C언어 기초

박성기

Contents

1. 입출력, 자료형, 스트림, 연산자, 조건문(if)
2. 조건문(if), 반복문(for, while), 난수(rand), 배열
3. 함수, 포인터
4. 문자열, 구조체, 헤더파일, 파일분할, 파일 입출력

Day 01 - Contents

1. C언어 소개(가치)
- Low Level 언어 vs High Level 언어
- 절차지향 vs 객체지향
- 프로그래밍 절차

2. C언어 기본 구조
- 헤더파일
- 함수 선언
- 프로그램 작성 규칙

3. 입력, 출력 코드
- Hello World
- scanf_s, printf
- escape sequence
- 주석

Day01 - 1. C언어 소개 - Low 언어 vs High 언어

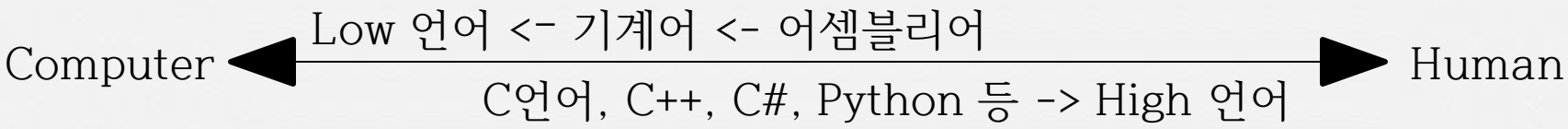
Low Level 언어 (컴퓨터 효율↑, 프로그램 가독성 ↓)

기계어

-전기회로에 의해 직접적으로 해석, 실행되는 언어 (0101010101100101)
-컴퓨터를 효율적으로 활용 할 수 있다.
-프로그래밍 시간이 오래 걸리고, 오류가 발생할 가능성이 높다

어셈블리어

-기계어와 1:1 대응시킨 기호언어
-어셈블러에 의해 실행 할 수 있는 언어
-프로그램의 수행시간이 빠르고 주기억장치를 매우 효율적으로 사용
-언어의 호환성이 매우 좋지 않음.(다른 기종의 컴퓨터에서 사용 불가)



Day01 - 1. C언어 소개 - Low 언어 vs High 언어

High Level 언어 (컴퓨터 효율↓, 프로그램 가독성↑)

C언어

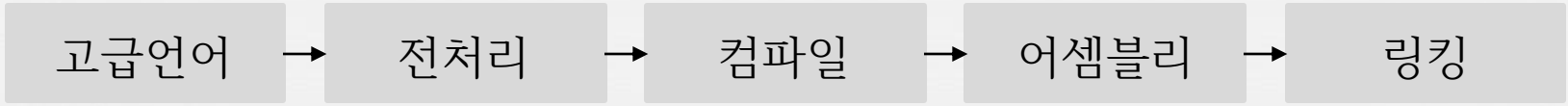
-절차지향 언어
-Unix 운영체제를 설계하기위해 고안된 언어(Linux는 Unix의 PC버전)
-운영체제, 임베디드 등 활용도 높고, 다른 고급언어의 기반이 되는 언어

C++

-C언어 확장판 객체지향 언어

이 외에도 고급언어는 매우 다양하게 있다.

컴파일-고급언어를 컴퓨터가 실행 가능하도록 변환 과정



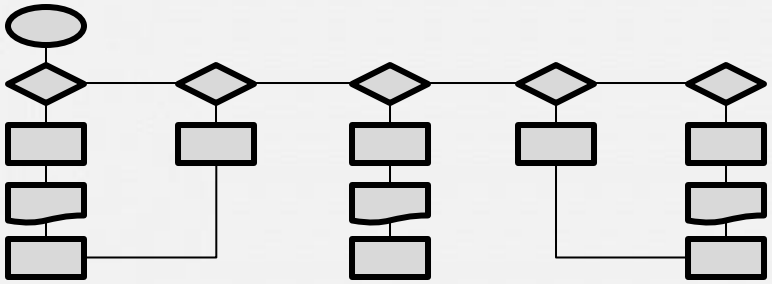
Day01 - 1. C언어 소개 - 절차지향 vs 객체지향

절차지향

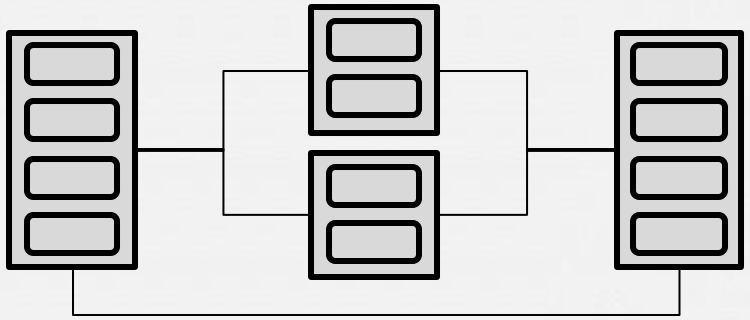
-프로그램을 코드화하고 순차적으로 명령을 실행시키는 방식
-실행속도, 효율성 ↑, 간결한 문법, 하드웨어 직접제어 가능
-실행 순서가 정해져 있어 코드 순서가 바뀌면 결과가 달라질 수 있음
-디버깅, 유지보수가 어렵다.

객체지향

-객체, 클래스, 상속의 개념을 기본으로 프로그램 구성
-객체: 데이터와 데이터에 관련된 연산으로 구성
-클래스: 객체들의 공통적인 성질을 정의
-상속: 객체의 정보를 세분화, 일반화의 원칙에 따라 조직화
-프로그램의 구조화로 코드의 재사용, 호환성, 디버깅, 유지보수가 좋다

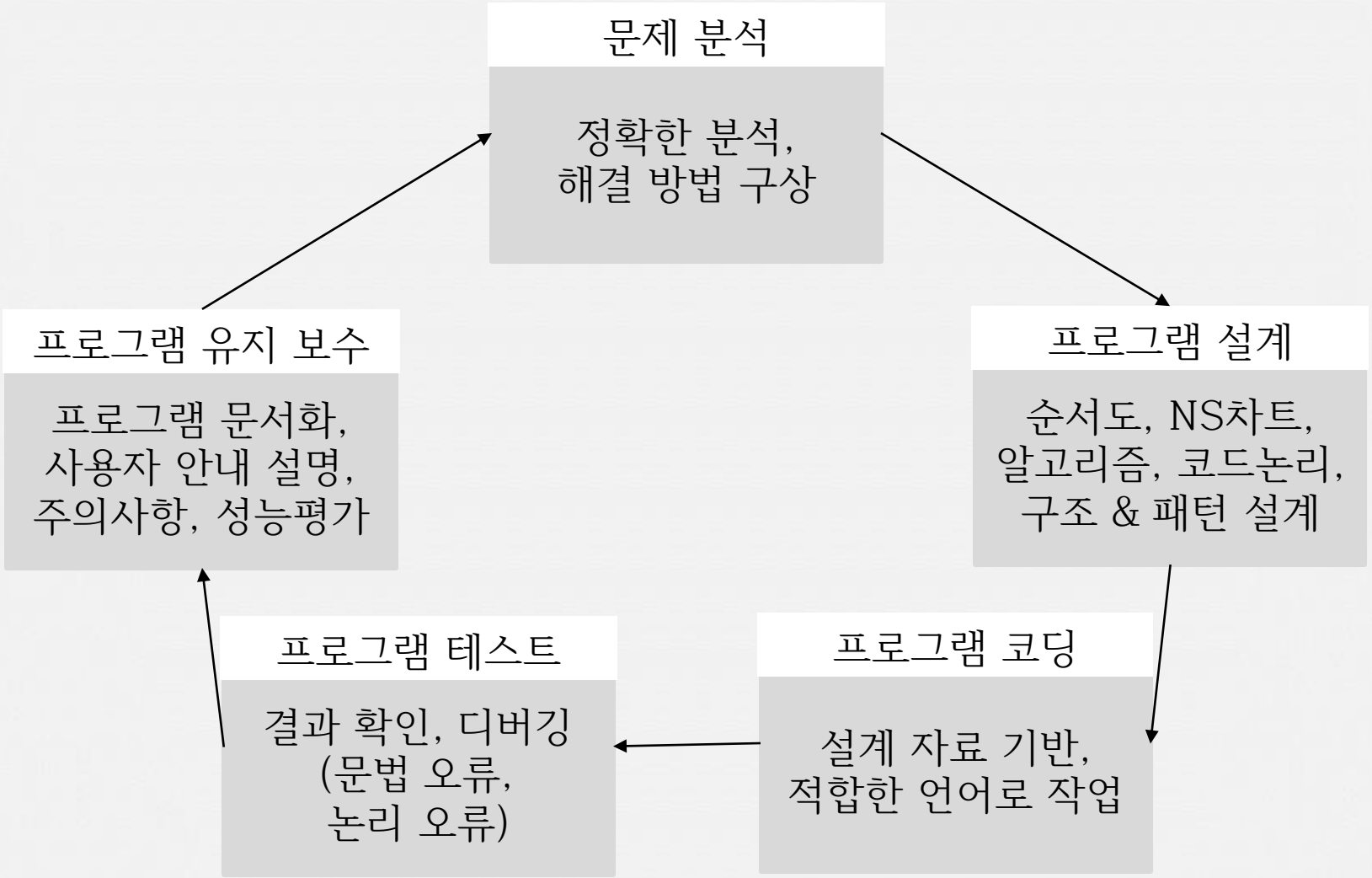


절차지향 <Flow Chart>



객체지향

Day01 - 1. C언어 소개 - 프로그래밍 절차



Day01 - 2. C언어 기본구조 - 헤더파일, 함수

헤더파일

```
#include 문          ---------------  파일처리 전처리기문 (헤더파일)
#define 문           ---------------  형태처리 전처리기문 (매크로)
```

함수

```
int main(void)       ---------------  메인함수 선언
{                    ---------------  메인함수 시작 중괄호
    명령문           ---------------  메인함수 정의(내용)
    // /*  */        ---------------  주석(한 줄 주석, 여러 줄 주석)
    printf("Text~~"); ---------------  출력 함수
    return 문;       ---------------  반환 함수
}                    ---------------  메인 함수 끝 중괄호
```

C프로그램은 반드시 1개의 main() 함수를 가져야 하며,
main() 함수부터 먼저 실행 하고 종료되면 프로그램 종료된다.
int - 리턴 타입 / main 함수 이름 / void 인수 타입(매개변수)

Day01 - 2. C언어 기본구조 - 프로그램 작성 규칙

- 프로그래밍은 공동의 작업임을 반드시 유념해 둘 것.

- C 프로그램은 영문 소문자를 기본으로 작성한다.
  식별어(identifier)를 제외한 모든 예약어, 표준 함수는 모두 소문자이다.
  예약어(reserved word) - printf, return, int, void … 등

- 식별어는 사용자가 임의대로 만들어 사용 할 수 있어 규칙을 정하라.

  #define square(x) x*x -형식
  위 예시처럼 의미를 갖고 사용 할 수 있도록 만드는 것이 좋다.

- C 프로그램은 한 문장이 끝날 때마다 반드시 세미콜론(;)을 표시한다.

- 프로그램을 작성 할 때 각 함수의 목적과 기능을 잘 설명해두자.

- 주석을 반드시 작성
  // ~ : 한 줄 모두 주석 처리
  /* ~~~ */ : /* 부터 */ 까지 주석 처리(여러 줄 주석 가능)

Day01 - 3. 입력, 출력 함수 Hello World, scanf, printf

printf 형식
(1) printf("문자열");
(2) printf(“서식 문자열”, 인수1, 인수2, …);

예시
(1) printf("Hello World!");
(2) printf(“X=%d Y=%d X+Y=%d”, 10, 20, 10+20);

scanf_s 형식
(1) scanf_s("%d",&인수1);
(2) scanf_s(“%d %d %c”, &인수1, &인수2, &…);
- (2)의 경우 Tab, Space Bar, Enter 키로 데이터 구분해야 한다.
- 배열명은 배열명 자체가 주소를 나타내므로 &를 쓰지 않아도 됨.

예시
(1) scanf_s("%d", &x);
(2)-1 scanf_s(“%d %d %c”, &x, &y, &z);
(2)-2 scanf_s(“%d %d %s”, &x, &y, (&)name, sizeof(name));

Day01 - 3. 입력, 출력 함수 Hello World, scanf, printf

```
#include <stdio.h>  // 헤더파일
int main() //프로그램 시작 함수(함수 선언)
{
    printf("Hello World!");  //함수 정의(내용)
    return 0; // 메인 함수의 반환값
}

#include <stdio.h>  // 헤더파일
int main() //프로그램 시작 함수
{
    int x;
    char name[15];
    printf("좌석번호(001~550)와 이름을 입력 하세요 :");  //서식 문자열
    scanf_s("%d %s", &x, name, sizeof(name)); //변수 x에 숫자 데이터 삽입
    printf("입력된 좌석은 %d, 이름은 %s 입니다\n", x, name);
    // 좌석번호, 이름 출력
    return 0; // 함수의 반환값
}
```

Day01 - 3. 입력, 출력 함수 - escape sequence

제어문자	기능	제어문자	기능
\n	줄 바꿈	\0	문자열의 끝을 의미
\t	수평으로 tab만큼 칸 띄우기	\f	인쇄 출력 중 page skip
\b	커서를 뒤로 한 칸 이동 후 출력	\\	\를 표시
\r	현재 라인의 처음으로 커서 이동	\’	‘를 표시
\a	‘Beep’음 출력	\”	“를 표시
		\ddd	8진수 ASCII 코드 값 부여

* \ = \ 역슬래쉬 키

Day01 - 3. 웹 컴파일 주소

Repl.it
https://repl.it/languages/c

onlinegdb
https://www.onlinegdb.com/online_c++_compiler

STEP 온라인 컴파일러
https://onlinelab.e-koreatech.ac.kr/compiler?c

Day01 - 2. 예제 코드

```
#include <stdio.h>  // stdio 헤더파일 불러들인다.
int main() //프로그램 시작 함수 main 함수 , 정수형 반환 값을 가진다.
{
  //출력 printf("문자열");
  printf("Hello World!\n"); //출력함수 "~~" ~~내용을 출력
  printf("X = %d Y = %d X + Y = %d\n", 10, 20, 10 + 20);
  //입력 scanf_s("%d",&x);
  int x, y; // int(정수형) 변수 x, y
  char z, name[15]; //char(문자형) 변수 z, name (: 15개 묶음)
  printf("X, Y, Z 입력하세요.:");
  scanf_s("%d %d %c", &x, &y, &z /*, (int)sizeof(z) */);
  // 변수 x, y 값을 정수형으로 입력 // 변수 z 는 문자형 입력
  printf("X = %d Y = %d Z = %c\n", x, y, z); // x, y, z(문자형) 출력
  printf("좌석번호(001~550)와 이름을 입력 하세요 : "); //서식 문자열
  scanf_s("%d %s", &x, /* & */name, (int)sizeof(name)); //변수 x에 정수 입력
  printf("입력된 좌석은 %d, 이름은 %s 입니다\n", x, name);
  printf("ABC");
  printf("\bD\n"); // \b
  printf("T\tA\tB\n"); // \t
  printf("123\n");
  printf("123\f456");// \f : 인쇄 Page skip
  printf("\r789\n"); // \r
  printf("\\ \' \" \n"); // \\ \' \"
  return 0; // 함수의 반환값
}
```

Day 02 - Contents

1. 자료형
- 문자 자료형
- 정수 자료형
- 실수 자료형
- 형변환, 출력변환문자

2. 상수
- 리터럴 상수 - 정수형, 실수형
- ASCII 코드
- 심볼릭 상수

3. 변수 선언
- 변수 이름을 짓는 규칙 (상수, 함수 모두 포함)

4. 예제 코드
- 성적 계산 프로그램

Day02 - 1. 자료형 - 문자 자료형

문자 자료형

예약어 char, unsigned char

8 비트 = 1 - 부호비트, 7 - 데이터비트

<unsigned 부호비트 제외한 양수범위>

□□□□□□□□

0 양수 10000000 = -128

1 음수 10000010 = -128 + 2 = -126

구분	허용범위	바이트 수	사용용도
char	-128 ~ 127	1	일반적인 영문자와 숫자 사용시
unsigned char	0 ~ 255	1	그래픽 문자나 바이트 단위 수치

* 컴퓨터의 운영체제 및 설정에 따라 자료형의 크기는 다를 수 있다

Day02 - 2. 상수 - ASCII코드

10진수	16진수	문자
0	0x00	NUL
1	0x01	SOH
2	0x02	STX
3	0x03	ETX
4	0x04	EOT
5	0x05	ENQ
6	0x06	ACK
7	0x07	BEL
8	0x08	BS
9	0x09	TAB
10	0x0A	LF
11	0x0B	VT
12	0x0C	FF
13	0x0D	CR
14	0x0E	SO
15	0x0F	SI
16	0x10	DLE
17	0x11	DC1
18	0x12	DC2
19	0x13	DC3
20	0x14	DC4
21	0x15	NAK
22	0x16	SYN

10진수	16진수	문자
23	0x17	ETB
24	0x18	CAN
25	0x19	EM
26	0x1A	SUB
27	0x1B	ESC
28	0x1C	FS
29	0x1D	GS
30	0x1E	RS
31	0x1F	US
32	0x20	Space
33	0x21	!
34	0x22	"
35	0x23	#
36	0x24	$
37	0x25	%
38	0x26	&
39	0x27	'
40	0x28	(
41	0x29	)
42	0x2A	*
43	0x2B	+
44	0x2C	,
45	0x2D	-

10진수	16진수	문자
46	0x2E	.
47	0x2F	/
48	0x30	**0**
49	0x31	**1**
50	0x32	**2**
51	0x33	**3**
52	0x34	**4**
53	0x35	**5**
54	0x36	**6**
55	0x37	**7**
56	0x38	**8**
57	0x39	**9**
58	0x3A	:
59	0x3B	;
60	0x3C	<
61	0x3D	=
62	0x3E	>
63	0x3F	?
64	0x40	@
65	0x41	**A**
66	0x42	**B**
67	0x43	**C**
68	0x44	**D**

10진수	16진수	문자
69	0x45	**E**
70	0x46	**F**
71	0x47	**G**
72	0x48	**H**
73	0x49	**I**
74	0x4A	**J**
75	0x4B	**K**
76	0x4C	**L**
77	0x4D	**M**
78	0x4E	**N**
79	0x4F	**O**
80	0x50	**P**
81	0x51	**Q**
82	0x52	**R**
83	0x53	**S**
84	0x54	**T**
85	0x55	**U**
86	0x56	**V**
87	0x57	**W**
88	0x58	**X**
89	0x59	**Y**
90	0x5A	**Z**
91	0x5B	[

