

- ▶ 주석, main 함수, printf 함수
- ▶ 변수의 원리
- ▶ 상수



Artificial Intelligence Laboratory

주석, main 함수, printf 함수

주석(Comment)

한 줄 주석

- **//**부터 해당 줄의 끝까지를 주석으로 간주

```
printf("Hello World"); // 여기서부터 이 줄 끝까지 주석
// 한 줄 전체를 주석으로 만든다.
```

한 줄 주석
한 줄 주석

여러 줄 주석

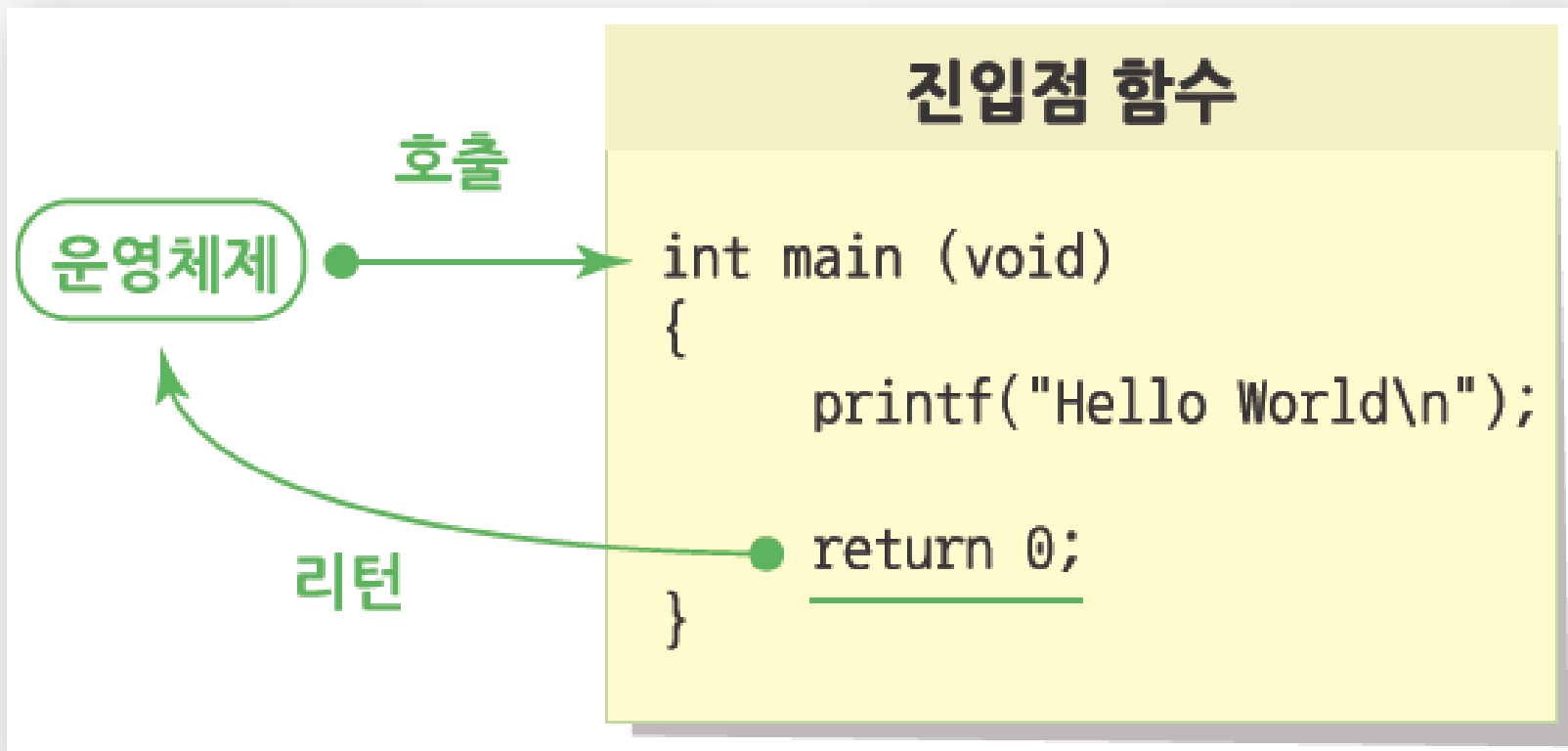
- **/***부터 ***/**까지를 주석으로 간주

```
printf(/* 출력할 내용 */ "Hello World");
/* 여러 줄로 된
   주석도 만들 수 있다. */
```

