2 또는 4		-2147438648 ~ 2147438647 (총 4294967296가지)
	unsigned int			0 ~ 4294967295 (총 4294967296가지)
long	long	4		-2147438648 ~ 2147438647 (총 4294967296가지)
	unsigned long			0 ~ 4294967295 (총 4294967296가지)
float	float	4	실 수	$\pm 3.4*10^{-38}$ ~ $\pm 3.4*10^{38}$
double	double	8		$\pm 1.7*10^{-308}$ ~ $\pm 1.7*10^{308}$
	long double	10		$\pm 3.4*10^{-4932}$ ~ $\pm 1.1*10^{4932}$

♲ 변수 (variable)

변수란 프로그래머가 프로그램에서 수정될 수 있는 값을 저장하기 위해 사용하는 특정 메모리 위치의 이름을 말한다. 쉽게 말하자면 어떠한 값을 담아둘 수 있도록 해주는 공간이라고 생각하면 된다. 프로그램에서는 모든 데이터를 메모리에 보관함으로 이 변수라 불리는 공간도 메모리에 만들어 지게 된다. 또한 메모리에 만든 이 변수라는 공간을 프로그래머가 원하는 시점에 편하게 불러올 수 있게 하려면 이름을 부여해 주어야 하고 이 이름을 변수명이라 한다. 변수명은 아래의 규칙을 지키는 내에서 프로그래머가 마음대로 지어 줄 수 있다.

1. 변수명을 만들기 위한 규칙

1) 변수명은 반드시 알파벳과 숫자를 이용해서 지어야 한다. (특수문자나 한글을 사용해서 변수명을 지을 수 없다.)
   단, _ (언더바) 는 예외로 변수명에 포함 시킬 수 있다.
2) 변수명의 첫 글자는 반드시 알파벳 또는 _(언더바) 이어야 한다.
3) C언어는 대소문자를 구분함으로 변수명 역시 대소문자를 구분한다. (ex. mAx와 Max는 서로 다른 변수이다.)
4) C언어에서 기본적으로 사용하기 있는 키워드(예약어) 들은 변수명으로 사용 할 수 없다.

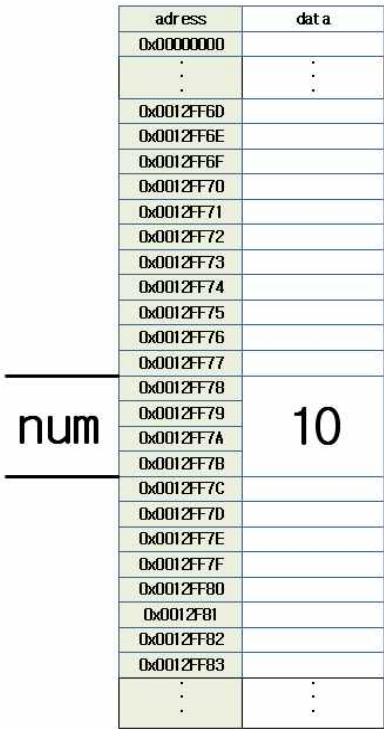
2. 변수를 만드는 방법 : 변수는 반드시 자료형과 같이 기술해 만들어야 한다.

문자를 보관할 수 있는 변수 - char mun;
정수를 보관할 수 있는 변수 - int a; short num; unsigned int jumsu;
실수를 보관할 수 있는 변수 - float sil; double PI;
주소값을 보관 할 수 있는 변수 - int *ip;

변수는 만듬과 동시에 값을 부여 할 수 있다. - int a = 10;

3. 변수가 만들어 지는 모양 (정수를 보관하기 위해 만든 int형 변수 num의 모양)

int num = 10; 을 실행했을 때의 모양



♲ 상수

고유한 크기를 가지며 더 이상 값이 변하지 않는 값이 상수이다. int num = 10; 이라는 문장에서 10이 상수가 된다. 상수 역시 메모리에 보관되며 그 값을 사용 할 수 는 있지만 변경할 수는 없다.

정수형 상수

정수형 상수는 메모리에 2진수로 변환되어 저장된다. 출력 시 %d (10진 정수형) 등의 서식을 이용해 출력 할 수 있다.