10진수	16진수	문자
92	0x5C	₩
93	0x5D	]
94	0x5E	^
95	0x5F	_
96	0x60	`
97	0x61	**a**
98	0x62	**b**
99	0x63	**c**
100	0x64	**d**
101	0x65	**e**
102	0x66	**f**
103	0x67	**g**
104	0x68	**h**
105	0x69	**i**
106	0x6A	**j**
107	0x6B	**k**
108	0x6C	**l**
109	0x6D	**m**
110	0x6E	**n**
111	0x6F	**o**
112	0x70	**p**
113	0x71	**q**
114	0x72	**r**

10진수	16진수	문자
115	0x73	**s**
116	0x74	**t**
117	0x75	**u**
118	0x76	**v**
119	0x77	**w**
120	0x78	**x**
121	0x79	**y**
122	0x7A	**z**
123	0x7B	{
124	0x7C	|
125	0x7D	}
126	0x7E	~
127	0x7F	DEL

정수 자료형

예약어 short, int, unsigned int, long, unsigned long

1 byte = 8 bit, □□□□□□□□, 1 - 부호비트, 31 - 데이터 비트

<unsigned 부호비트 제외한 양수범위>

구분	허용범위	바이트수
short	-32768 ~ 32767 $(-2^{15}{\sim}0{\sim}2^{15}-1)$	2
int	-2,147,483,648~ 2,147,483,647 $(-2^{31}{\sim}0{\sim}2^{31}-1)$	4
unsigned int	0~4,294,967,295 $(0{\sim}2^{32}-1)$	4
long	-9,223,372,036,854,775,808~ 9,223,372,036,854,775,807 $(-2^{63}{\sim}0{\sim}2^{63}-1)$	8
unsigned long	0~18,446,744,073,709,551,615 $(0{\sim}2^{64}-1)$	8

* 컴퓨터의 운영체제 및 설정에 따라 자료형의 크기는 다를 수 있다

실수 자료형

예약어 float, double, long double
10진법의 실수를 2진법의 실수형으로 변환하여 저장
32 비트 = 1 - 부호비트, 23 - 가수비트, 8 - 지수 비트
64 비트 = 1 - 부호비트, 52 - 가수비트, 11 - 지수 비트
부동소수점 규약 (부호) $\pm$ (가수) $\times 2^{\text{지수}}$

구분	최소 지수	최대 지수	유효 자릿수	바이트 수
float	10^{-38}	10^{38}	6 ~ 7	4
double	10^{-308}	10^{308}	15 ~ 16	8
long double	10^{-4932}	10^{4932}	19 ~ 20	16

* 컴퓨터의 운영체제 및 설정에 따라 자료형의 크기는 다를 수 있다

Day02 - 1. 자료형 - 형변환

형변환

연산자가 서로 다른 형의 피연산자를 가질 때 형변환이 필요

일반적으로 크기가 작은 피연산자를 큰 연산자로 바꿈

배열의 첨자에 float를 쓰는 것과 같은 의미가 맞지 않는 수식은 형변환이 허용되지 않음

형변환 규칙

Rule 1. 피연산자 중 long double이 있으면 나머지는 long double로 변환
Rule 2. 피연산자 중 double이 있으면 나머지는 double로 변환
Rule 3. 피연산자 중 float가 있으면 나머지는 float로 변환
Rule 4. 피연산자 중 char나 short는 int로 변환
Rule 5. 피연산자 중 long이 있으면 나머지는 long으로 변환

type cast: 명시적 형 변환

```
int i ;
float f = 12.34 ;
i = (int) f ;
```

Day02 - 1. 자료형 - 입출력 변환문자

변환문자	기능	자료형	변환문자	기능	자료형
%d	10진수	정수	%ld	Long형(확장형) 10진수	정수
%o	8진수	정수	%lo	Long형(확장형) 8진수	정수
%x	16진수	정수	%lx	Long형(확장형) 16진수	정수
%p	데이터 주소 16진수	정수	%lu	Long형(확장형) 부호 없는 10진수	정수
%u	부호 없는 10진수	정수	%f, %lf	실수 (소수점)	실수
%c	하나의 문자	문자	%e	실수 (지수)	실수
%s	문자열	문자열	%g	%f or %e 길이가 짧은 기준	실수

* % “숫자” “변환문자” 출력 자릿수 설정 가능 - 예시 : %3d, %.2f

Day02 - 4. 예제 코드

```
#include <stdio.h>
int main(){
  int num = 97; //대소문자 별개 인식
  char a = 130, b = -130; // overflow
  float n1 = 1.23456789;
  double n2 = 123.456789;
  printf("a : %5d \n", a);
  printf("b : %-5d \n", b);
  printf("n1 (소수점): %20.2f\n", n1); // 오른쪽 정렬
  printf("n1 (지수형): %20.2e\n", n1);
  printf("n2 (소수점): %-20lf\n", n2); // 왼쪽 정렬
  printf("n2 (지수형): %-20le\n", n2);
  printf("10진수 : %5d \n", num);
  printf(" 8진수 : %5o \n", num);
  printf("16진수 : %5x \n", num);
  printf("주소값 : %5p \n", &num);
  printf("10진수 : %5u \n", num);
  printf("문자값 : %5c \n", num);
  printf("문자열 : %20s \n", "문자열 % s 출력");
  return 0;
}
```

Day02 - 2. 상수 - 리터럴 상수 정수형, 실수형

리터럴 상수

- 읽는 그대로의 의미가 있는 상수

리터럴 상수 정수형

- 10진 상수
  앞에 0이외의 숫자로 시작하는 상수 <100,-20,30,15>
- 2진 상수 <0b00101010>
- 8진 상수 <062(10진수 50), 057(10진수 47)>
- 16진 상수 <0xAA24(10진수 43556), 0xBCDE(48350), 0x1142(4418)>

리터럴 상수 실수형

- 실수 형태 상수
  <12.34, -56.78, 215546.78967, -125512.12523>
- 지수 형태 상수
  <1.23e4, -3.5622e-3, 5.2164e29>

Day02 - 2. 상수 - 심볼릭 상수

심볼릭 상수

상수를 기호화하여 의미있는 이름으로 지어서 쓰는 상수

```
#include <stdio.h>
int main()
{
#define Pyeong 3.3058 // #define A B : A를 B로 교환
  const int Max = 100; //반드시 선언과 동시에 초기화
  const double PI = 3.14; // const 사용시 초기화 하지 않으면 에러
  //Max = 200; //이미 위에서 NUM 상수화
  //PI = 3.15; //이미 위에서 PI 상수화
  printf("Pyeong = %f  \n", Pyeong);   // 3.3058 출력
  printf(" Max     = %d  \n", Max);   // 100 출력
  printf(" PI        = %lf \n", PI);   // 3.14 출력
  return 0;
}
```

Day02 - 3. 변수 선언 - 변수 이름을 짓는 규칙

- 다른 사람도 쉽게 이해 할 수 있도록 직관적인 의미로 이름으로 짓는다.

- 변수 이름은 영문자 대문자, 소문자, _(underline), 숫자로 조합한다.
  (대문자, 소문자는 다른 변수로 인식한다 A, a 는 별개의 변수)

- 숫자는 변수명 첫글자로 쓸 수 없다.

- 예약어는 변수로 사용 할 수 없다.

- 변수명의 길이는 제한이 없지만 되도록 간결하게 작성한다.

- 변수명의 사이에 공백을 넣을 수 없다.

변수 선언 형태
```
자료형 변수명_1, 변수명_2,…;
자료형 변수명 = 상수; // 변수 선언과 동시에 초기화
```

Day02 - 4. 예제 코드 - 성적 종합 프로그램

3과목의 (국어, 수학, 영어) 성적을 입력 받아 출력하기
- 3과목의 점수를 저장할 변수 선언
- 점수 입력 받아 변수에 저장
- 점수 총점 계산하기
- 평균 계산하기
- 3과목의 (국어, 수학, 영어) 성적, 총점, 평균 출력하기

- 편차, 분산 계산하기

- 표준편차 (math.h 헤더파일, sqrt() 사용 : 제곱근 함수)

편차 = 점수 - 평균
분산 =
{ (점수1 - 평균) * (점수1 - 평균) +
  (점수2 - 평균) * (점수2 - 평균) +
  (점수3 - 평균) * (점수3 - 평균) }
   / 과목 수(3)
표준편차 = sqrt(분산);

Day02 - 4. 예제 코드 - 성적 종합 프로그램

```
#include <stdio.h>
int main()
{
    int kor, math, eng, total;
    float ave;
    printf("국어 : ");
    scanf_s("%d", &kor);
    printf("수학 : ");
    scanf_s("%d", &math);
    printf("영어 : ");
    scanf_s("%d", &eng);
    printf("국어 : %d 수학 : %d 영어 : %d\n", kor, math, eng);
    total = kor + math + eng;
    ave = total / 3.0;
    printf("--------------------------------\n");
    printf("총점 : %d\n", total);
    printf("평균 : %f\n", ave);
    return 0;
}
```

Day02 - 4. 예제 코드 - 성적 종합 프로그램

```
#include <stdio.h>
#include <math.h> // sqrt()
int main(){
  int kor, math, eng, total;
  float ave;
  printf("국어 : ");
  scanf_s("%d", &kor);
  printf("수학 : ");
  scanf_s("%d", &math);
  printf("영어 : ");
  scanf_s("%d", &eng);
  printf("국어 : %d 수학 : %d 영어 : %d\n", kor, math, eng);
  total = kor + math + eng;
  ave = total / 3.0;

  float dev_kor = kor - ave;
  float dev_math = math - ave;
  float dev_eng = eng - ave;

  float sq_kor = dev_kor * dev_kor;
  float sq_math = dev_math * dev_math;
  float sq_eng = dev_eng * dev_eng;

  float variance = (sq_kor + sq_math + sq_eng) / 3.0;
  float std_dev = sqrt(variance);
  printf("--------------------------------\n");
  printf("총점 : %d\n", total);
  printf("평균 : %f\n", ave);
  printf("국어 편차 : %f 수학 편차 : %f 영어 편차 : %f\n", dev_kor, dev_math, dev_eng);
  printf("분산 : %f\n", variance);
  printf("표준편차 : %f\n", std_dev);
  return 0;
}
```

Day 03 - Contents

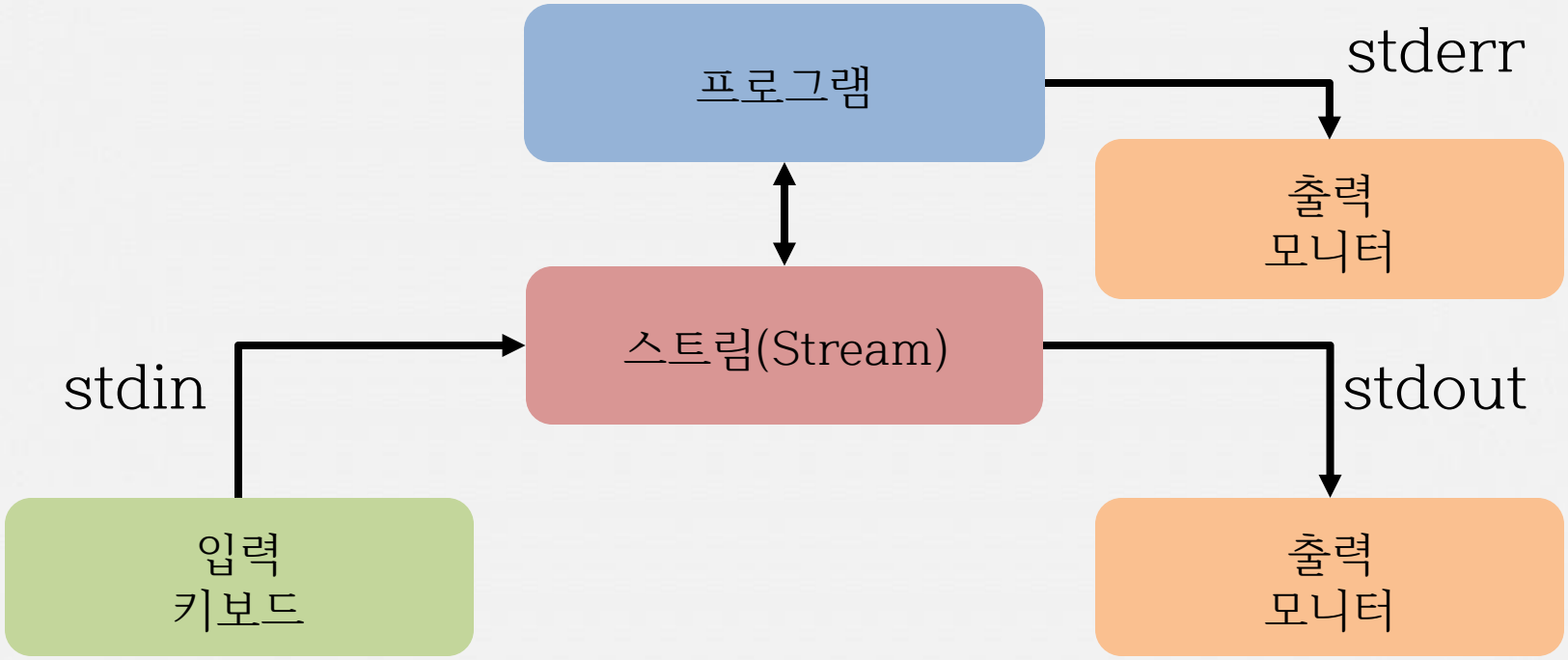
1. 스트림(Stream)

- 스트림
- 입력 버퍼
- 출력 버퍼
- getchar(), getch(), getche(), putchar()
- gets_s(), puts()

2. 예제 코드

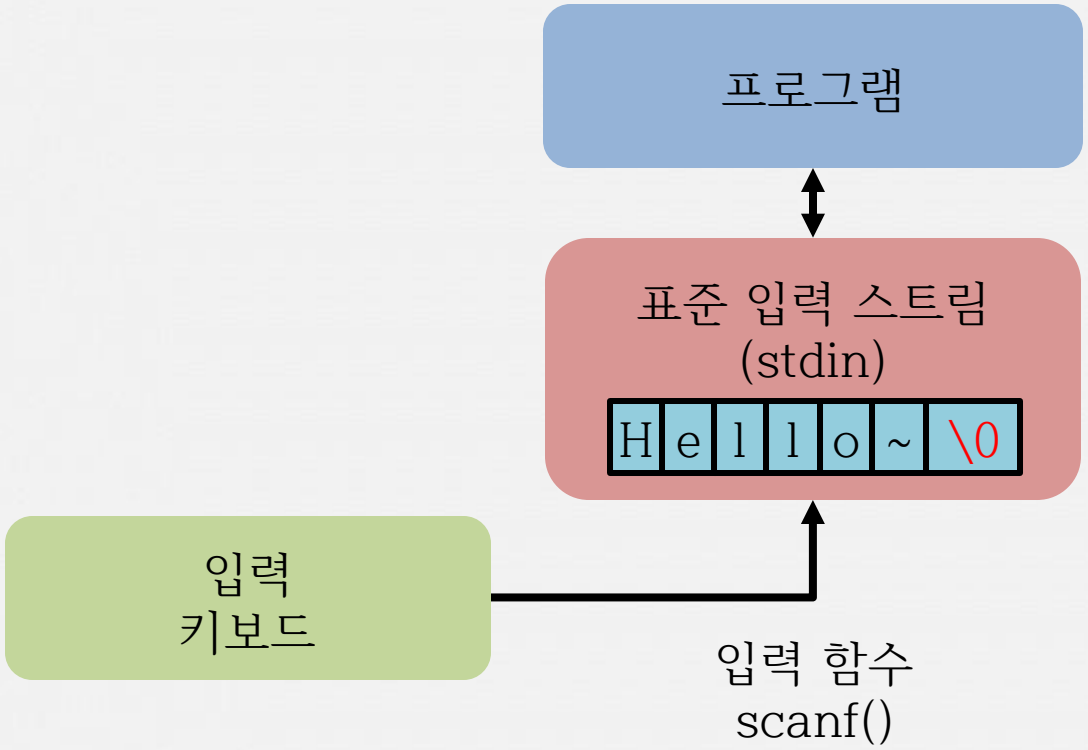
스트림(Stream)

운영체제에서 프로그램과 입출력 장치를 연결하는 가상의 통로를 의미한다.
콘솔 장치에 대한 스트림은 프로그램 실행시 자동으로 생성 & 종료
파일과의 연결은 사용자가 직접 생성과 소멸이 필요 (파일 입출력)



표준 입력 스트림 (stdin)

키보드(입력 장치) <-> 프로그램을 연결하는 입력 통로(버퍼)를 말한다.
입력 장치로부터 데이터를 1개씩 저장하는 버퍼

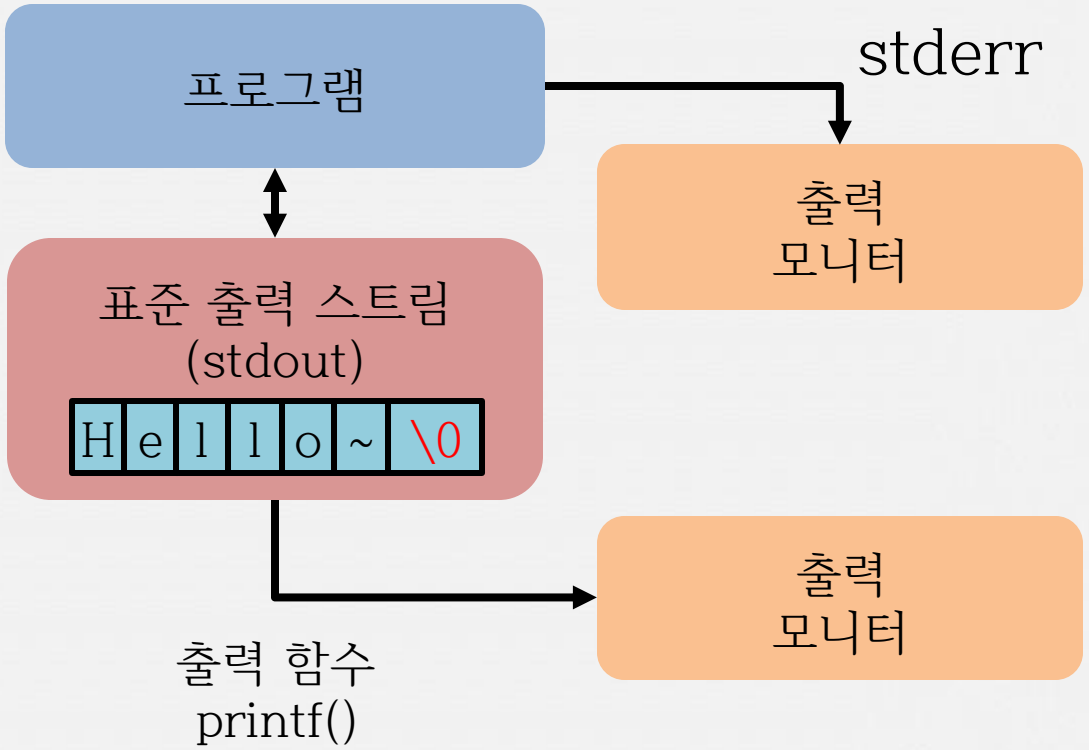


표준 출력 스트림 (stdout)

모니터(출력 장치) <-> 프로그램을 연결하는 출력 통로(버퍼)를 말한다.