문장 중간에도 주석 사용 가능
여러 줄 주석

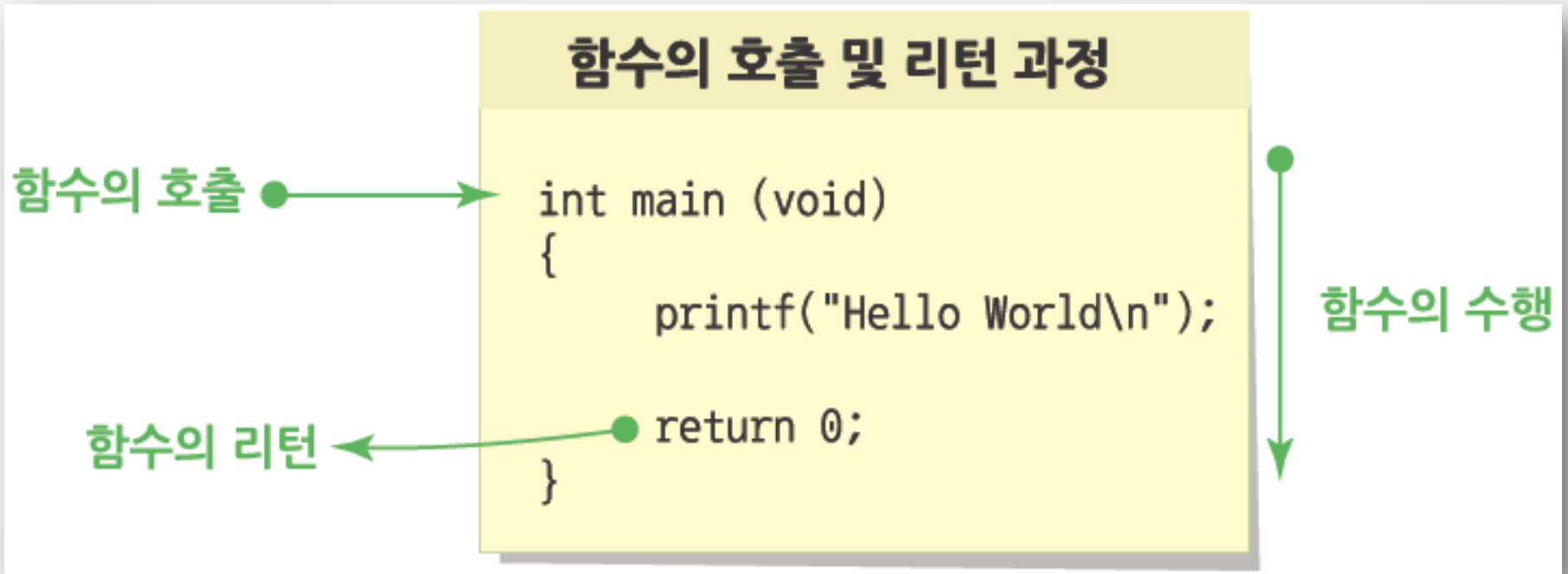
Main 함수

- 프로그램이 처음 시작될 때 호출되는 함수
 - main 함수는 프로그램의 시작과 끝, 알파와 오메가.
 - 진입점 함수(entry-point function)이라고도 함.
- main 함수는 프로그램에 단 하나만 존재해야 함



함수의 원칙

- 함수 안에 있는 문장들이 순차적으로 수행됨
- 함수의 끝에 도달하거나 return을 만나면, 함수를 호출한 곳으로 되돌아감
 - main 함수는 프로그램이 종료된다.