예) short num = 10; 일 때 num의 모양

16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

실수형 상수

실수형 상수는 메모리에 부동 소수점 방식으로 변환되어 저장 된다. 출력 시 %f (float형)나 %lf (long float 형) 등의 서식을 이용해 출력 할 수 있다.

예) float형 변수가 메모리에 보관되는 형태

부호	지수부	소수부(가수부)
1bit	7bit	24bit

예) double형 변수가 메모리에 보관되는 형태

부호	지수부	소수부(가수부)
1bit	7bit	56bit

문자형 상수

문자형 상수는 메모리에 2진수로 변환되어 저장된다. 출력 시 %c (문자형) 등의 서식을 이용해 출력 할 수 있으며 만약 %d로 출력하게 되면 이진수로 변환된 수가 정수로 출력된다.

문자가 보관되는 모양

아스키 코드표

상위 / 하위	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	END	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	₩(\)	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

♲ 변수와 상수의 사용방법

정수형 변수와 상수

정수값을 표현할 때는 정수형 상수를 사용하고 그 값을 보관시킬 공간을 만들고자 할 때는 정수형 변수를 만들어 값을 보관시켜야 한다. 정수는 short, int, long 등의 자료형을 사용해 보관한다.

정수는 printf()함수를 이용해 출력할 경우 일반적으로 %d (10진 정수형) 서식을 이용해 출력하며 %u(부호없는 정수형), %o(8진 정수형), %x(16진 정수형) 등의 서식으로 출력하는 것도 가능하다.

예제 1-4 : 정수형 변수를 만들어 정수형 상수 값을 보관시키는 방법 - 1

```
#include <stdio.h>

void main( ) {
    int a;                  // int의 모양을 가지는 변수를 만드는 방법

    a = 10;                 // int형 변수 a에 int형 상수 10을 보관하는 방법

    printf("%d", a);        // int형 변수에 보관되어 있는 값을 printf함수를 이용해 출력하는 방법

    printf("₩n");

}
```

예제 1-5 : 정수형 변수를 만들어 정수형 상수 값을 보관시키는 방법 - 2

```
#include <stdio.h>

void main( ) {
    long a;                  // long의 모양을 가지는 변수를 만드는 방법

    a = 10l;                 // long형 변수 a에 long형 상수 10을 보관하는 방법

    printf("%d", a);         // long형 변수에 보관되어 있는 값을 printf함수를 이용해 출력하는 방법
    printf("₩n");
    printf("%d", 55);        // 55가 출력됨. 정수형 상수를 바로 출력 할 수 있다.
    printf("₩n");

}
```

예제 1-6 : 변수를 생성함과 동시에 초기화 하는 방법

```
#include <stdio.h>

void main( ) {
    int a = 10;              // a가 생성됨과 동시에 10으로 초기화 된다.

    printf("%d", a);         // 10이 출력됨

    printf("₩n");

}
```

예제 1-7 : 정수형 변수의 값을 변경하는 예제

```
#include <stdio.h>

void main( ) {
    int a = 10;              // a가 생성됨과 동시에 10으로 초기화 된다.

    printf("%d", a);         // 10이 출력됨
    printf("₩n");

    a = 20;                  // 20을 변수 a에 보관시킨다.

    printf("%d", a);         // 20이 출력됨
    printf("₩n");

}
```

예제 1-8 : printf( ) 함수를 이용해 정수와 문자열을 함께 출력하는 방법

```
#include <stdio.h>

void main( ) {
    int a = 10;              // a가 생성됨과 동시에 10으로 초기화 된다.

    printf("a = %d₩n", a);   // a = 10이 출력되고 줄이 바뀜

}
```

예제 1-9 : 정수형 변수의 값을 수식의 결과로 변경하는 방법

```
#include <stdio.h>

void main( ) {
    int a = 10;              // a가 생성됨과 동시에 10으로 초기화 된다.

    a = a * 2;               // a에 2를 곱해 a에 넣는다. 즉, a가 20이된다.

    printf("a = %d₩n", a);   // a = 20이 출력됨

    printf("%d ₩n", a*a);    // a*a의 결과인 400이 출력됨.

}
```