표준 에러 스트림 (stderr)

에러에 관련한 출력을 담당하는 스트림 (버퍼 없이 바로 출력이 가능)



단일문자 입출력 함수

int getchar(void); (함수의 원형)
표준 입력 스트림(stdin)인 키보드로부터 하나의 문자를 입력 받는 함수

int _getch(void); (함수의 원형) 화면에 키 값 출력 X
콘솔 입력으로 표준 입력 버퍼에 저장되지 않고 입력한 키 값을 즉시 반환
unsigned short _getwch(void); 유니코드(2 byte) 입력 받을 때

int _getche(void); (함수의 원형) 화면에 키 값 출력 O
콘솔 입력으로 표준 입력 버퍼에 저장되지 않고 입력한 키 값을 즉시 반환
unsigned short _getwche(void); 유니코드(2 byte) 입력 받을 때

int putchar(int c); (함수의 원형)
표준 출력 스트림(stdout)에 하나의 문자를 출력하는 함수

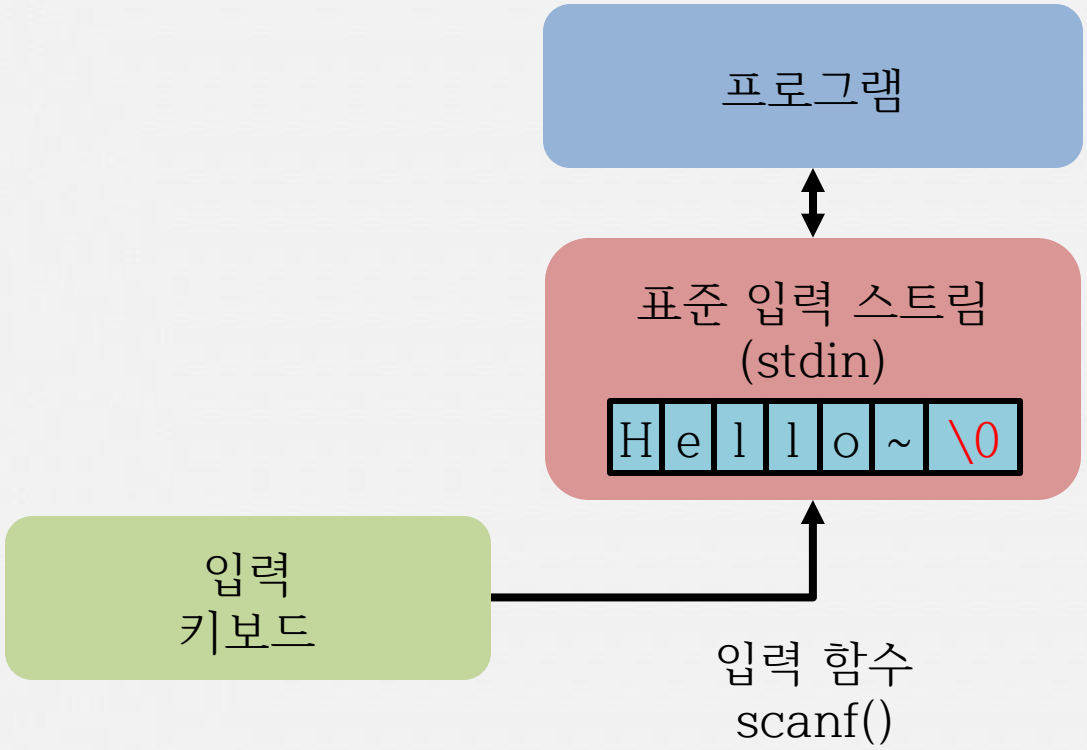
Day03 - 1. 스트림(Stream)

```
#include<stdio.h>
#include<conio.h> // console 입출력 헤더파일 _getch(), _getche()
int main() {
    char 문자;
    int 정수;
    printf("10(LF : Line Feed 다음줄로 이동), 13(CR : Cariage Return 이 줄 첫째칸으로 이동)\n");
    printf("문자를 입력 하세요.");
    scanf_s("%d", &정수);
    printf("%d\n", 정수);
    while (1) {
        printf("\ngetchar :");
        문자 = getchar();
        printf(" 입력한 문자 :");
        putchar(문자);
        printf("\tASCII코드 값은 %d 입니다.\n", 문자);
        while (getchar() != '\n');

        printf("\n_getch :");
        문자 = _getch(); // Enter 없이 바로 입력, 에코 없음
        printf(" 입력한 문자 :");
        putchar(문자);
        printf("\tASCII코드 값은 %d 입니다.\n", 문자);

        printf("\n_getche :");
        문자 = _getche(); // Enter 없이 바로 입력, 에코 있음
        printf(" 입력한 문자 :");
        putchar(문자);
        printf("\tASCII코드 값은 %d 입니다.\n", 문자);
    }
    return 0;
}
```

Day03 - 1. 스트림(Stream)

문자열

여러 개의 문자를 이어서 저장하고 있는 데이터
(1개의 데이터 공간에 1개의 글자 값(ASCII)을 저장)
문자열의 마지막 끝은 \0(NULL 문자) 필요하다.



문자열 입출력 함수

char* gets_s(char* _buffer, rsize_t _size); (함수의 원형)
표준 입력 스트림 (stdin) 에서 문자열을 읽고 buffer에 저장
문자열의 첫 번째 줄 바꿈 문자(\n)를 포함하여 최대의 모든 문자열 구성
줄 바꿈 문자(\n)를 null 문자(\0)로 변경
문자열의 시작 주소 반환

_buffer : 저장 공간
_size : 저장 공간의 크기
문자열 중간에 공백이나 탭이 있어도 Enter 키를 누를 때까지 입력 받는다.

int puts(char const * _Buffer); (함수의 원형)
표준 출력 스트림(stdout)에 문자열을 쓰는 함수

문자열의 종료 null 문자('\0')를 출력 스트림의 줄 바꿈 문자('\n')로 변경

Day03 - 1. 스트림(Stream)

```
#include<stdio.h>
int main() {
    char GetName[20], Sname[20];
    puts("영문 이름을 입력 하세요.");
    puts("gets_s :");
    gets_s(GetName,  sizeof(GetName));
    printf("이름은 : %s 입니다.\n", GetName);
    puts(GetName);

    puts("scanf_s :");
    scanf_s("%s", Sname, (int)sizeof(Sname));
    printf("이름은 : %s 입니다.\n", Sname);
    puts(Sname);

    return 0;
}
```

Day 04 - Contents

1. 연산자

- 산술연산자
- 관계연산자
- 논리연산자
- 증감, 이동연산자
- 대입연산자
- 비트 논리연산자
- 조건연산자
- 형변환연산자
- sizeof연산자
- 연산자 우선순위

2. 예제 코드

- 연산자 예제 코드
- 과제

Day04 - 1. 연산자

산술연산자

연산자	기능	사용 예	우선순위
*	곱셈	2 * 3 = 6	1
/	나눗셈	3 / 2 = 1 (정수형 자료) 3.0 / 2.0 = 1.5 (실수형 자료)	1
%	나머지	6 % 4 = 2 (정수형 자료만 가능)	1
+	덧셈	2 + 3 = 5	2
-	뺄셈	3 - 2 = 1	2

관계연산자

연산자	기능	사용 예	우선순위
>	크다	a=2>3 a=0 (거짓)	1
>=	크거나 같다	a=3>=2 a=1 (참)	1
<	작다	a=2<3 a=1 (참)	1
<=	작거나 같다	a=3<=2 a=0 (거짓)	1
==	같다	a=3==2 a=0 (거짓)	2
!=	같지 않다	a=3!=2 a=1 (참)	2

Day04 - 1. 연산자

논리연산자

연산자	기능	사용 예	우선순위
!	거짓->참 (!0=1) 참->거짓(!1=0)	re=!a a=0 -> re=1	1
&&	AND 연산 조건이 모두 만족 -> 참	re=a&&b a=1,b=1 -> re=1	2
||	OR 연산 조건 1개라도 만족 -> 참	re=a||b a=1,b=0 -> re=1	3

Day04 - 1. 연산자

증감연산자

연산자	기능	사용 예 전위	사용 예 후위
++	값을 1 증가	++a -> a=a+1	a++ -> a=a+1
--	값을 1 감소	--a -> a=a-1	a-- -> a=a-1

전위 연산자 먼저 증감 실행
후위 연산자 식 계산이 먼저 끝나고 다음 식으로 넘어갈 때 증감 실행

시프트연산자

연산자	기능	사용 예
<<	비트를 좌측으로 이동	a=b<<2
>>	비트를 우측으로 이동	a=b>>2

00001100101 = a(101)
00110010100 = a << 2 = 404

00001100101 = a(101)
00000011001 = a >> 2 = 25

비트논리연산자

연산자	기능	사용 예	우선순위
~	비트를 반전	re=~a	1
&	두 비트가 모두 1이면 참	re=a&b	2
^	두 비트가 서로 다르면 1	re=a^b	3
|	두 비트가 하나라도 1이면 참	re=a|b	4

~0000 = 1111

```
  1010 0101        1010 0101        1010 0101
& 1111 0000      ^ 1111 0000      | 1111 0000
-----------      -----------      -----------
  1010 0000        0101 0101        1111 0101
```

대입연산자

연산자	기능	사용 예
=	우변의 결과를 좌측 변수에 저장	a=b+c
+=	a=a+b	a+=b
-=	a=a-b	a-=b
*=	a=a*b	a*=b
/=	a=a/b	a/=b
%=	a=a%b	a%=b
&=	a=a&b	a&=b
|=	a=a|b	a|=b
^=	a=a^b	a^=b
<<=	a=a<<b	a<<=b
>>=	a=a>>b	a>>=b

Day04 - 1. 연산자

조건연산자

형식

(조건)?조건 참 결과: 조건 거짓 결과;

```
#include <stdio.h>

int main() {
    int a=10, b=30, c=20, max;
    max=(a>b)?a:b;
    max=(max>c)?max:c;
    printf("가장 큰 수 : %d",max);
    return 0;
}
```

형 변환 연산자

형식

(변환 할 자료형)변수 또는 상수;

```
#include <stdio.h>

int main() {
    int a=10, b=30, sum;
    float ave;
    sum = a+b;
    ave = (float)sum/2;
    printf("합 : %d, 평균 : %.2f",sum,ave);
    return 0;
}
```

Day04 - 1. 연산자

sizeof 연산자

형식

sizeof(인수); 인수의 자료형 기억장소의 크기를 구한다.

```
#include <stdio.h>
int main(){
    char str[10];
    printf("       \"a\" : %3d \n", (int)sizeof("a")); // 문자 인식
    printf("       \'a\' : %3d \n", (int)sizeof('a')); // 숫자 인식
    printf("     str[10]: %3d \n", (int)sizeof(str));
    printf("       int  : %3d \n", (int)sizeof(int));
    printf("       long : %3d \n", (int)sizeof(long));
    printf("      float : %3d \n", (int)sizeof(float));
    printf("     double : %3d \n", (int)sizeof(double));
    printf("long double : %3d \n", (int)sizeof(long double));
    return 0;
}
```

Day04 - 1. 연산자

기호	연산 유형	결합 실행순서	우선순위
[ ] ( ) . -> ++--(후위)	식	왼쪽 -> 오른쪽	1
sizeof & * + - ~ ! ++--(전위)	단항	**오른쪽 -> 왼쪽**	2
형 변환	단항	**오른쪽 -> 왼쪽**	3
* / %	곱하기	왼쪽 -> 오른쪽	4
+ -	더하기	왼쪽 -> 오른쪽	5
<< >>	비트 시프트	왼쪽 -> 오른쪽	6
< > <= >=	관계	왼쪽 -> 오른쪽	7
== !=	같음	왼쪽 -> 오른쪽	8
&	비트 AND	왼쪽 -> 오른쪽	9
^	비트 제외 OR	왼쪽 -> 오른쪽	10
|	비트 포함 OR	왼쪽 -> 오른쪽	11
&&	논리 AND	왼쪽 -> 오른쪽	12
||	논리 OR	왼쪽 -> 오른쪽	13
? :	조건식	**오른쪽 -> 왼쪽**	14
= *= /= %= += -= <<= >>= &= ^= |=	단순 및 복합 할당	**오른쪽 -> 왼쪽**	15
,	순차적 계산	왼쪽 -> 오른쪽	16

https://learn.microsoft.com/ko-kr/cpp/c-language/precedence-and-order-of-evaluation?view=msvc-170

Day04 - 2. 예제코드

```
#include <stdio.h>
int main() {
  int x, y, z, i, j, a, b, c, d, re, re1, re2, re3;

  i = 5, j = 3;
  printf("%d + %d = %d\n", i, j, i + j);
  printf("%d - %d = %d\n", i, j, i - j);
  printf("%d * %d = %d\n", i, j, i * j);
  printf("%d / %d = %d\n", i, j, i / j);
  printf("%d %% %d = %d\n", i, j, i % j);
  printf("전위 연산 i = %d, j = %d\n", ++i, ++j);
  printf("후위 연산 i = %d, j = %d\n", i++, j++);
  printf("연산 i = %d, j = %d\n", i, j);

  a = 10, b = 10;
  printf("원래의 a 값 : %d\n", a);
  printf("원래의 b 값 : %d\n", b);
  printf("--a의 결과 : %d\n", --a);
  printf("--연산후의 a 값 : %d\n", a);
  printf("b--의 결과 : %d\n", b--);
  printf("--연산후의 b 값 : %d\n", b);

  x = 5;
  a = ++x * x--;// 전치 연산으로 x=6으로 계산
  b = x * 10;
  // 위에 식에서 후치 연산으로 x=5로 감소
  printf("a=%d b=%d x=%d\n", a, b, x);
  a = 2, b = 3, c = 9;
  re = ++a + ++b;
  printf("re = %d, a = %d, b = %d, c = %d \n\n", re, a, b, c);
  re = c++;
  printf("re = %d, a = %d, b = %d, c = %d \n\n", re, a, b, c);
  re = --b * --b - b++; // 전치 연산으로 b=2로 계산
  printf("re = %d, a = %d, b = %d, c = %d \n\n", re, a, b, c);
  re = --c + c-- + a++;
  printf("re = %d, a = %d, b = %d, c = %d \n\n", re, a, b, c);

  a = 10;
  printf("원래 a값 : %d\n", a);
  printf("2를 더한 값 : %d\n", a += 2);
  printf("2를 뺀 값 : %d\n", a -= 2);
  printf("2를 곱한 값 : %d\n", a *= 2);
  printf("2를 나눈 값 : %d\n", a /= 2);
  a++;
  printf("2를 나눈 나머지 값 : %d\n", a %= 2);

  a = 4;
  re1 = a > 2;
  printf("re1= %d\n", re1);
  re2 = a < 2;
  printf("re2= %d\n", re2);
  re3 = a == 4;
  printf("re3= %d\n", re3);
  a = 3, b = 5, c = 1, d = 3;
  printf("1. %d\n", a > b);
  printf("2. %d\n", a < b);
  printf("3. %d\n", c > d);
  printf("4. %d\n", !(c < d));
  printf("5. %d\n", (a > b) && (c < d));
  printf("6. %d\n", (a > b) || (c < d));
  printf("7. %d\n", (a < b) && (c < d));
  printf("8. %d\n", (a < b) || (c < d));

  a = 0xD1, b = 0xC4;
  printf("9. %X %X\n", a, b);
  printf("10. %X %X\n", ~a, ~b);
  printf("11. %X\n", a & b);
  printf("12. %X\n", a | b);
  printf("13. %X\n", a ^ b);

  a = 209;
  printf("14. %d\n", a);
  printf("15. %d\n", a << 1);
  printf("16. %d\n", a << 2);
  printf("17. %d\n", a << 3);
  printf("18. %d\n", a >> 1);
  printf("19. %d\n", a >> 2);
  printf("20. %d\n", a >> 3);
  x = 1, y = 0, z = 1;
  re = ++x && y++ && z++;
  // y=0 이므로 y까지 실행, z를 실행하지 않음
  printf("21. re = %d, x = %d, y = %d, z = %d \n", re, x, y, z);
  re = ++x || ++y || ++z;
  // x가 참이므로 y,z 를 실행하지 않음
  printf("22. re = %d, x = %d, y = %d, z = %d \n", re, x, y, z);
  re = --x && ++y || ++z;
  //y까지 참이므로 z를 실행하지 않음
  printf("23. re = %d, x = %d, y = %d, z = %d \n", re, x, y, z);
  re = ++x || ++y && z++;
  // x가 참이므로 y,z를 실행하지 않음
  //re = (++x || ++y) && z++;
  // && 연산자로 z까지 실행
  printf("24. re = %d, x = %d, y = %d, z = %d \n", re, x, y, z);
  re = x++ < y++ ? x++ : y++;
  // (x < y) ? x : y; 조건식이 끝나면 후위 연산 실행,
  // 결과값에서 1번 더 실행
  printf("25. re = %d, x = %d, y = %d, z = %d \n", re, x, y, z);
  return 0;
}
```

```
  1101 0001    D1 = a
  1100 0100    C4 = b
----------------
  1100 0000    C0 (&)
  1101 0101    D5 ( | )
  0001 0101    15 ( ^ )
  0010 1110    2E ( ~ )
  0011 1011    3B ( ~ )
```

Day 05 - Contents

1. 제어(조건)문
   - if문
   - if~else문
   - 다중 if문
   - 중첩 if문
2. 예제 코드

Day05 - 1. 제어(조건)문

if문

형식

```
if (조건식)
{
    명령문1;
    명령문2;
    …;
}
```

예시

```
#include <stdio.h>
int main() {
  int code;
  printf("숫자를 입력하세요 : ");
  scanf_s("%d", &code);
  if (code == 1) printf("1을 입력했습니다.\n");
  printf("END\n");
  return 0;
}
```

조건식이 0이외의 값 or 참이면 중괄호 { } 안의 명령문 실행
값이 0이거나, 거짓이면 중괄호 { } 다음 문장으로 넘어간다.