printf 함수

문자열을 출력하기 위한 함수

줄 바꿈 문자를 쓰지 않은 경우

```
printf("Hello World");  
printf("Good Bye");
```



다음에 출력할 위치

줄 바꿈 문자를 쓰는 경우

```
printf("Hello World\n");  
printf("Good Bye\n");
```



다음에 출력할 위치

형식문자열의 종류

| %d

- ▶ 정수(decimal number)
- ▶ 예) `printf("%d", 10);`

| %f

- ▶ 실수(float number)
- ▶ 예) `printf("%f", 10.0);`

| %c

- ▶ 문자(character)
- ▶ 예) `printf("%c", 'a');`

| %s

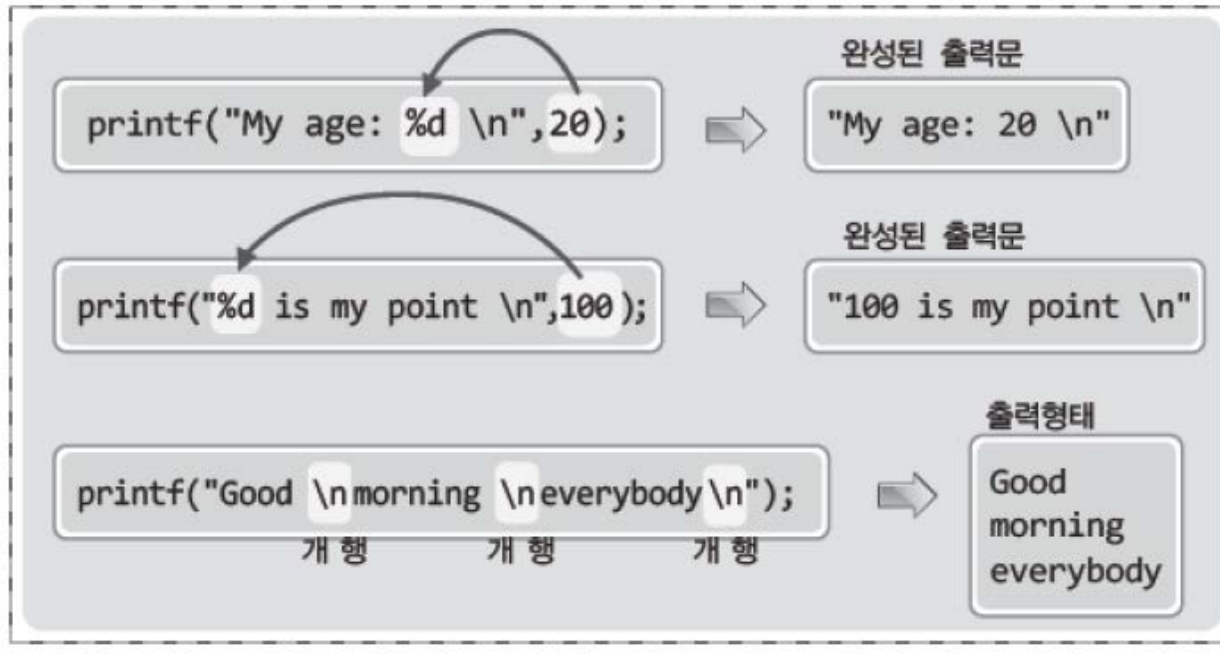
- ▶ 문자열(string)
- ▶ 예) `printf("%s", "a");`

실습해보기

```
int main(void)
{
    printf("My age: %d \n", 20);
    printf("%d is my point \n", 100);
    printf("Good \nmorning \neverybody\n");
    return 0;
}
```

실행결과

My age: 20
100 is my point
Good
morning
everybody



Artificial Intelligence Laboratory

변수의 원리

I 변수

- ▶ 값이 변경될 수 있는 데이터
- ▶ 숫자나 문자를 보관하는 역할

I 상수

- ▶ 값이 변경될 수 없는 데이터
- ▶ 숫자나 문자 자체를 의미

변수의 선언

형식

데이터형 변수명;
데이터형 변수명1, 변수명2, ... ;

예제

char	choice;
int	income2012;
double	discount;
int	amount, price;

↓
데이터형

↓
변수

변수의 선언 및 초기화

형식

데이터형 변수명 = 초기값 ;

데이터형 변수명 = 초기값, 변수명 = 초기값, ... ;