조건 수행문이 단문일 경우 중괄호 생략 가능.

Day05 - 1. 제어(조건)문

절대값 만들기

- 정수형 변수 1개를 입력 받기
- 입력 받은 수

  음수의 경우 {
   양수로 변경
  }

  양수의 경우 { 내용 없음 }

- 결과 출력

Day05 - 1. 제어(조건)문

```
#include <stdio.h>
int main(){
  int abs;
  printf("\t [ 절대값 만들기 ] \n");
  printf("숫자를 입력하세요 : ");
  scanf_s("%d", &abs);
  if (abs < 0){
    abs = abs * (-1);
    printf("음수 -> 양수 변경 완료.\n");
  }
  printf("절대값 결과 : %5d \n", abs);
  return 0;
}
```

Day05 - 1. 제어(조건)문

if~else문

형식

```
if (조건식)
{
    명령문1;
    명령문2;
    …;
}
else
{
    명령문1;
    명령문2;
    …;
}
```

예시

```
#include <stdio.h>
int main() {
  int code;
  printf("숫자를 입력하세요 : ");
  scanf_s("%d", &code);
  if (code == 1) printf("1을 입력했습니다.\n");
  else printf("1이 아닙니다.\n");
  printf("END\n");
  return 0;
}
```

조건식이 0이외의 값 or 참이면 if 중괄호 { } 안의 명령문 실행
값이 0이거나, 거짓이면 else 중괄호 { } 안의 명령문 실행

명령문이 단문일 경우 중괄호 생략 가능.

Day05 - 1. 제어(조건)문

다중 if문

형식

```
if (조건식1)
{
    명령문1;
}
else if(조건식2)
{
    명령문2;
}
else if(조건식3)
{
    명령문3;
}
else
{
    명령문1;
    명령문2;
    …;
}
```

조건식1이 참이면 명령문1 실행,
조건식2가 참이면 명령문2 실행,
조건식3이 참이면 명령문3 실행,
조건식1~3이 모두 거짓이면 else 명령문 실행

예시

```
#include <stdio.h>
int main() {
  int code;
  printf("숫자를 입력하세요 : ");
  scanf_s("%d", &code);
  if (code == 1) printf("1을 입력했습니다.\n");
  else if (code == 2) printf("2을 입력했습니다.\n");
  else if (code == 3) printf("3을 입력했습니다.\n");
  else printf("1~3이 아닙니다.\n");
  printf("END\n");
  return 0;
}
```

Day05 - 1. 제어(조건)문

중첩 if문

형식

```
if (조건식1)
{
    if (조건식2)
    {
        명령문1;
    }
    else
    {
        명령문2;
    }
}
else
{
    명령문3;
}
```

조건식1도 참이고 조건식2도 참이면 명령문1 실행,
조건식1이 참이고 조건식2가 거짓이면 명령문2 실행,
조건식1이 거짓이면 명령문3 실행,

예시

```
#include <stdio.h>
int main() {
  int code;
  printf("숫자를 입력하세요 : ");
  scanf_s("%d", &code);
  if (code >= 10){
    if (code < 20) printf("10이상~20미만 입니다.\n");
    else printf("20이상 입니다.\n");
  }
  else printf("10보다 작습니다. \n");
  printf("END\n");
  return 0;
}
```

Day05 - 2. 예제 코드

알파벳 소문자 1글자를 대문자로 변환하는 프로그램
(알파벳 이외 문자를 입력 받을 시 예외처리)

알파벳 소문자를 입력하세요 : a
입력하신 알파벳 대문자는 A 입니다.
알파벳 소문자를 입력하세요 : A
입력하신 알파벳 대문자는 A 입니다.
알파벳 소문자를 입력하세요 : 7
입력하신 문자는 알파벳이 아닙니다.

Hint - ASCII 코드
1. 소문자 & 대문자의 규칙 찾기
2. 프로그램 순서 정리
   (1) 변수 개수 몇 개로 할 지 정하기
   (2) 입력 -> 변환처리 -> 출력 순서 정리

Day02 - 2. 상수 - ASCII코드

10진수	16진수	문자
0	0x00	NUL
1	0x01	SOH
2	0x02	STX
3	0x03	ETX
4	0x04	EOT
5	0x05	ENQ
6	0x06	ACK
7	0x07	BEL
8	0x08	BS
9	0x09	TAB
10	0x0A	LF
11	0x0B	VT
12	0x0C	FF
13	0x0D	CR
14	0x0E	SO
15	0x0F	SI
16	0x10	DLE
17	0x11	DC1
18	0x12	DC2
19	0x13	DC3
20	0x14	DC4
21	0x15	NAK
22	0x16	SYN

10진수	16진수	문자
23	0x17	ETB
24	0x18	CAN
25	0x19	EM
26	0x1A	SUB
27	0x1B	ESC
28	0x1C	FS
29	0x1D	GS
30	0x1E	RS
31	0x1F	US
32	0x20	Space
33	0x21	!
34	0x22	"
35	0x23	#
36	0x24	$
37	0x25	%
38	0x26	&
39	0x27	'
40	0x28	(
41	0x29	)
42	0x2A	*
43	0x2B	+
44	0x2C	,
45	0x2D	-

10진수	16진수	문자
46	0x2E	.
47	0x2F	/
48	0x30	**0**
49	0x31	**1**
50	0x32	**2**
51	0x33	**3**
52	0x34	**4**
53	0x35	**5**
54	0x36	**6**
55	0x37	**7**
56	0x38	**8**
57	0x39	**9**
58	0x3A	:
59	0x3B	;
60	0x3C	<
61	0x3D	=
62	0x3E	>
63	0x3F	?
64	0x40	@
65	0x41	**A**
66	0x42	**B**
67	0x43	**C**
68	0x44	**D**

10진수	16진수	문자
69	0x45	**E**
70	0x46	**F**
71	0x47	**G**
72	0x48	**H**
73	0x49	**I**
74	0x4A	**J**
75	0x4B	**K**
76	0x4C	**L**
77	0x4D	**M**
78	0x4E	**N**
79	0x4F	**O**
80	0x50	**P**
81	0x51	**Q**
82	0x52	**R**
83	0x53	**S**
84	0x54	**T**
85	0x55	**U**
86	0x56	**V**
87	0x57	**W**
88	0x58	**X**
89	0x59	**Y**
90	0x5A	**Z**
91	0x5B	[

10진수	16진수	문자
92	0x5C	₩
93	0x5D	]
94	0x5E	^
95	0x5F	_
96	0x60	`
97	0x61	**a**
98	0x62	**b**
99	0x63	**c**
100	0x64	**d**
101	0x65	**e**
102	0x66	**f**
103	0x67	**g**
104	0x68	**h**
105	0x69	**i**
106	0x6A	**j**
107	0x6B	**k**
108	0x6C	**l**
109	0x6D	**m**
110	0x6E	**n**
111	0x6F	**o**
112	0x70	**p**
113	0x71	**q**
114	0x72	**r**

10진수	16진수	문자
115	0x73	**s**
116	0x74	**t**
117	0x75	**u**
118	0x76	**v**
119	0x77	**w**
120	0x78	**x**
121	0x79	**y**
122	0x7A	**z**
123	0x7B	{
124	0x7C	|
125	0x7D	}
126	0x7E	~
127	0x7F	DEL

Day05 - 2. 예제 코드

```
#include <stdio.h>
int main() {
  char c;
  while (1) {
    printf("알파벳 소문자를 입력하세요 : ");
    scanf_s("%c", &c);
    while (getchar() != '\n');
    if (c >= 'a' && c <= 'z') {
      c = c - 32;
      printf("입력하신 알파벳 대문자는 %c 입니다.\n", c);
    }
    else if (c >= 'A' && c <= 'Z') {
      printf("입력하신 알파벳 대문자는 %c 입니다.\n", c);
    }
    else {
      printf("입력하신 문자는 알파벳이 아닙니다.\n");
    }
  }
  return 0;
}
```

Day05 - 2. 예제 코드

3개의 과목 점수를 입력 받아 평균값으로 등급 나누기
(1과목 점수의 범위는 0~100점)

- 3개 과목 점수 입력 받기

- 3과목 평균값 계산하기

- 평균값 90 ~ 이상 : A 등급
  평균값 80 ~ 이상 : B 등급
  평균값 70 ~ 이상 : C 등급
  평균값 60 ~ 이상 : D 등급
  평균값 ~ 60 미만 : F 등급

- 결과 출력
  3개의 과목 점수 , 평균값, 등급

Day05 - 2. 예제 코드

```
#include <stdio.h>
int main(){
    int sub1, sub2, sub3;
    char grade;

    printf(" 3 과목 점수 입력 : ");
    scanf_s("%d %d %d", &sub1, &sub2, &sub3);

    double ave = (double)(sub1 + sub2 + sub3) / 3;

    if (ave >= 90){ grade = 'A'; }
    else if (ave >= 80){ grade = 'B'; }
    else if (ave >= 70){ grade = 'C'; }
    else if (ave >= 60){ grade = 'D'; }
    else{ grade = 'F'; }

    printf("평균은 %.2f로 %c등급입니다.\n", ave, grade);
    return 0;
}
```

Day05 - 2. 예제 코드

3개의 숫자 입력 받아 작은 값부터 출력하기

- 값 1~3 : 입력 받기

- 방법 1)
  가장 작은 값 찾기 (반복)
  값 1~3 중에서 2개를 골라 우선 비교
  결과와 나머지 값 비교

- 방법 2)
  작은 값을 찾으면 교환해 순서 변경

- 결과 출력

Day05 - 2. 예제 코드

```
#include <stdio.h>
int main() {
    int a, b, c;
    while (1) {
        printf("숫자 3개를 입력하세요! : ");
        scanf_s("%d %d %d", &a, &b, &c);
        if (a > b) {
            if (a > c) {
                if (b > c) {
                    printf("%d %d %d\n", c, b, a);
                }
                else {
                    printf("%d %d %d\n", b, c, a);
                }
            }
            else {
                if (a > b) {
                    printf("%d %d %d\n", b, a, c);
                }
                else {
                    printf("%d %d %d\n", a, b, c);}
                }
            }
        }
        else {
            if (b > c) {
                if (a > c) {
                    printf("%d %d %d\n", c, a, b);
                }
                else {
                    printf("%d %d %d\n", a, c, b);
                }
            }
            else {
                if (a>b) {
                    printf("%d %d %d\n", b, a, c);
                }
                else {
                    printf("%d %d %d\n", a, b, c);
                }
            }
        }
    }
return 0;
```

Day05 - 2. 예제 코드

```
#include <stdio.h>
// #define   A   B  :  A를 B로 변경한다
#define swap(type,val1,val2) {type temp=val1; val1=val2; val2=temp;}
int main() {
    int a, b, c;
    while (1) {
        printf("숫자 3개를 입력하세요! : ");
        scanf_s("%d %d %d", &a, &b, &c);
        if (a > b) {
            swap(int, a, b);
        }
        if (a > c) {
            swap(int, a, c);
        }
        if (b > c) {
            swap(int, b, c);
        }
        printf("%d %d %d\n", a, b, c);
    }
    return 0;
}
```

Day 06 - Contents

1. 제어(조건)문
   - switch~case
   - 중첩 switch 문
   - if 문, switch 문 비교
2. 예제 코드

Day06 - 1. 제어(조건)문

switch~case문 (break)

형식

```
switch(수식)
{
  case 값1: 명령문1; break;
  case 값2: 명령문2; break;
  case 값3: 명령문3; break;
  default : 명령문4:
}
```

(수식) 정수형 데이터만 가능.
(int, char 자료형만 가능)

값1,…은 정수만 가능하다.

'f', 'e', 'h' 문자로 인식하지 않고 숫자로 인식

```
#include <stdio.h>
int main() {
  char c;
  printf("f, e, h 메뉴 코드를 입력하세요 : ");
  scanf_s("%c", &c);
  switch (c)
  {
  case 'f': printf("file menu\n"); break;
  case 'e': printf("edit menu\n"); break;
  case 'h': printf("help menu\n"); break;
  default: printf("error");
  }
  return 0;
}
```

수식의 결과에 따라 해당하는 값 부분의 명령을 수행한다.
해당하는 값이 없을 경우에는 default의 명령문을 수행한다.
특히, case값과 명령문 사이에 콜론(:)이 사용된다.

중첩 switch문

형식

```
switch(수식1)
{
  case 값1-1:
  {
        switch(수식2)
        {
              case 값2-1: 명령문1; break;
              case 값2-2: 명령문2; break;
              case 값2-3: 명령문3; break;
              default : 명령문4:
        }
  }
  break;
  case 값1-2: 명령문2; break;
  case 값1-3: 명령문3; break;
  default : 명령문4:
}
```

수식1의 결과에 따라 해당하는 값 '1-' 부분의 명령을 수행한다.
결과값이 1-1일경우 수식2의 결과에 따라 해당하는 값 '2-' 부분의 명령 수행.
해당하는 값이 없을 경우에는 default의 명령문을 수행한다.

Day06 - 1. 제어(조건)문

if 문 , switch문 비교

	if 문	switch 문
사용 가능한 자료형	char, int, float, double	char, int 정수만 가능
특정 범위 조건	부분 조건 설정 가능	복잡함
비교 연산	비교 가능	복잡함
변수 선언	가능	중괄호 필요
실행속도	조건 모두 실행, 비효율적	효율적

```
if(조건)
{ if(조건)
  { 명령문 1; }
}
else if(조건)
{ 명령문 2; }
else if(조건)
{ 명령문 3; }
else
{ 명령문 4; }
```

```
switch (수식1)
  {
  case 값1: 명령문1; break;
  case 값2: 명령문2; break;
  case 값3: 명령문3; break;
  default: 명령문4:
  }
```

Day06 - 2. 예제 코드

점수 1개 입력하여 등급 정하기
(점수의 범위는 0~100점)

- 1개의 점수 입력 받기

- switc문을 사용해 등급 나누기
  90~ 이상의 경우 : A 등급
  80~ 이상의 경우 : B 등급
  70~ 이상의 경우 : C 등급
  60~ 이상의 경우 : D 등급
  ~ 60미만의 경우 : F 등급

- 결과값 출력
  입력한 점수 [ ][ ][ ]점, [ ] 등급

Day06 - 2. 예제 코드

```
#include <stdio.h>
int main()
{
  int score;
  char grade;
  while (1) {
    printf("점수 입력 : ");
    scanf_s("%d", &score);
    switch (score / 10)
    {
    case 10:
    case 9: grade = 'A'; break;
    case 8: grade = 'B'; break;
    case 7: grade = 'C'; break;
    case 6: grade = 'D'; break;
    default: grade = 'F';
    }
    printf("%d점 %c등급\n", score, grade);
  }
  return 0;
}
```

Day06 - 2. 예제 코드

[단위변환 프로그램]
1. 인치 <-> 센티미터
    (1) 인치 -> 센티미터
    (2) 센티미터 -> 인치
2. 화씨 <-> 섭씨
    (1) 화씨 -> 섭씨
    (2) 섭씨 -> 화씨

프로그램 순서 정리
(1) 변수 개수 몇 개로 할 지 정하기
(2) 입력 -> 변환처리 -> 출력 순서 정리

1. 인치 <-> 센티미터
2. 화씨 <-> 섭씨
메뉴를 입력하세요. :

(1) 인치 -> 센티미터
(2) 센티미터 -> 인치
메뉴를 입력하세요. :

(1) 인치 -> 센티미터
인치를 입력하세요. :

Day06 - 2. 예제 코드

```
#include <stdio.h>
int main() {
  int main_menu, sub_menu;
  float cel, fah;
  float centi, inch;

  while (1) {
    printf("[단위 변환 프로그램]\n");
    printf("1. inch <-> cm\n");
    printf("2. ℉ <-> ℃\n");
    printf("메뉴를 선택하세요. :");
    scanf_s("%d", &main_menu);

    switch (main_menu) {
    case 1:
      printf("1. inch -> cm\n");
      printf("2. cm -> inch\n");
      printf("메뉴를 선택하세요. :");
      scanf_s("%d", &sub_menu);
      switch (sub_menu) {
      case 1:
        printf(" inch 를 입력하세요.:");
        scanf_s("%f", &inch);
        centi = inch * 2.54;
        printf("%.3f inch = %.3f cm\n", inch, centi);
        break;
      case 2:
        printf(" cm 를 입력하세요.:");
        scanf_s("%f", &centi);
        inch = centi / 2.54;
        printf("%.3f cm = %.3f inch\n", centi, inch);
        break;
      }
      break;
    case 2:
      printf("1. ℉ <-> ℃\n");
      printf("2. ℃ <-> ℉\n");
      printf("메뉴를 선택하세요. :");
      scanf_s("%d", &sub_menu);
      switch (sub_menu) {
      case 1:
        printf(" ℉ 를 입력하세요.:");
        scanf_s("%f", &fah);
        cel = (5.0 / 9.0) * (fah - 32.0);
        printf("%.3f ℉ = %.3f ℃\n", fah, cel);
        break;
      case 2:
        printf(" ℃ 를 입력하세요.:");
        scanf_s("%f", &cel);
        fah = (9.0 / 5.0) * cel + 32.0;
        printf("%.3f ℃ = %.3f ℉\n", cel, fah);
        break;
      }
      break;
    }
  }
  return 0;
}
```

Day06 - 2. 예제 코드

```
#include <stdio.h>
int main() {
  int main_menu, sub_menu;
  float cel, fah;
  float centi, inch;
  while (1) {
    printf("[단위 변환 프로그램]\n");
    printf("1. inch <-> cm\n");
    printf("2. ℉ <-> ℃\n");
    printf("메뉴를 선택하세요. :");
    scanf_s("%d", &main_menu);
    if (main_menu == 1) {
      printf("1. inch -> cm\n");
      printf("2. cm -> inch\n");
      printf("메뉴를 선택하세요. :");
      scanf_s("%d", &sub_menu);
      if (sub_menu == 1) {
        printf(" inch 를 입력하세요.:");
        scanf_s("%f", &inch);
        centi = inch * 2.54;
        printf("%.3f inch = %.3f cm\n", inch, centi);}
      }
      else if (sub_menu == 2) {
        printf(" cm 를 입력하세요.:");
        scanf_s("%f", &centi);
        inch = centi / 2.54;
        printf("%.3f cm = %.3f inch\n", centi, inch);
      }
    }
    else if (main_menu == 2) {
      printf("1. ℉ <-> ℃\n");
      printf("2. ℃ <-> ℉\n");
      printf("메뉴를 선택하세요. :");
      scanf_s("%d", &sub_menu);
      if (sub_menu == 1) {
        printf(" ℉ 를 입력하세요.:");
        scanf_s("%f", &fah);
        cel = (5.0 / 9.0) * (fah - 32.0);
        printf("%.3f ℉ = %.3f ℃\n", fah, cel);
      }
      else if (sub_menu == 2) {
        printf(" ℃ 를 입력하세요.:");
        scanf_s("%f", &cel);
        fah = (9.0 / 5.0) * cel + 32.0;
        printf("%.3f ℃ = %.3f ℉\n", cel, fah);
      }
    }
  }
  return 0;
```

Day 07,08 - Contents

1. **반복문**
   - for문
   - 중첩 for문
   - while문
   - do~while문
   - break, continue문
2. **예제 코드**

Day7~8 - 1. 반복문

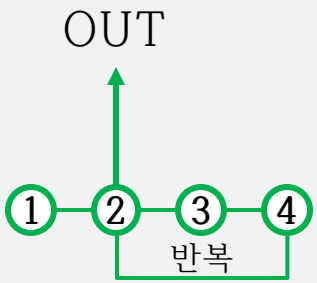
for문

형식 ① ② ⑤ ④ ⑦

```
for(초기값; 조건식; 증감값)
{
  명령문1;
  명령문2;
  명령문3;
  …;
} // 루프(loop)
```

③ ⑥

```
#include <stdio.h>
int main(){
  for (int i = 1; i <= 10; i++){
    printf("%d  ", i);
  }
  printf("\n");
  return 0;
}
```



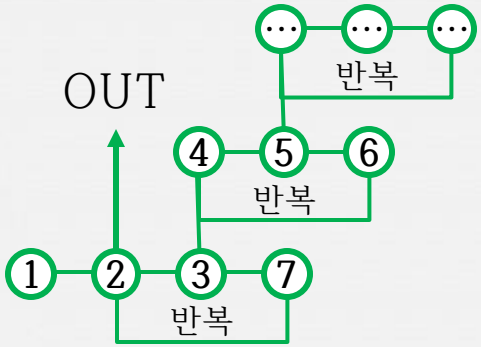
for문은 주어진 조건식이 참이면 중괄호 안을 1번 실행.
증감값 이후 조건식 다시 확인 후 실행 반복
특정 조건에 명령문을 반복적으로 수행하는 제어문으로,
명령어를 반복해서 실행하는 부분을 루프(loop)라고 한다.

Day7~8 - 1. 반복문

이중 for문 (중첩)

형식

```
 for(초기값; 조건식; 증감값)
 {
    for(초기값; 조건식; 증감값)
    {
       명령문1;
       …;
    }
   명령문2;
   …;
 } // 루프(loop)
```




```
#include <stdio.h>
int main(){
    for (int i = 1; i <= 5; i++){ // 줄
        for (int j = 1; j <= 3; j++){ // 칸
            printf("■");
        }
        printf("\n");
    }
    return 0;
}
```

for문 안에 다른 for문이 중첩 된 형태이다.

Day7~8 - 1. 반복문

while문

형식

```
while(조건식)
{
  명령문1;
  명령문2;
  명령문3;
  …;
} // 루프(loop)
```

```
#include <stdio.h>
int main() {
    int i = 1;
    while (i <= 3) { // 줄
        int j = 1;
        while (j <= 5) { //칸
            printf("■");
            j++;
        }
        printf("\n");
        i++;
    }
    return 0;
}
```

조건식이 참인 경우 문장을 반복 실행한다.
while문은 초기값과 증감값을 직접 지정할 수 없으므로 미리 변수의 값을 초기화하고,
while문 내부에 변수의 값을 변화시키는 문장을 삽입해야 한다.

Day7~8 - 1. 반복문

do~while문

형식

```
do
{
  명령문1;
  명령문2;
  명령문3;
  …;
}while(조건식); // 루프(loop)
```

```
#include <stdio.h>
int main(){
    int i = 1, sum = 0;
    do{
        sum += i;
        i++;
    } while (i <= 100);
    printf("1부터~100까지의 합 = %d\n", sum);
    return 0;
}
```

while문과 유사한 형태이지만 명령을 먼저 1회 실행하는 차이가 있다.
1번 실행한 다음 조건을 검사, 조건이 참이면 반복한다.

Day7~8 - 1. 반복문

break문

```
#include <stdio.h>
int main() {
    for (int i = 1; i <= 15; i++) { // 줄
        for (int j = 1; j <= 10 - i; j++) { // 칸
            printf("■");
        }
        printf("\n");
    }
    for (int i = 1; i <= 15; i++) { // 줄
        for (int j = 1; j <= 10 - i; j++) { // 칸
            printf("■");
            if (j == 5)break;
        }
        printf("\n");
    }
    return 0;
}
```

반복 명령의 실행 도중 강제적으로 반복문을 빠져나올 때 사용된다.
여러 개의 루프가 중첩되어 있는 다중 루프에서는 현재 루프만 빠져나온다.

Day7~8 - 1. 반복문

continue문

```
#include <stdio.h>
int main() {
    for (int i = 1; i <= 15; i++) { // 줄
        for (int j = 1; j <= i; j++) { // 칸
            if (5 < j && j < 10) { printf("□"); }
            printf("■");
        }
        printf("\n");
    }
    for (int i = 1; i <= 15; i++) { // 줄
        for (int j = 1; j <= i; j++) { // 칸
            if (5 < j && j < 10) { printf("□"); continue; }
            printf("■");
        }
        printf("\n");
    }
    return 0;
}
```

루프의 나머지 부분을 무시하고 그 반복문의 선두로 다시 이동할 때 사용된다.

Day7~8 - 2. 예제 코드

for문 사용하기

1번 줄은 ★ 1개 2번 줄은 ☆ 2개
3번 줄은 ★ 3개 4번 줄은 ☆ 4개
…
9번 줄은 ★ 9개 10번 줄은 ☆ 10개

```
if(n % 2 == 0) // 짝수
if(n % 2 == 1) // 홀수
```

결과

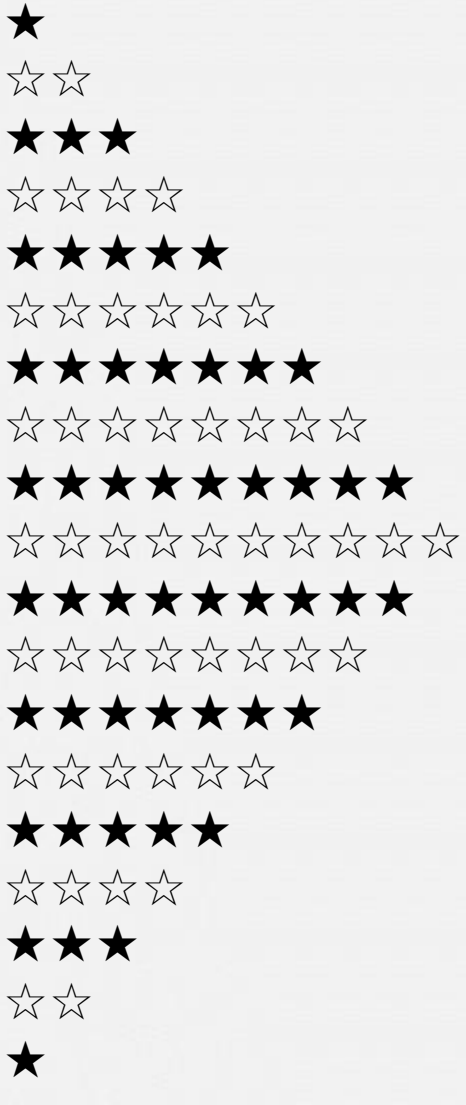
Day7~8 - 2. 예제 코드

```
#include<stdio.h>
int main(){
  int i, j;
  for (i = 1; i <= 10; i++){
    for (j = 1; j <= i; j++){
      if (i % 2 == 1) printf("★");
      else printf("☆");
    }
    printf("\n");
  }
  for (i = 1; i <= 10; i++) {
    for (j = 1; j <= 10-i; j++) {
      printf("  ");
    }
    for (j = 1; j <= i; j++) {
      if (i % 2 == 1) printf("★");
      else printf("☆");
    }
    printf("\n");
  }
  return 0;
}
```

Day7~8 - 2. 예제 코드

for문 사용하기

결과



결과

Day7~8 - 2. 예제 코드

```
#include<stdio.h>
int main() {
  int i, j;
  for (i = 1; i <= 10; i++) {
    for (j = 1; j <= i; j++) {
      if (i % 2 == 1) printf("★");
      else printf("☆");
    }
    printf("\n");
  }
  for (i = 9; i >= 1; i--) {
    for (j = 1; j <= i; j++) {
      if (i % 2 == 1) printf("★");
      else printf("☆");
    }
    printf("\n");
  }
  return 0;
}
```

```
#include<stdio.h>
int main() {
  int i, j;
  for (i = 1; i <= 10; i++) {
    for (j = 1; j <= 10 - i; j++) {
      printf("  ");
    }
    for (j = 1; j <= i; j++) {
      if (i % 2 == 1) printf("★");
      else printf("☆");
    }
    printf("\n");
  }
  for (i = 9; i >= 1; i--) {
    for (j = 1; j <= 10 - i; j++) {
      printf("  ");
    }
    for (j = 1; j <= i; j++) {
      if (i % 2 == 1) printf("★");
      else printf("☆");
    }
    printf("\n");
  }
  return 0;
}
```

Day7~8 - 2. 예제 코드

```
#include<stdio.h>
#include<conio.h>
#include<Windows.h>
int main() {
  CONSOLE_CURSOR_INFO CurInfo;
  CurInfo.dwSize = 1;
  CurInfo.bVisible = FALSE;

SetConsoleCursorInfo(GetStdHandle(STD_OUTPUT
_HANDLE), &CurInfo);
  COORD pos = { 2 * 0,0 };
  int line=1;
  while (1) {
    if (_kbhit()) { // 키보드 입력이 있다면
      int key = _getch(); // 키 값 읽기
      switch (key) { // 키 마다 기능 설정
      case 'z': line++; break;
      case 'x': line--; break;
      case 'q':
      case 'Q': exit(1); break;
      }
      if (line <= 0) line = 1;
      else if (line > 10) line = 10;
    }
SetConsoleCursorPosition(GetStdHandle(STD_OUT
PUT_HANDLE), pos); // 커서 좌표(0,0) 이동
    for (int i = 1; i <= 10; i++) {
      for (int j = 1; j <= i; j++) {
        if (i % line == 0) printf("★");
        else printf("☆");
      }
      printf("\n");
    }
    for (int i = 9; i >= 1; i--) {
      for (int j = 1; j <= i; j++) {
        if (i % line == 0) printf("★");
        else printf("☆");
      }
      printf("\n");
    }
  }
  return 0;
}
```

Day7~8 - 2. 예제 코드

while문 사용하기

제곱근 구하기 (반복 사용)

#include<math.h>
double sqrt(double x); (함수의 원형)
 x값을 입력하면 결과로 제곱근 값 반환
 사용법 double x, y; x = sqrt(y);

제곱근 구할 숫자를 입력
(q/Q) 키 입력시 종료키 설정

출력 화면

제곱근을 구할 숫자를 입력하세요 : 153
153의 제곱근은 12.37 입니다.

계속(아무키) / 종료(Q/q) : q

프로그램을 마칩니다.

Day7~8 - 2. 예제 코드

```
#include <stdio.h>
#include <math.h> //sqrt()
int main(){
    int n, ask;
    while (1){ //무한루프 (for(;;) 같은 의미)
        printf("제곱근을 구할 숫자 입력 : ");
        scanf_s("%d", &n);
        while (getchar() != '\n'); //입력 버퍼 비우기
        if (n < 0){
            printf("0이상 입력하세요.\n");
            continue; // while(1) 반복문 처음으로 이동
        }
        printf("%d의 제곱근은 %.2f입니다.\n", n, sqrt(n));
        printf("계속 (아무키) / 종료 (Q/q) ");
        ask = getchar();
        if (ask == 'Q' || ask == 'q'){
            printf("\n프로그램을 마칩니다.\n");
            break; //while(1) 무한루프 탈출
        }
        printf("\n");
    }
    return 0;
}
```

Day7~8 - 2. 예제 코드

while문 사용하기

화씨에서 섭씨 변환표 만들기

(-100 ℉ ~ 100 ℉ 까지 , 5 ℉ 간격으로)
-100 ℉ = -73.333 ℃
-95 ℉ = -70.556 ℃
-90 ℉ = -67.778 ℃
… = …

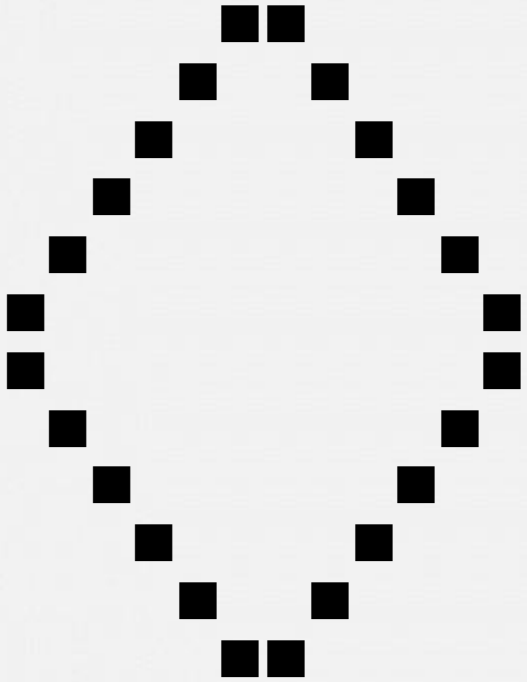
Day7~8 - 2. 예제 코드

```
#include <stdio.h>
#define START -100.0
#define END 100.0
#define DELTA 5
int main(){
    float cel, fah;
    printf("℉를 ℃로 변환 테이블 작성.\n");
    printf("          ℉              ℃\n");
    fah = START;
    while (fah <= END){
        cel = (5.0 / 9.0) * (fah - 32.0);
        printf(" %10.3f ℉     %10.3f ℃\n", fah, cel);
        fah = fah + DELTA;
    }
    return 0;
}
```

Day7~8 - 2. 예제 코드

마름모 그리기 프로그램
1. 대각선의 길이를 입력 하세요.

6 입력 할 경우)



우선순위
- 입력 숫자 *2 = 다이아몬드 총 높이

Day7~8 - 2. 예제 코드

```
#include<stdio.h>
#include<stdlib.h>  //system()
int main() {
    int n;
    while (1) {
        system("cls");
        printf("마름모  그리기 ( 종료 : -1 )\n");
        printf("대각선  높이 : ");
        scanf_s("%d",  &n);
        while (getchar() != '\n');
        if (n == -1) break;
        for (int i = 1; i <= n; i++) {
            for (int j = 1; j <= n - i; j++) printf("  ");
            printf("■");
            for (int j = 1; j <= 2 * (i - 1); j++) printf("  ");
            printf("■\n");
        }
        for (int i = n; i >= 1; i--) {
            for (int j = 1; j <= n - i; j++)printf("  ");
            printf("■");
            for (int j = 1; j <= 2 * (i - 1); j++)printf("  ");
            printf("■\n");
        }
        system("pause");
    }
    printf("\n\t\t 프로그램 종료 합니다.\n");
    return 0;
}
```

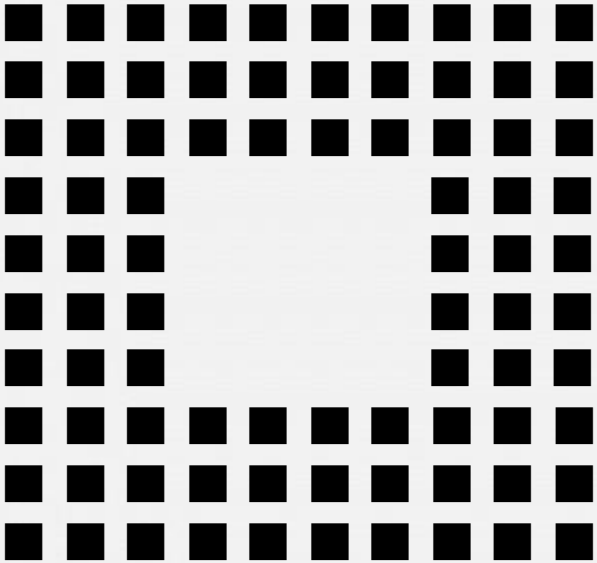
Day7~8 - 2. 예제 코드

```
#include<stdio.h>
#include<stdlib.h>
int main() {
  int n, thick;
  while (1) {
    system("cls");
    printf("마름모 그리기 ( 종료 : -1 )\n");
    printf("대각선 높이 : ");
    scanf_s("%d", &n);
    while (getchar() != '\n');
    if (n == -1) break;
    printf("두께 : ");
    scanf_s("%d", &thick);
    while (getchar() != '\n');
    if (thick == -1) break;
    for (int i = 1; i <= n; i++) {
      for (int j = 1; j <= n - i; j++) printf("  ");
      for (int j = 0; j < thick; j++) printf("■");
      for (int j = 1; j <= 2 * (i - 1) && i <=
thick; j++) printf("■");
      for (int j = 1; j <= 2 * (i - 1) && i >
thick; j++) printf("  ");
      for (int j = 0; j < thick; j++) printf("■");
      printf("\n");
    }
    for (int i = n; i >= 1; i--) {
      for (int j = 1; j <= n - i; j++)printf("  ");
      for (int j = 0; j < thick; j++) printf("■");
      for (int j = 1; j <= 2 * (i - 1) && i <=
thick; j++) printf("■");
      for (int j = 1; j <= 2 * (i - 1) && i >
thick; j++) printf("  ");
      for (int j = 0; j < thick; j++) printf("■");
      printf("\n");
    }
    system("pause");
  }
  printf("\n\t\t 프로그램 종료 합니다.\n");
  return 0;
}
```

Day7~8 - 2. 예제 코드

직사각형 그리기 프로그램
0을 입력하면 종료합니다.
1. 가로, 세로, 두께를 입력 하세요.