예제

char
int
double

choice
data
average

= '2' ;
= 100 ;
= 0.0 ;

데이터형

변수명

초기 값

int

amount

= 20 ,

price =

1000;

데이터형

변수명

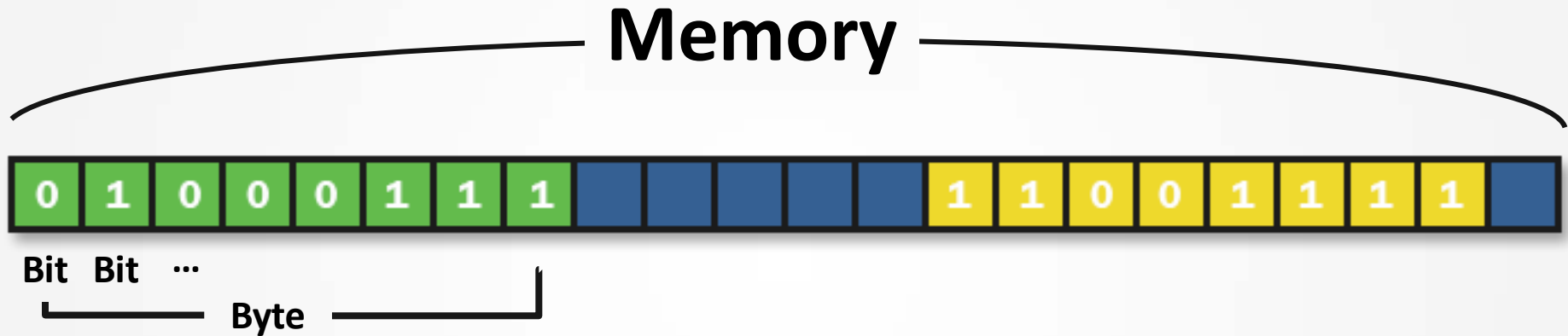
초기값

변수명

초기값

메모리의 개념적 이해

- 메모리는 일렬로 늘어선 저장 공간이다.
 - 한 칸의 저장 공간에는 0 혹은 1만 저장 가능하다.



Bit (비트)

- 컴퓨터에서 처리하는 <정보의 최소 단위>
- 1bit는 0 또는 1, 즉 이진수의 한 자리 숫자만 표현 가능.

Byte (바이트)

- 1Byte = 8Bit
- 프로그래머가 <제어 가능한 최소 단위>

메모리의 개념적 이해

- 메모리는 1바이트마다 주소가 매겨져 있다
 - 이 '주소값'은 양의 정수이다. 표기할 땐 16진수로 표기한다

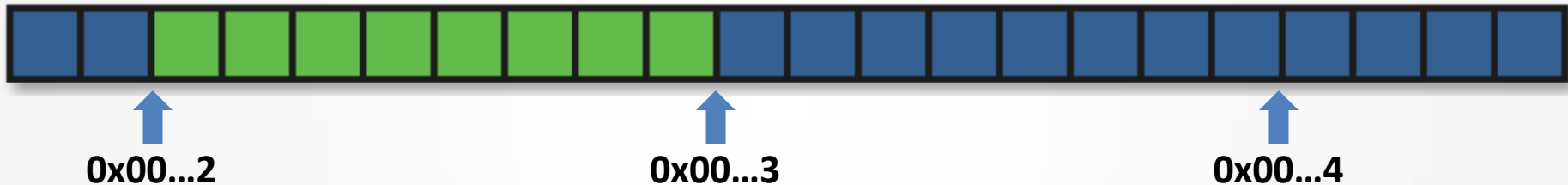


변수의 선언 및 초기화

char myChr = 'A';