10 10 3 입력 할 경우)



우선순위
- 가로, 세로의 크기는 변하지 않을 것
- 두께에 따라 내부 공백의 크기 변경
- 0을 입력하면 종료 할 것

Day7~8 - 2. 예제 코드

```
#include<stdio.h>
#include<stdlib.h>
int main() {
    int width, height, thick;
    while (1) {
        printf("직사각형 그리기 프로그램\n");
        printf("0을 입력하면 종료합니다.\n");
        printf("1. 가로, 세로, 두께를 입력 하세요. ");
        scanf_s("%d %d %d", &width, &height, &thick);
        while (getchar() != '\n');
        printf("\n");
        if (width < 0 || height < 0 || thick < 0) {
            printf("가로, 세로, 두께는 양수이어야 합니다.\n");
            continue;
        }
        else if (width == 0 || height == 0 || thick == 0) {
            printf("프로그램 종료합니다. END...\n");
            break;
        }
        else if (height < thick * 2 || width < thick * 2) {
            for (int i = 1; i <= height; i++) {
                for (int j = 1; j <= width; j++) printf("■");
                printf("\n");
            }
        }
        else {
            for (int i = 1; i <= thick; i++) {
                //윗변
                for (int j = 1; j <= width; j++)printf("■");
                printf("\n");
            }
            for (int i = 1; i <= height - thick * 2; i++) {
                //좌변
                for (int j = 1; j <= thick; j++) printf("■");
                //공백 구간
                for (int j = 1; j <= width - thick * 2; j++) printf("  ");
                //우변
                for (int j = 1; j <= thick; j++) printf("■");
                printf("\n");
            }
            for (int i = 1; i <= thick; i++) {
                //밑변
                for (int j = 1; j <= width; j++)printf("■");
                printf("\n");
            }
            printf("\n");
        }
    }
    return 0;
}
```

Day7~8 - 2. 예제 코드

```
#include<stdio.h>
int main() {
  int width, height, thick;
  while (1) {
    printf("직사각형 그리기 프로그램\n");
    printf("0을 입력하면 종료합니다.\n");
    printf("1. 가로, 세로, 두께를 입력 하세요. ");
    scanf_s("%d %d %d", &width, &height, &thick);
    while (getchar() != '\n');
    printf("\n");
    if (width < 0 || height < 0 || thick < 0) {
      printf("가로, 세로, 두께는 양수이어야 합니다.\n");
      continue;
    }
    else if (width == 0 || height == 0 || thick == 0) {
      printf("프로그램 종료합니다. END...\n");
      break;
    }
    else {
      for (int i = 1; i <= height; i++) {
        for (int j = 1; j <= width; j++) {
          if (thick < j && j <= width - thick && thick < i && i <= height - thick) {
            printf("  ");
            continue;
          }
          printf("■");
        }
        printf("\n");
      }
    }
  }
  return 0;
}
```

Day 09 - Contents

1. 난수 (Random 제어문 활용)
   - 난수 정의
   - 의사 난수
   - 중앙제곱법
   - 선형합동법
2. 예제 코드

Day09 - 1. 난수 (Random Number)

난수(Random Number)

정의

특정 주기, 규칙이 없이 무작위로 연속적인 임의의 수
무작위 : (통계적 균일 분포) 모든 숫자가 같은 확률 생성
예측 불가능 : 다음 수의 순서를 알 수 없어야 한다.
재현 불가능 : 반복하지 않는 주기를 가져야한다.

의사난수(PRN : Pseudo-Random Number)

특정 주기를 가지고 만들어진 임의의 수
주기가 길수록 진짜 난수에 가까운 값을 생성한다 (무작위성이 높다)

Day09 - 1. 난수 (Random Number)

난수(Random Number)

중앙제곱법 (의사난수)

- 시드(seed) 매칭
- 초기값을 제곱 xxxxxxxx
- 중앙값 추출 xxXXXXxx
- 반복

						7	4	1	9	6	**3**
5	5	0	5	0	9	0	9	3	3	6	**9**
						0	9	0	9	3	**3**
		8	2	6	8	8	1	0	4	8	**9**
						6	8	8	1	0	**4**
4	7	3	4	8	7	1	1	4	8	1	**6**
						8	7	1	1	4	**8**
x	x	x	x	x	x	x	x	x	x	x	**x**
						x	x	x	x	x	**x**

초기 시드값에 제곱

중앙값 추출

제곱

추출

제곱

추출

제곱

추출 (반복)

제곱 (반복)

Day09 - 2. 예제 코드

```
#include <stdio.h>
int main(void) {
    long long front = 1e8;/* 1 X 10^8 */
    long long rear = 1e2;/* 1 X 10^2 */
    long long seed = 741963, middle;
    for (int i = 1; i <= 100; i++) {
        middle = seed * seed;
        middle = middle % front;
        middle = middle / rear;
        printf("%20lld\n", middle);
        seed = middle;
    }
    return 0;
}
```

Day09 - 1. 난수 (Random Number)

난수(Random Number)

선형합동법

어떤 두 수를 x로 나눴을 때 나머지가 같은 수를 x에 대한 합동이다.

17 과 29 는 12 (x) 에 합동이다.

4,561 과 995,857,555 는 218 (x) 에 합동이다.

주기가 길수록 (x가 클수록) 무작위성이 강한 의사난수를 만들수 있다.

						7	4	9	**6**
					5	9	9	6	**8**
					5	9	9	9	**5**
						9	9	9	**5**
					7	9	9	6	**0**
					7	9	9	8	**7**
						9	9	8	**7**
					x	x	x	x	**x**
						x	x	x	**x**

$$R_{n+1} = (A \times R_n + C) \bmod M$$

R_0 = Seed

A, C, M은 0보다 큰 정수

A < M, C < M, R_0<M

R_0 = 7496

A = 8

C = 27

M = 10000

A < M, C < M

Day09 - 2. 예제 코드

```
#include <stdio.h>
int main() {
  int r0 = 7496, A = 8, C = 27, M = 10000;
  for (int i = 1; i < 100; i++) {
    printf("%6d ", r0);
    if (i % 10 == 0)printf("\n");
    r0 = (A * r0 + C) % M;
  }
  return 0;
}
```

난수(Random Number)

난수 생성 함수
#include <stdlib.h>
#include <time.h>

int rand (void); (함수의 원형)
seed(고정)값에 따라 0~32767 값을 랜덤 반환

(void) srand (unsigned int seed); (함수의 원형)
seed값을 변경시킨다.

(long long) time (time_t * timer);
매개변수가 NULL일 때에는 실행한 현재 시간을 반환

사용법
srand((unsigned int)time(NULL));
rand();
결론 : 프로그램 실행 시간에 따라 랜덤 변수 발생

Day09 - 2. 예제 코드

```
#include <stdio.h>
#include <stdlib.h>// srand(), rand()
#include <time.h> // time()
#include <Windows.h> // Sleep()
int main() {
  srand((unsigned int)time(NULL)); // seed 초기화
  for (int i = 1; i < 100; i++) {
    printf("%6d %10d \n", rand(),(unsigned
int)time(NULL));
    Sleep(1000); // 1초의 delay
  }
  return 0;
}
```

가위 바위 보

(가위 : 1, 바위 : 2, 보 : 3)

사용자로부터 입력 (입력 받은 값 출력)

랜덤 변수 생성 (컴퓨터 값 출력)

비교 : 가위 < 바위 < 보 < 가위

승리자 결과 출력

Day09 - 2. 예제 코드

```
#include <stdio.h>
#include <stdlib.h>// srand(), rand()
#include <time.h> // time()
#define 가위 1
#define 바위 2
#define 보 3
#define computer 0
#define user 1
#define draw 2
int main() {
    int in_user, victory = 0;
    srand((unsigned int)time(NULL));
    while (1) {
        printf("\t [ 가위 바위 보 ] \n");
        printf("1. 가위 \n");
        printf("2. 바위 \n");
        printf("3. 보 \n");
        printf(" 1~3번을 입력하세요. (가위 : 1, 바위 : 2, 보 : 3) :  ");
        scanf_s("%d", &in_user);
        while (getchar() != '\n');
        int in_com = rand() % 3 + 1; // 가위 : 1, 바위 : 2, 보 : 3

        printf("User : %s\n", (in_user == 가위 ? "가위" : in_user == 바위 ? "바위" : "보"));
        printf("Computer : %s\n", (in_com == 가위 ? "가위" : in_com == 바위 ? "바위" : "보"));

        if (in_user == in_com) printf(" 무승부 ! \n");
        else if (in_user - (in_com % 3) == 1 ) printf(" 승리자 : User \n");
        else printf(" 승리자 : Computer \n");

        system("pause");
    }
    return 0;
}
```

Day 10 - Contents

1. **일차원 배열**
   - 배열 선언과 배열 형식, 구조
   - 배열의 초기화
   - 최대값, 최소값 구하는 알고리즘
2. **예제 코드**

Day10 - 1. 일차원 배열

배열(array)

정의
기억장치에 같은 자료형의 값들을 연속적으로 저장하는 것
자료들은 요소라 한다.

형식
  자료형 배열의 이름[원소의 수];
  예시) int ary[4]; ⟶

구조

메모리 주소	배열 순서	예제 값
(배열명이 주소) ary	int ary[0]	11
ary+1	int ary[1]	12
ary+2	int ary[2]	13
ary+3	int ary[3]	14
…	…	…

```
#include <stdio.h>
int main() {
    int ary[4] = { 11, 12, 13, 14 };
    for (int i = 0; i < 4; i++) {
        printf("ary : %p, ary[%d] : %d\n", ary + i , i , ary[i]);
    }
    return 0;
}
```

Day10 - 2. 예제 코드

```
#include <stdio.h>
int main()
{
    int score[5];
    int i, tot = 0;
    float ave = 0;   score 배열의 묶음 개수 = score 전체 크기 / score 자료 1개 크기
    for (i = 0; i < sizeof(score) / sizeof(score[0]); i++) {
        printf("%d번 학생의 성적을 입력하세요 : ", i + 1);
        //scanf_s("%d", &score[i]);
        scanf_s("%d", score+i);
        tot += score[i];
    }
    ave = (float)tot / (sizeof(score) / sizeof(score[0]));
    printf("\n총점은 %d점이고 평균은 %.2f점입니다.\n", tot, ave);
    return 0;
}
```

sizeof(score)/sizeof(score[0])

- sizeof를 사용하여 배열의 크기를 구할 수 있다
- 배열의 크기를 수정 할 때, score[5]의 요소 값만 변경하면 된다.

Day10 - 1. 일차원 배열

배열의 초기화

정의

(1) 선언과 동시에 초기 값을 중괄호({})안에 차례대로 입력.
(2) 배열의 크기를 생략하고 초기 값을 입력 할 경우
    초기 값의 개수가 배열의 크기가 된다.
(3) 배열의 크기보다 적은 개수의 초기 값을 할당 할 경우
    뒤의 값은 무조건 0으로 초기화.
(4) static 선언 시 모든 배열의 요소는 0으로 초기화.
    //static은 프로그램 시작과 동시에 초기화
(5) 배열의 크기보다 많은 초기 값을 설정하면 에러가 발생

형식

(1) int a[4]={11,12,13,14};

(2) int b[]={1,2,3,4,5};

(3) int c[6]={1};

(4) static int d[7];

(5) int e[2]={11,12,13,14}; **Error**

	a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]
(1)	11	12	13	14			
(2)	1	2	3	4	5		
(3)	1	0	0	0	0	0	
(4)	0	0	0	0	0	0	0
(5)	11	12	13	14			

Day10 - 1. 일차원 배열

```
#include <stdio.h>
int main()
{
  int a[4]={11,12,13,14};
  int b[]={1,2,3,4,5};
  int c[6]={1};
  static int d[7];
  //int e[2]={11,12,13,14};

  printf("%d %d %d %d\n",a[0],a[1],a[2],a[3]);
  printf("%d %d %d %d %d\n",b[0],b[1],b[2],b[3],b[4]);
  printf("%d %d %d %d %d %d\n",c[0],c[1],c[2],c[3],c[4],c[5]);
  printf("%d %d %d %d %d %d %d\n",d[0],d[1],d[2],d[3],d[4],d[5],d[6]);
  //printf("%d %d %d %d\n",e[0],e[1],e[2],e[3]);
        return 0;
}
```

	a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]
(1)	11	12	13	14			
(2)	1	2	3	4	5		
(3)	1	0	0	0	0	0	
(4)	0	0	0	0	0	0	0
(5)	11	12	13	14			

Day10 - 2. 예제 코드

```
#include <stdio.h>
int main()
{
    int days[13] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
    int i, month, day, total=0;

    printf("해당 일까지 날짜 계산 \n");
    printf("월 : ");
    scanf_s("%d", &month);
    printf("일 : ");
    scanf_s("%d", &day);

    for (i = 1; i < month; i++) total += days[i];
    total += day;
    printf("\n1월1일 ~ %d월%d일 : %d일\n", month, day, total);
    return 0;
}
```

Day10 - 2. 예제 코드

최대값 구하기

1차원 배열 10칸 메모리 공간 할당

각 방에 랜덤 변수 생성 (1 ~ 500)

기준 값([0]번) 1개를 설정, 각 방의 값을 비교하며 큰 값 저장 [최대값]
기준 값([0]번) 1개를 설정, 각 방의 값을 비교하며 작은 값 저장 [최소값]

결과 출력

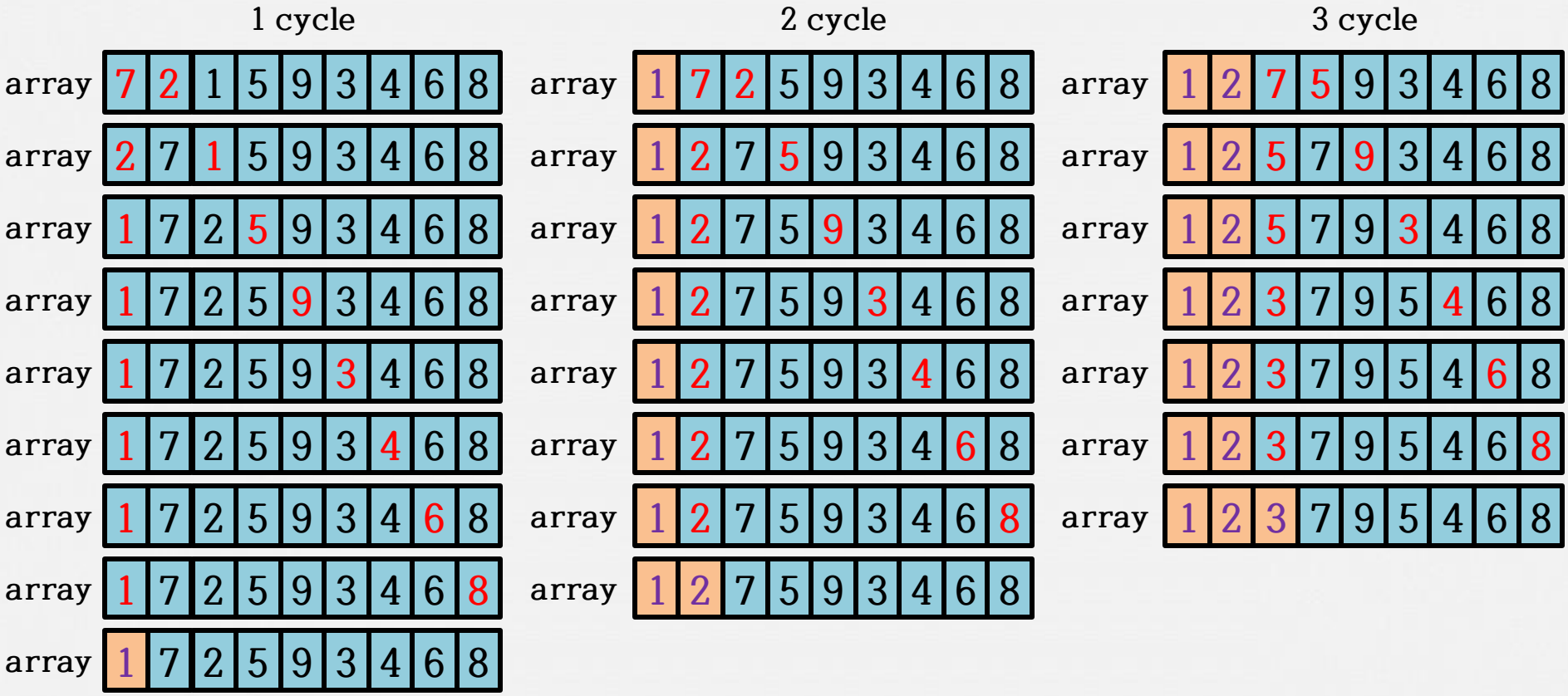
10개의 배열 데이터를 정렬하기

순차 정렬, 선택 정렬

정렬 후 결과 출력

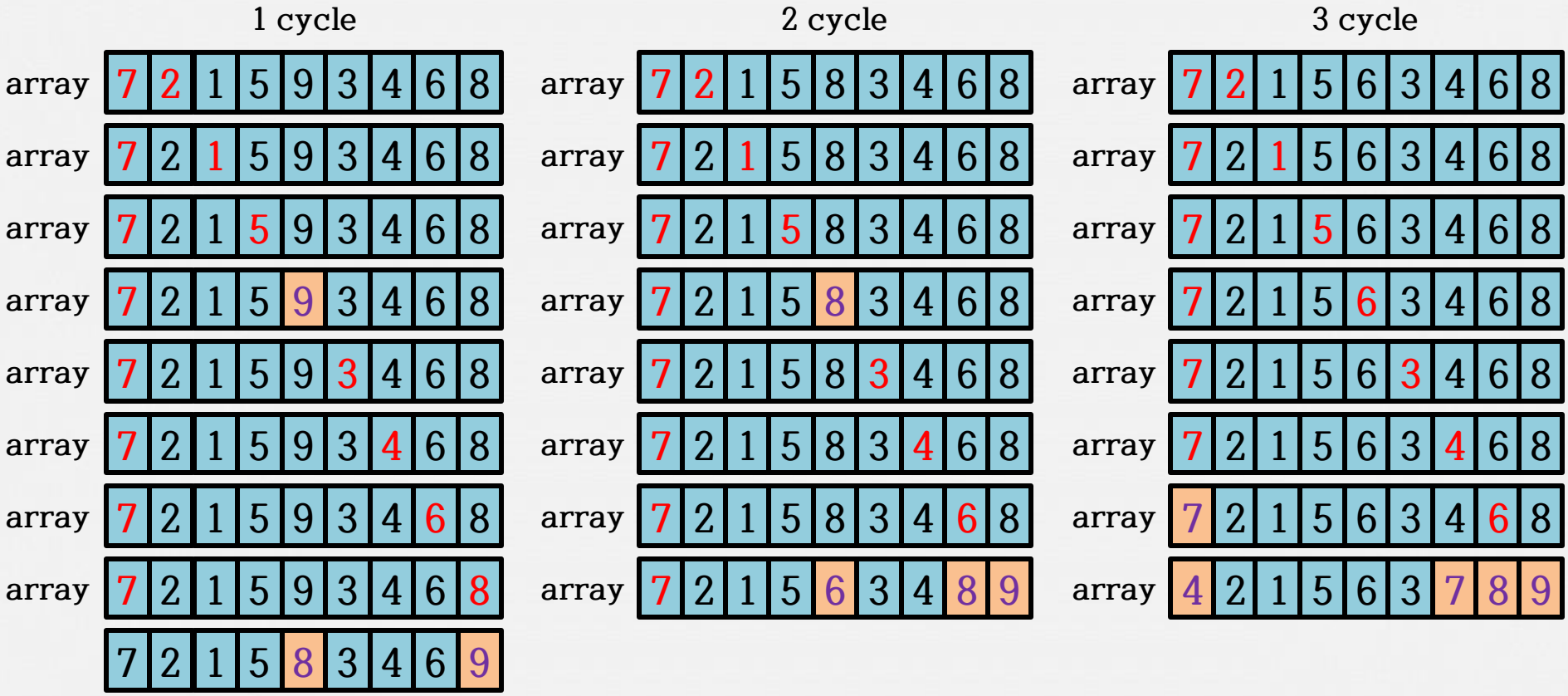
Day10 - 2. 예제 코드

순차정렬
맨 앞에서부터 제일 작은 원소를 배치하게 만들어 나가는 알고리즘
배치할 자리에 있는 원소를 뒤쪽에 있는 원소들과 비교
작은 것을 발견하면 배치할 위치의 원소와 교환
위 과정을 n-1번 반복, 정렬 수행
1회전을 수행하고 나면 가장 작은 값의 자료가 맨 앞에 위치한다.
다음 회전에서는 두 번째 자료를 가지고 비교 마찬가지로 3회전에서는 세 번째 자료를 정렬, …