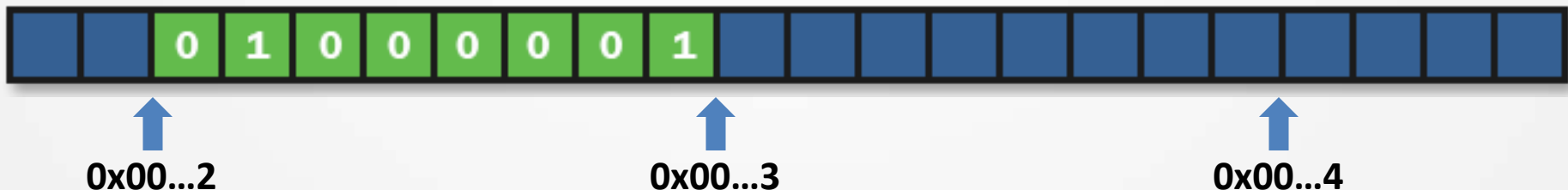
메모리에서 1바이트를 확보한다

- 이를 '메모리를 할당' 한다고 표현한다.
- 할당되는 메모리는 데이터형에 따라 다르다. (char는 1바이트를 요구)



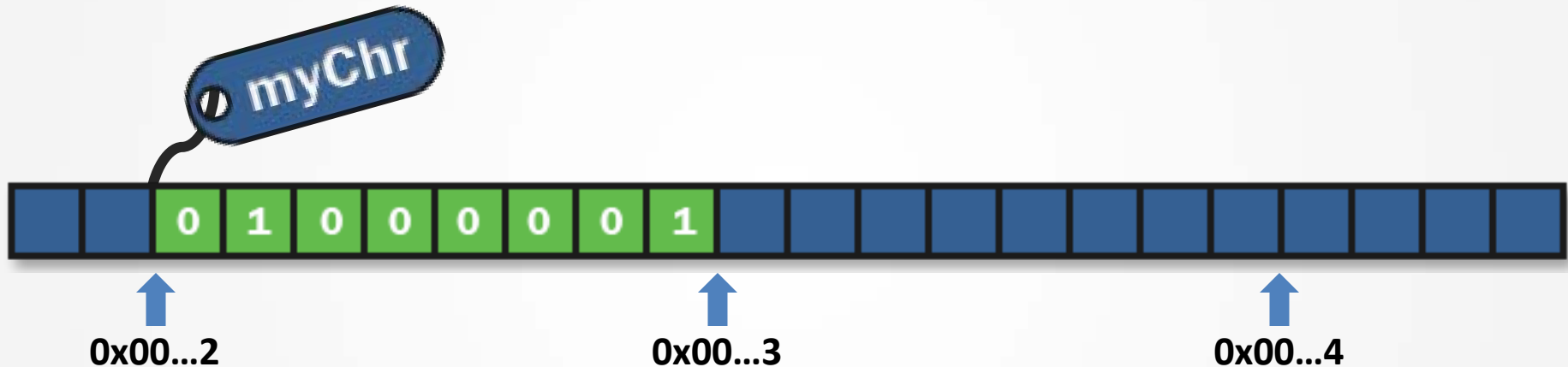
데이터를 이진수로 변환한 후 메모리에 저장한다

- 'A'는 이진수로 변환하면 '01000001'이 된다.



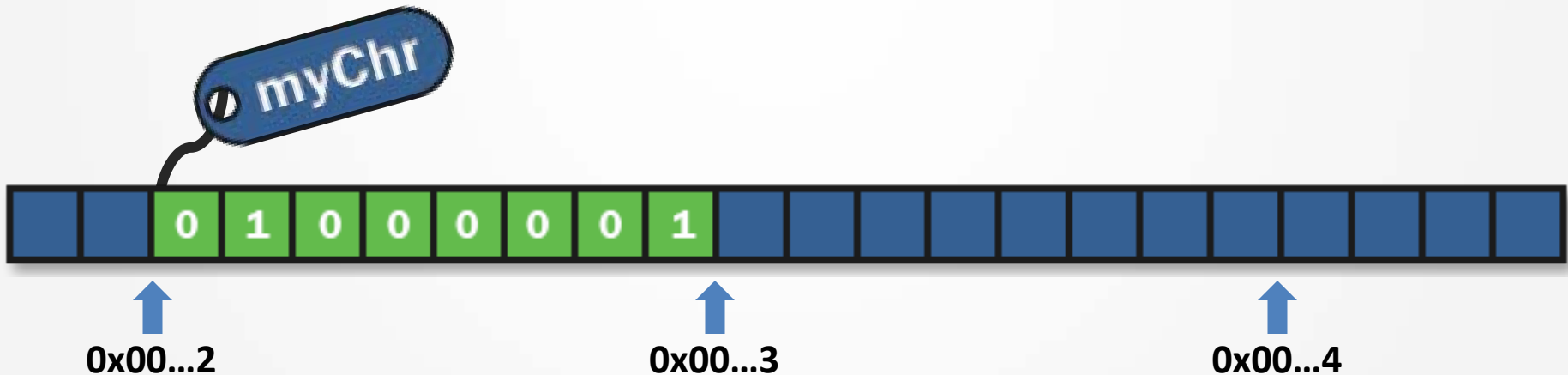
변수의 선언 및 초기화

- 해당 공간의 시작 주소를 'myChr'라고 이름 붙인다
 - 이제부터 myChr라는 이름은 0x00...2 를 의미한다.



변수의 사용

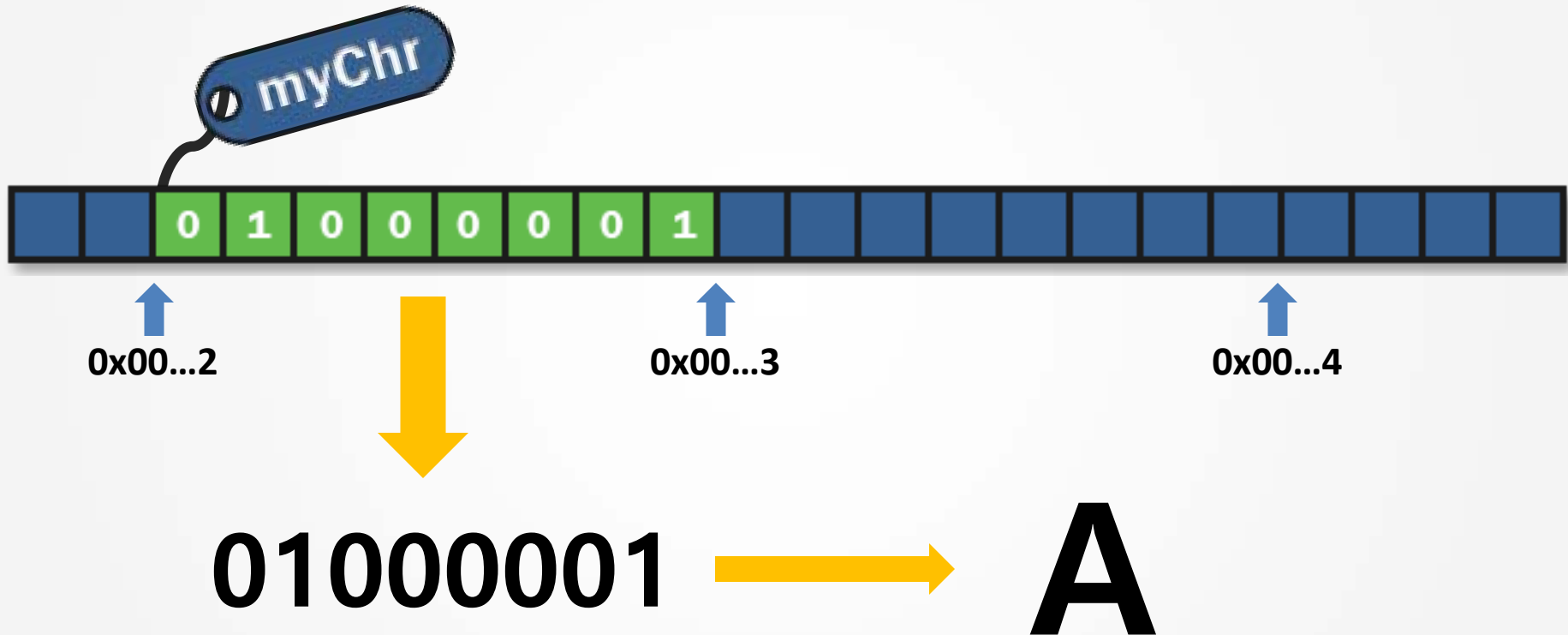
- | `printf("%c", myChr);`
- | `myChr`가 가리키는 시작 주소를 찾아 간다
- | 시작 주소로부터 변수의 자료형의 크기만큼 읽어 들인다.
 - `Char`는 1바이트의 자료형이므로 8bit 만큼을 읽어 들이는 것이다.