Day10 - 2. 예제 코드

선택정렬
제일 큰 값을 찾아 맨 뒤의 요소와 교체하는 방법을 반복하여 전체를 정렬하는 알고리즘
(제일 작은 값을 찾아 맨 앞의 요소와 교체하는 방법을 반복할 수도 있습니다.)
위 과정을 n-1번 반복, 정렬 수행
1회전을 수행하고 나면 가장 큰 값과 맨 뒤의 자료를 교체한다.
다음 회전에서는 두 번째 큰 값을 교체, 3회전에서는 세 번째 큰 값을 교체 정렬, …



Day10 - 2. 예제 코드

```
#include <stdio.h>
#include <stdlib.h>// srand(), rand()
#include <time.h> // time()
#define swap(type,a,b) {type temp=a;a=b;b=temp;}
int main() {
  int value[10] = { 0 };
  srand((unsigned int)time(NULL));
  for (int i = 0; i < 10; i++) {
    value[i] = rand() % 500 + 1;
    printf("value[%02d] = %5d \n", i, value[i]);
  }
  int MAX = value[0], MIN = value[0];
  for (int i = 1; i < 10; i++) {
    if (MAX < value[i]) MAX = value[i];
    if (MIN > value[i]) MIN = value[i];
  }
  printf("최대값 : %5d \n", MAX);
  printf("최소값 : %5d \n", MIN);

  for (int i = 0; i < 10; i++) {
    for (int j = i + 1; j < 10; j++) {
      if (value[i] > value[j]) swap(int, value[i], value[j]);
    }
    for (int j = 0; j < 10; j++) {
      printf("%3d ", value[j]);
    }
    printf("\n");
  }

  for (int i = 0; i < 10; i++) {
    value[i] = rand() % 500 + 1;
    printf("value[%02d] = %5d \n", i, value[i]);
  }
  MAX = value[0], MIN = value[0];
  for (int i = 1; i < 10; i++) {
    if (MAX < value[i]) MAX = value[i];
    if (MIN > value[i]) MIN = value[i];
  }
  printf("최대값 : %5d \n", MAX);
  printf("최소값 : %5d \n", MIN);

  for (int i = 9; i >= 0; i--) {
    MAX = 0;
    for (int j = 0; j <= i; j++) {
      if (value[MAX] < value[j]) MAX = j;
    }
    swap(int, value[MAX], value[i]);
    for (int j = 0; j < 10; j++) {
      printf("%3d ", value[j]);
    }
    printf("\n");
  }

  return 0;
}
```

Day10 - 2. 예제 코드

홀수, 짝수 개수 세기

1차원 배열 100칸 메모리 공간 할당
배열 칸 당 랜덤 변수 생성 (1 ~ 1000)
각 칸의 값이 홀수 & 짝수 판별 -> 홀수, 짝수 수량 파악

홀수 값을 오름차순 정렬 (순차정렬)
짝수 값을 오름차순 정렬 (선택정렬)

결과 출력

Day10 - 2. 예제 코드

```
#include <stdio.h>
#include <stdlib.h>// srand(),rand()
#include <time.h> // time()
#define COUNT 100
#define RANGE 1000
#define swap(type,a,b) {type temp=a;a=b;b=temp;}
int main() {
    int value[COUNT] = { 0 };
    srand((unsigned int)time(NULL));
    for (int i = 0; i < COUNT; i++) {
        value[i] = rand() % RANGE + 1;
        printf("value[%05d] = %5d \n", i, value[i]);
    }

    int odd_ary[COUNT] = { 0 },even_ary[COUNT] = { 0 };
    int odd = 0, even=0;
    for (int i = 0; i < COUNT; i++) {
        if(value[i] %2 == 0) even_ary[even++] = value[i];
        else odd_ary[odd++] = value[i];
    }

    for (int i = 0; i < odd; i++) {
        for (int j = i + 1; j < odd; j++) {
            if (odd_ary[i] > odd_ary[j]) swap(int, odd_ary[i], odd_ary[j]);
        }
        for (int j = 0; j < odd; j++) {
            printf("%6d ", odd_ary[j]);
        }
        printf("\n");
    }

    for (int i = even-1; i >= 0; i--) {
        int MAX = 0;
        for (int j = 0; j <= i; j++) {
            if (even_ary[MAX] < even_ary[j]) MAX = j;
        }
        swap(int, even_ary[MAX], even_ary[i]);
        for (int j = 0; j < even; j++) {
            printf("%6d ", even_ary[j]);
        }
        printf("\n");
    }

    printf("홀수 목록: ");
    for (int j = 0; j < odd; j++) {
        printf("%6d ", odd_ary[j]);
    }
    printf("\n");

    printf("짝수 목록: ");
    for (int j = 0; j < even; j++) {
        printf("%6d ", even_ary[j]);
    }
    printf("\n");

    return 0;
}
```

Day10 - 2. 예제 코드

```
#include <stdio.h>
#include <stdlib.h>// srand(),rand()
#include <time.h> // time()
#define COUNT 10
#define RANGE 1000
#define swap(type,a,b) {type temp=a;a=b;b=temp;}
int main() {
    int value[COUNT] = { 0 };
    int odd = 0, even = COUNT, number ;
    srand((unsigned int)time(NULL));

    for (int i = 0; i < COUNT; i++) {
        number = rand() % RANGE + 1;
        if (number % 2 == 0) value[--even] = number;
        else value[odd++] = number;
    }

    for (int i = 0; i < COUNT; i++) {
        printf("value[%05d] = %5d \n", i, value[i]);
    }

    for (int i = 0; i < odd; i++) {
        for (int j = i + 1; j < odd; j++) {
            if (value[i] > value[j]) swap(int, value[i], value[j]);
        }
        for (int j = 0; j < odd; j++) {
            printf("%6d ", value[j]);
        }
        printf("\n");
    }

    for (int i = COUNT-1; i > even; i--) {
        int MAX = 0;
        for (int j = 0; j <= i; j++) {}
            if (value[MAX] < value[j]) MAX = j;
        }
        swap(int, value[MAX], value[i]);
        for (int j = even; j < COUNT; j++) {
            printf("%6d ", value[j]);
        }
        printf("\n");
    }

    printf("홀수 목록: ");
    for (int j = 0; j < odd; j++) {
        printf("%6d ", value[j]);
    }
    printf("\n");

    printf("짝수 목록: ");
    for (int j = even; j < COUNT; j++) {
        printf("%6d ", value[j]);
    }
    printf("\n");

    return 0;
```

Day 11 - Contents

1. **다차원 배열**
   - 배열 선언과 배열 형식, 구조
   - 배열의 초기화
   - 구구단 출력
2. **예제 코드**

Day11 - 1. 다차원 배열

다차원 배열

형식

자료형 배열 이름[크기][크기][…];

예시

int ary[3][3]={8, 2, 4, 1, 5, 6};

	a[0]	a[1]	a[2]
a[0]	8	2	4
a[1]	1	5	6
a[2]	0	0	0

	a[0]	a[1]	a[2]
a[0]	a[0][0]	a[0][1]	a[0][2]
a[1]	a[1][0]	a[1][1]	a[1][2]
a[2]	a[2][0]	a[2][1]	a[2][2]

구조

메모리 주소	배열 순서	예제 값
(배열명이 주소) ary	int ary[0][0]	8
ary+1	int ary[0][1]	2
ary+2	int ary[0][2]	4
ary+3	int ary[1][0]	1
ary+4	int ary[1][1]	5
ary+5	int ary[1][2]	6
ary+6	int ary[2][0]	0
ary+7	int ary[2][1]	0
ary+8	int ary[2][2]	0
…	…	…

Day11 - 1. 다차원 배열

초기화 예시

(1) int a[2][2]={11,12,13,14};
(2) int b[3][2]={{3},{0,7},{4}};
(3) int c[2][3]={1};
(4) static int d[2][2];

결과

(1)	a[0]	a[1]
a[0]	11	12
a[1]	13	14

(2)	a[0]	a[1]
a[0]	3	0
a[1]	0	7
a[2]	4	0

(3)	a[0]	a[1]	a[2]
a[0]	1	0	0
a[1]	0	0	0

(4)	a[0]	a[1]
a[0]	0	0
a[1]	0	0

```
#include <stdio.h>
int main()
{
    int a[2][2] = { 11,12,13,14 };
    int b[3][2] = { {3},{0,7},{4} };
    int c[2][3] = { 1 };
    static int d[2][2];

    printf("{ %d %d } { %d %d }\n", a[0][0], a[0][1], a[1][0], a[1][1]);
    printf("{ %d %d } { %d %d } { %d %d }\n", b[0][0], b[0][1], b[1][0], b[1][1], b[2][0], b[2][1]);
    printf("{ %d %d %d } { %d %d %d }\n", c[0][0], c[0][1], c[0][2], c[1][0], c[1][1], c[1][2]);
    printf("{ %d %d } { %d %d }\n", d[0][0], d[0][1], d[1][0], d[1][1]);
    return 0;
}
```

Day11 - 1. 다차원 배열

구구단 출력 (1단 ~ 9단)

- 다차원 배열 9 x 9 를 선언
- 구구단 결과 값 입력
- 1단 ~ 9단 출력 (오름차순)
- 9단 ~ 1단 출력 (내림차순)

결과

오름차순

1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	21	24	27
…	…	…	…	…	…	…	…	…
9	18	27	36	45	54	63	72	81

내림차순

9	18	27	36	45	54	63	72	81
8	16	24	32	40	48	56	64	72
7	14	21	28	35	42	49	56	63
…	…	…	…	…	…	…	…	…
1	2	3	4	5	6	7	8	9

Day11 - 1. 다차원 배열

```
#include <stdio.h>
int main() {
    int i, j, times_table[9][9];
    printf("1~9단 구구단 결과 값 출력 \n\n");
    printf("오름차순\n");
    for (i = 0; i < 9; i++) {
        for (j = 0; j < sizeof(times_table)/  sizeof(times_table[0]);  j++) {
            times_table[i][j] = (i + 1) * (j + 1);
            printf(" | %3d | ", times_table[i][j]);
        }
        printf("\n");
    }
    printf("\n내림차순\n");
    for (i = sizeof(times_table)  / sizeof(times_table[0])-1;  i >= 0; i--) {
        for (j = 0; j < 9; j++) {
            printf(" | %3d | ", times_table[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

Day11 - 1. 다차원 배열

5x5 배열에 1~10사이의 랜덤한 숫자를 저장한 후,
6열에는 가로의 합을 6행에는 세로의 합을 각각 저장한다.
가장 마지막 값은 대각선의 합을 저장한다.

결과창
5x5 배열,
6x6 배열 출력

난수 생성 함수
#include <stdlib.h>
#include <time.h>
rand()
srand(time(NULL));

4	8	3	7	5
5	4	3	6	8
2	2	5	6	9
3	9	9	7	2
4	4	6	2	7

4	8	3	7	5	27
5	4	3	6	8	26
2	2	5	6	9	24
3	9	9	7	2	30
4	4	6	2	7	23
18	27	26	28	31	27

Day11 - 1. 다차원 배열

```
#include<stdio.h>
#include<stdlib.h>  //rand() : 랜덤 함수,
srand(time(NULL));  : 랜덤 함수 seed 변경
#include<time.h>//time()
int main() {
    int ary[6][6] = { 0 }, sum = 0;
    srand((unsigned int)time(NULL));
    for (int i = 0; i < 5; i++) {
        for (int j = 0; j < 5; j++) {
            ary[i][j] = rand() % 10 + 1; // 난수 생성
            sum = ary[i][j] + sum;
            printf("%5d", ary[i][j]); // 출력
        }
        ary[i][5] = sum;
        sum = 0;
        printf("\n");
    }//6열 계산 완료
    printf("\n");
    for (int i = 0; i < 6; i++) {
        for (int j = 0; j < 6; j++) {
            printf("%5d", ary[i][j]); // 출력
        }
        printf("\n");
    }
    printf("\n");
    for (int i = 0; i < 5; i++) {
        for (int j = 0; j < 5; j++) {
            sum = ary[j][i] + sum;
        }
        ary[5][i] = sum;
        sum = 0;
    }//6행 계산 완료
    for (int i = 0; i < 6; i++) {
        for (int j = 0; j < 6; j++) {
            printf("%5d", ary[i][j]); // 출력
        }
        printf("\n");
    }
    printf("\n");
    for (int i = 0; i < 5; i++) {
        sum = ary[i][i] + sum;
    }// (6, 6) 계산 완료
    ary[5][5] = sum;
    sum = 0;
    printf("\n");
    for (int i = 0; i < 6; i++) {
        for (int j = 0; j < 6; j++) {
            printf("%5d", ary[i][j]); // 출력
        }
        printf("\n");
    }
    return 0;
}
```

Day11 - 1. 다차원 배열

50명 학생 별 성적 데이터 관리 프로그램 작성

- 50명 학생 별 국어, 수학, 영어, 탐구 성적
  * 50명 , 4과목을 저장 할 2차원 배열 선언
  * 1명 당 4과목 성적 데이터 랜덤 생성 & 대입 (1~100점 범위)

- 정렬 할 성적의 과목 입력 받기
  1. 국어
  2. 수학
  3. 영어
  4. 탐구

- 입력 받은 과목 성적 순위 정렬(내림차순) & 결과 출력

결과
선택과목 내림차순

정렬 전	1	2	3	4	5	6	7	…	50
	42	70	63	82	96	92	93	…	11
정렬 후	5	7	6	19	17	31	4	…	40
	96	93	92	91	89	87	82	…	1

Day11 - 1. 다차원 배열

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#define student 50
#define subject 4
#define 국어 0
#define 수학 1
#define 영어 2
#define 탐구 3
#define swap(type,a,b) {type temp=a;a=b;b=temp;}
int main() {
  srand((unsigned int)time(NULL));
  int score[student][subject] = { 0 };
  for (int i = 0; i < student; i++) {
    for (int j = 0; j < subject; j++) {
      score[i][j] = rand() % 100 + 1;
    }
  }
  printf(" %15s %15s %4s %4s %4s \n", "[ 학생 성적 ]","국어", "수학", "영어", "탐구");
  printf("------------------------------------------------------------\n");
  for (int i = 0; i < student; i++) {
    printf("%3d번 학생 성적 입니다 -> ", i + 1);
    for (int j = 0; j < subject; j++) {
      printf("%4d ", score[i][j]);
    }
    printf("\n");
  }
  int rank[2][student], menu;
  printf("성적 기준 정렬\n");
  printf("0. 국어\n");
  printf("1. 수학\n");
  printf("2. 영어\n");
  printf("3. 탐구\n");
  scanf_s("%d", &menu);
  while (getchar() != '\n');
  for (int i = 0; i < student; i++) {
    rank[0][i] = i+1;
    rank[1][i] = score[i][menu];
  }
  for (int i = 0; i < student; i++) {
    printf(" %4d", rank[0][i]);
  }
  printf("\n");
  for (int i = 0; i < student; i++) {
    printf(" %4d", rank[1][i]);
  }
  printf("\n");
  for (int i = student - 1; i >= 0; i--)    {
    int MAX = 0;
    for (int j = 0; j <= i; j++) {
      if (rank[1][MAX] > rank[1][j])
        MAX = j;
    }
    swap(int, rank[0][MAX], rank[0][i]);
    swap(int, rank[1][MAX], rank[1][i]);
  }
  switch (menu) {
  case 국어: printf(" 국어 성적 기준 \n"); break;
  case 수학: printf(" 수학 성적 기준 \n"); break;
  case 영어: printf(" 영어 성적 기준 \n"); break;
  case 탐구: printf(" 탐구 성적 기준 \n"); break;
  }
  for (int i = 0; i < student; i++) {
    printf(" %4d", rank[0][i]);
  }
  printf("\n");
  for (int i = 0; i < student; i++) {
    printf(" %4d", rank[1][i]);
  }
printf("\n");
  return 0;
}
```

Day 12 - Contents

1. **함수**
   - 함수의 정의, 선언, 호출
   - 인자 전달 방법 및 리턴 값 확인하기
2. **예제 코드**

Day12 - 1. 함수

함수의 정의, 선언, 호출

특징

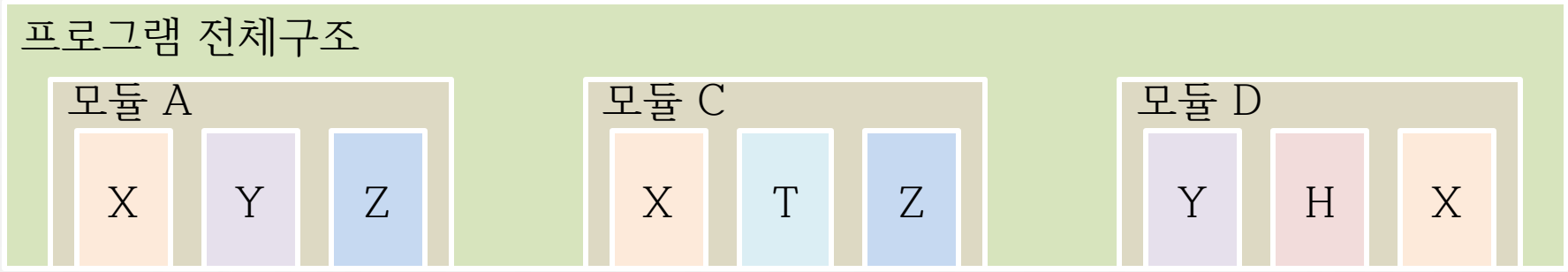
- C언어는 대부분이 함수들의 집합
- 함수는 반복되는 작업을 간소화 (가독성 ↑)
- 프로그램을 기능, 순서, 효율별로 구분하기 위해 사용된다.