변수의 사용

읽어 들인 이진수 값을 형식 문자열에 맞춰 해석한다.

• '01000001'을 문자형 데이터로 해석하면 'A'가 된다.



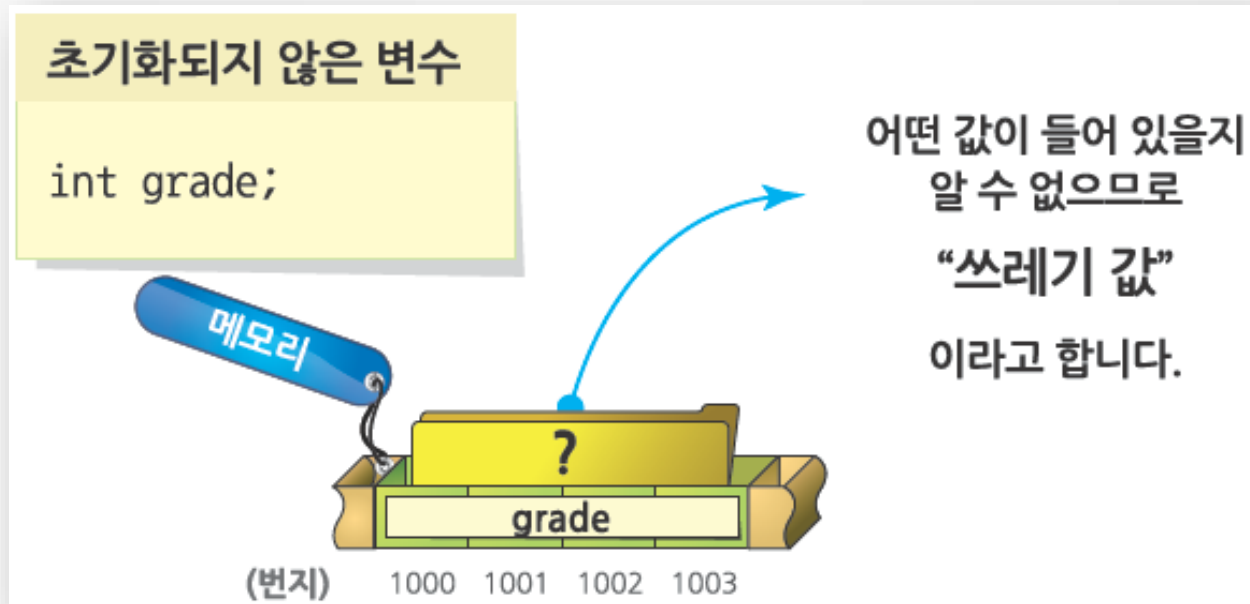
C의 대표적 데이터형

- 데이터형(type)에 따라 메모리가 얼마만큼 필요한지 달라짐

데이터 유형	데이터형	바이트 수
문자형	char	1
정수형	short	2
	int	4
	long	4
	long long	8
실수형	float	4
	double	8

변수의 초기화

- 변수를 선언할 때 따로 초기화를 하지 않으면 쓰레기 값을 가짐
- 초기화되지 않은 변수를 사용하는 것은 위험



식별자(identifier)

식별자(identifier)

- ▶ 변수, 함수, 자료형 등의 이름.
이런 유형의 '이름'들을 통틀어 '식별자'라고 지칭한다.
- ▶ 현재 단계에서는 '변수, 함수, 자료형의 이름'이라고만 이해해도 무방하다.

식별자 작성 규칙(C언어 한정)

- ▶ 반드시 영문자, 숫자, 밑줄 기호(_)만을 사용해야 한다.
- ▶ 숫자로 시작하면 오류가 난다.
- ▶ 대소문자를 구분한다. name, Name, NAME은 모두 다른 이름으로 간주된다.
- ▶ C 언어의 키워드(keyword)는 사용할 수 없다.

auto	break	case	char	const
continue	default	do	double	else
enum	extern	float	for	goto
if	int	long	register	return
short	signed	sizeof	static	struct
switch	typedef	union	unsigned	void
volatile	while			

변수의 선언 예

올바른 변수 선언의 예

`int income2012;` ○ 변수명의 첫 글자 외에는 숫자를 사용할 수 있다.
`double _pi;` ○ `_`로 시작하는 변수명은 유효하다.
`int amount, price;` ○ 여러 개의 변수를 선언할 때는 `,`를 사용한다.
`long text_color;` ○ 여러 단어를 연결할 때는 `_`를 사용한다.
`int totalAmount;` ○ 연결되는 단어의 첫 글자를 대문자로 지정한다.

잘못된 변수 선언의 예

`long text-color;` ○ 변수명에 `'-'` 기호를 사용할 수 없다.
`int total amount;` ○ 변수명에 빈칸을 포함할 수 없다.
`int 2012income;` ○ 변수명은 숫자로 시작할 수 없다.
`char case;` ○ C 키워드는 변수명으로 사용할 수 없다.

실습과제 1 – 덧셈 프로그램

아래와 같이 덧셈 프로그램을 작성하시오.

소스파일명: add.c

```
int main(void)
{
    int num1=3;
    int num2=4;
    int result=num1+num2;

    printf("덧셈 결과: %d \n", result);
    printf("%d+%d=%d \n", num1, num2, result);
    printf("%d와(과) %d의 합은 %d입니다.\n", num1, num2, result);
    return 0;
}
```

Artificial Intelligence Laboratory

상수

리터럴(Literal 상수)

I 값 자체를 직접 사용하는 것

상수형	구분	예
문자형 상수	일반 문자	'a', 'b', 'c'
	특수 문자	'\t', '\n', '\\', '\007', '\xa'
	유니코드 문자	L'a', L'b', L'c'
정수형 상수	10진수 정수	10, -10
	16진수 정수	0xabcd, 0X12EF
	8진수 정수	012, 0234
	unsigned형 정수	123u, 123U
	long형 정수	123456l, 123456L
	unsigned long형 정수	12345678ul, 12345678UL
실수형 상수	부동소수점 표기 실수	3.1425, -0.12345
	지수 표기 실수	3.5e13, 4.5E-30
	float형 실수	3.14f, 3.14F

여러가지 리터럴 상수

```
01: /* Ex03_02.c */
02: #include <stdio.h>
03:
04: int main(void)
05: {
06:     char grade = 'Wx41';
07:     int data = 0x7b;
08:     unsigned int age = 75U;
09:     long fileSize = 1234567L;
10:     double area = 123.25;
11:     double taxRate = 25e-2;
12:     float temperature = 17.5F;
13:
14:     printf("grade = %cWn", grade);
15:     printf("data = %d, %o, %xWn", data, data, data);
16:     printf("age = %uWn", age);
17:     printf("fileSize = %dWn", fileSize);
18:     printf("area = %f, %e, %gWn", area, area, area);
19:     printf("taxRate = %fWn", taxRate);
20:     printf("temperature = %fWn", temperature);
21:
22:     return 0;
23: }
```

리터럴 상수로 초기화된 변수

실행 결과

```
grade = A
data = 123, 173, 7b
age = 75
fileSize = 1234567
area = 123.250000, 1.232500e+002, 123.25
taxRate = 0.250000
temperature = 17.500000
```

실습과제 2 – 사칙연산 프로그램

■ 아래의 실행결과가 나오도록 사칙연산 프로그램을 작성하시오.

● 소스파일명: calc.c

9 + 2 = 11

9 - 2 = 7

9 × 2 = 18

9 ÷ 2의 몫 = 4

9 ÷ 2의 나머지 = 1

```
int main(void)
{
    int num1=9, num2=2;
    printf("%d+%d=%d \n", num1, num2, num1+num2);
    printf("%d-%d=%d \n", _____);
    printf("%d×%d=%d \n", _____);
    printf("%d÷%d의 몫=%d \n", _____);
    printf("%d÷%d의 나머지=%d \n", _____);
    return 0;
}
```