표준함수

- C언어에서 자체적으로 제공하는 함수
- 헤더 파일을 #include < >를 활용해 바로 사용 할 수 있다.

사용자 정의 함수

- 전체 프로그램을 짧은 길이의 단위 프로그램으로 나누어 정의함
- 유지 보수가 효율적이라는 장점
- 모듈화된 구조(Modulization, hierarchy)



Day12 - 1. 함수

표준함수

헤더파일	분류	함수 예
stdio.h	입, 출력	printf, scanf, getchar, …
	파일	fopen, fclose, fprintf, …
conio.h	콘솔 입, 출력	putch, cputs, cprintf, …
string.h	문자열 처리	strcat, strcmp, strcpy, …
math.h	수학	sqrt, sin, cos, tan, log, exp, pow, …
ctype.h	문자 형태 판별	isalpha, isdigit, islower, …
	문자 변환	tolower, toupper, …
stdlib.h	수치 변환	atoi, atoa
	난수 관련	rand, srand
	정렬, 검색	qsqrt, lfind, …
time.h	날짜, 시간	clock, ctime, localtime, mktime, …
…	…	…

https://www.ibm.com/docs/ko/i/7.3?topic=extensions-standard-c-library-functions-table-by-name

Day12 - 1. 함수

사용자 정의 함수

매크로 함수

```
#define     A        B     // A를 B로 치환
#define square(x) x*x
#define swap(type,a,b) {type temp=a; a=b; b=temp;}
```

```
#include<stdio.h>
#define square1(x) x*x
#define square2(x) (x)*(x)
int main() {
    printf("%d\n", square1(1 + 2));
    printf("%d\n", square2(1 + 2));
    printf("%f\n", square1(1.1));
    printf("%d\n", square2('d'));
    // d = 100 아스키 코드
    return 0;
}
```

square1 : 1 + 2 * 1 + 2 = 5

square2 : (1 + 2) * (1 + 2) = 9

define 사용에 주의사항 : 연산이 아닌 치환!!

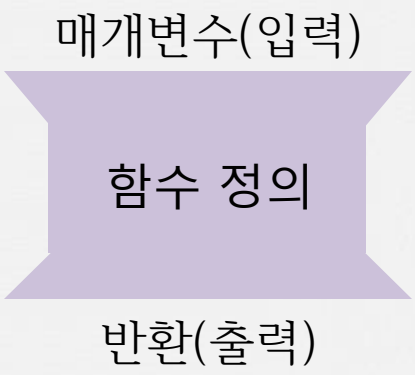
```
#include <stdio.h>
#define swap(type,a,b) {type temp=a; a=b;
b=temp;}
int main() {
  int m = 3, n = 4, temp;
  double x = 3.5, y = 8.4;
  printf("m=%d n=%d\n", m, n);
  printf("x=%.2f y=%.2f\n", x, y);
  swap(int, m, n);
  swap(double, x, y);
  printf("m=%d n=%d\n", m, n);
  printf("x=%.2f y=%.2f\n", x, y);
  return 0;
}
```

Day12 - 1. 함수

사용자 정의 함수

형식

```
반환(리턴)자료형 함수이름(자료형 인수1, 자료형 인수2, …) //함수의 선언
{
  함수 몸체 //함수의 정의
}
```



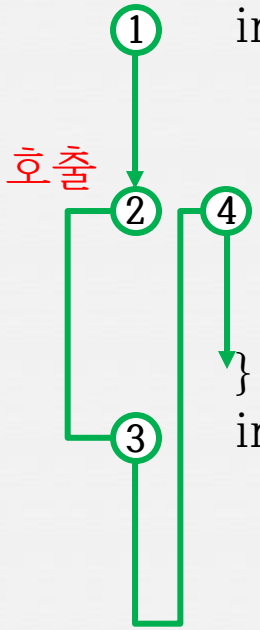
인수(매개 변수)
- 호출함수에서 함수로 전달되는 값의 자료 타입
- 인수가 여러 개일 경우에는 콤마(,)로 구분한다.

반환(리턴) 자료 타입
- 함수에서 계산된 결과값을 호출한 함수에 되돌려 주는 값의 자료형
- return을 사용하여 함수를 강제 종료 후 호출한 함수로 이동 가능
- 리턴이 없을 경우는 void형을 사용한다.

함수 이름
- 기능을 알 수 있는 함수명으로 사용!
- 중복으로 사용 할 수 없다

사용자 정의 함수


```c
#include <stdio.h>
int Plus(int x, int y){ //가인수 x, y
    int z;
    z=x+y;
    return (z);
}
int main(){
    int a,b,c;
    a=10, b=20;
    c=Plus(a,b); // 실인수 x, y
    printf("A + B = %d\n",c);
    return 0;
}
```




```c
#include <stdio.h>
int Plus(int x, int y); //함수 선언
int main(){
    int a,b,c;
    a=10, b=20;
    c=Plus(a,b); //함수 호출
    printf("A + B = %d\n",c);
    return 0;
}
int Plus(int x, int y){ //함수 정의
    int z;
    z=x+y;
    return (z);
}
```

함수의 호출
- 함수명과 인수를 작성하면 호출.
- 실행 순서 : 해당 함수로 이동 -> 완료 -> 호출한 함수로 이동
- 호출하는 함수의 인수를 실인수, 호출 당하는 함수의 인수를 가인수.
- 실인수와 가인수는 반드시 자료형과 변수의 개수가 일치해야 한다.

Day12 - 1. 함수

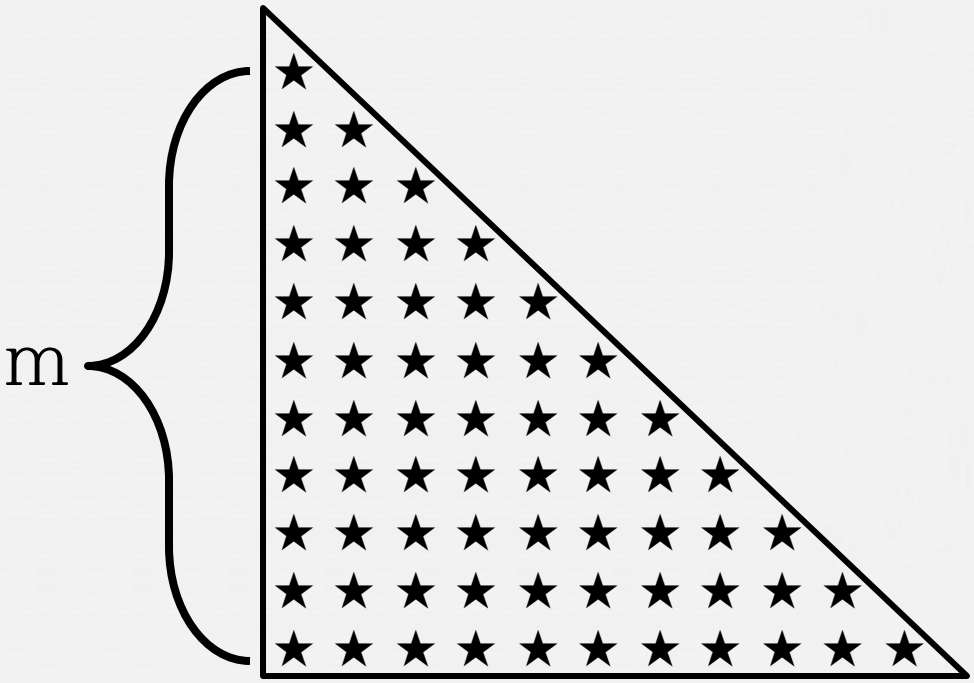
인수 전달 방법 및 리턴 값 확인하기
(1) 인수와 결과값의 반환이 모두 없는 경우

```
#include <stdio.h>
void display_menu(void) {
  printf("[단위 변환 프로그램]\n");
  printf("1. inch <-> cm\n");
  printf("2. cm <-> inch\n");
  printf("3. ℉ <-> ℃\n");
  printf("4. ℃ <-> ℉\n");
  printf("5. 끝내기\n");
  printf("메뉴를 선택하세요. :");
}
int main() {
  display_menu();
  return 0;
}
```

Day12 - 1. 함수

인수 전달 방법 및 리턴 값 확인하기
(2) 인수는 있고 결과값의 반환이 없는 경우

```
#include <stdio.h>
void fun(int m){
  for(int i=1;i<=m;i++){
    for(int j=1;j<=i;j++){
      printf("★");
    }
    printf("\n");
  }
}
int main(){
  int n=10;
  fun(n);
  return 0;
}
```



Day12 - 1. 함수

인수 전달 방법 및 리턴 값 확인하기
(3) 인수는 없고 결과값의 반환이 있는 경우

```
#include <stdio.h>
int display_menu(void) {
  int menu_func;
  printf("[단위 변환 프로그램]\n");
  printf("1. inch <-> cm\n");
  printf("2. cm <-> inch\n");
  printf("3. ℉ <-> ℃\n");
  printf("4. ℃ <-> ℉\n");
  printf("5. 끝내기\n");
  printf("메뉴를 선택하세요. :");
  scanf_s("%d", &menu_func);
  return menu_func;
}
int main() {
  int menu = display_menu();
  printf("%d", menu);
  return 0;
}
```

Day12 - 1. 함수

인수 전달 방법 및 리턴 값 확인하기
(4) 인수와 결과값의 반환이 모두 있는 경우

```
#include <stdio.h>
int Plus(int x, int y) {
    int z;
    z = x + y;
    return (z);
}
int main() {
    int a, b, c;
    a = 10, b = 20;
    c = Plus(a, b);
    printf("A + B = %d\n", c);
    return 0;
}
```

Day12 - 1. 함수

절대값 함수

int absolute(int value); [함수의 원형]

알파벳 소문자 -> 대문자 변환 함수

int upper(int value); [함수의 원형]

두 수를 비교 함수 (A, B)
A > B 경우) 1 반환
A = B 경우) 0 반환
A < B 경우) -1 반환

int compare(int A, int B); [함수의 원형]

```
반환형 함수_이름(매개변수_1, 매개변수_2){
  return 반환값;
}
```

Day12 - 1. 함수

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
int absolute(int value);//절대값 함수
int upper(int value);//알파벳 소문자-> 대문자 변환 함수
//두 수를 비교 함수(A, B)
//A > B 경우 )  1 반환
//A = B 경우 )  0 반환
//A < B 경우 ) - 1 반환
int compare(int A, int B);
int main() {
  srand((unsigned int)time(NULL));
  int alphabet = rand() % 26 + 97;
  printf("절대값 전 : %3d 절대값 후 : %3d\n", -97, absolute(-97));
  printf("소문자 : %3c 대문자 : %3c\n", alphabet, upper(alphabet));
  printf("compare(3,1) : %3d\n", compare(3, 1));
  printf("compare(2,2) : %3d\n", compare(2, 2));
  printf("compare(1,3) : %3d\n", compare(1, 3));
  return 0;
}
int absolute(int value){
  if (value < 0) value = -value;
  return value;
}
int upper(int value){
  if (96 < value && value < 123) value = value - 32;
  return value;
}
int compare(int A, int B){
  if (A > B) return 1;
  else if (A < B) return -1;
  else /* A==B */ return 0;
}
```

Day12 - 1. 함수

2개의 실수형 매개변수
실수형 반환값

```
float addition(float x, float y); // 더하기 함수 선언
float subtraction(float x, float y); // 빼기 함수 선언
float multiplication(float x, float y); // 곱하기 함수 선언
float divide(float x, float y); // 나누기 함수 선언
```

함수 작성하기

```
반환형 함수_이름(매개변수_1, 매개변수_2){
  return 반환값;
}
```

Day12 - 1. 함수

```
#include <stdio.h>
float addition(float x, float y); // 더하기 함수 선언
float subtraction(float x, float y); // 빼기 함수 선언
float multiplication(float x, float y); // 곱하기 함수 선언
float divide(float x, float y); // 나누기 함수 선언
int main() {
    float x = 20.2, y = 2.5;
    float result[4];
    result[0] = addition(x, y);
    result[1] = subtraction(x, y);
    result[2] = multiplication(x, y);
    result[3] = divide(x, y);
    printf("X + Y = %10.2f\n", result[0] );
    printf("X - Y = %10.2f\n", result[1] );
    printf("X * Y = %10.2f\n", result[2]);
    printf("X / Y = %10.2f\n", result[3]);
    return 0;
}
float addition(float x, float y){ return x+y; } // 더하기 함수 선언
float subtraction(float x, float y){ return x-y; } // 빼기 함수 선언
float multiplication(float x, float y){ return x*y; } // 곱하기 함수 선언
float divide(float x, float y){ return x/y; } // 나누기 함수 선언
```

Day12 - 2. 예제 코드

함수ver. 작성
[단위 변환 프로그램]
1. inch <-> cm
2. cm <-> inch
3. ℉ <-> ℃
4. ℃ <-> ℉
5. 끝내기
메뉴를 선택하세요. :

```
void display_menu(void);   // 메뉴판 출력
float input_value(void);   // 변환 할 값 입력 함수
float fah_to_cel(float fah); // 화씨 -> 섭씨 변환 함수
float cel_to_fah(float cel);  // 섭씨 -> 화씨 변환 함수
float inch_to_cm(float inch); // 인치 -> 센치미터 변환 함수
float cm_to_inch(float cm);   // 센치미터 -> 인치 변환 함수
```

```
반환형 함수_이름(매개변수_1, 매개변수_2){
  return 반환값;
}
```

Day12 - 2. 예제 코드

```
#include <stdio.h>
#include <stdlib.h>
///////////////함수의 원형(선언부)/////////////
void display_menu(void);
float input_value(void);
float fah_to_cel(float fah);
float cel_to_fah(float cel);
float inch_to_cm(float inch);
float cm_to_inch(float cm);
////////////////////////////////////////////
int main()
{
    int main_menu, end = 0;
    float cel, fah;
    float centi, inch;
    while (1) {
        display_menu();
        scanf_s("%d", &main_menu);
        while (getchar() != '\n');
        switch (main_menu) {
        case 1:
            inch = input_value();
            printf("%.3f inch = %.3f cm\n\n", inch, inch_to_cm(inch));
            break;
        case 2:
            centi = input_value();
            printf("%.3f cm = %.3f inch\n\n", centi, cm_to_inch(centi));
            break;
        case 3:
            fah = input_value();
            printf("%.3f ℉ = %.3f ℃\n\n", fah, fah_to_cel(fah));
            break;
        case 4:
            cel = input_value();
            printf("%.3f ℃ = %.3f ℉\n\n", cel, cel_to_fah(cel));
            break;
        case 5:
            printf("\n프로그램 종료 합니다.\n\n");
            exit(0); //프로그램 종료
        default:
            printf("잘못 입력했습니다.\n\n");
        }
    }
    return 0;
}
///////////////함수의 정의/////////////
void display_menu() {
    printf("[단위 변환 프로그램]\n");
    printf("1. inch <-> cm\n");
    printf("2. cm <-> inch\n");
    printf("3. ℉ <-> ℃\n");
    printf("4. ℃ <-> ℉\n");
    printf("5. 끝내기\n");
    printf("메뉴를 선택하세요. :");
}
float input_value(void) {
    float output;
    printf(" 값을 입력하세요. : ");
    scanf_s("%f", &output);
    while (getchar() != '\n');
    return output;
}
float fah_to_cel(float fah) {
    float cel;
    cel = (5.0 / 9.0) * (fah - 32.0);
    return cel;
}
float cel_to_fah(float cel) {
    float fah;
    fah = (9.0 / 5.0) * cel + 32.0;
    return fah;
}
float inch_to_cm(float inch) {
    float cm;
    cm = inch * 2.54;
    return cm;
}
float cm_to_inch(float cm) {
    float inch;
    inch = cm / 2.54;
    return inch;
}
////////////////////////////////////////
```

Day12 - 2. 예제 코드

로또 번호 생성기

- 로또 번호를 생성할 개수 저장 (최종 N 개 번호 생성)
- 1개 번호 생성
  * 1 ~ 45의 숫자 중 6개 랜덤 생성
  * 중복 숫자 검사 (이전에 나온 값 확인 or 숫자 체크 배열)
- 번호 정렬
  순차 or 선택 정렬
- 번호 출력

```
int TotalNumber(void); // 로또 번호 생성할 개수 입력 받는 함수
void create_value_check(int ary[]); //1set 번호 생성 함수 (이전 값 체크)
void Create_arr_check(int ary[]);// 1set 번호 생성 함수 (체크 배열)
void SortNumber(int ary[]); //6개 숫자 정렬 함수
void NumberPrint(int ary[]); // 6개 숫자 출력 함수
```

Day12 - 2. 예제 코드

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#define swap(type,a,b) {type t=a;a=b;b=t;}
#define ball 6
#define range 45
////함수의 원형(선언부)
int TotalNumber(void);
void create_value_check(int ary[]);
void Create_arr_check(int ary[]);
void SortNumber(int ary[]);
void NumberPrint(int ary[]);
int main() {
    srand((unsigned int)time(NULL));
    int selec[ball] = { 0 };
    int number;

    number = TotalNumber();

    for (int i = 0; i < number; i++) {
        create_value_check(selec);
        //Create_arr_check(selec);
        SortNumber(selec);
        printf("%04d : ", i + 1);
        NumberPrint(selec);
    }

    return 0;
}
////함수의 정의
int TotalNumber(void) {
    int n;
    printf(" [ 로또 번호 생성기 ] \n");
    printf("몇개를 만들고 싶나요? : ");
    scanf_s("%d", &n);
    while (getchar() != '\n');
    return n;
}
void create_value_check(int ary[])
{
    for (int i = 0; i < ball; ) {
        int check = 0;
        ary[i] = rand() % range + 1;
        for (int j = 0; j < i; j++) {
            if (ary[i] == ary[j]) {
                check = 1;
                break;
            }
        }
        if (check == 1) continue;
        i++;
    }
}
void Create_arr_check(int ary[]) {
    int count = 0, random, rotto[46]
= { 0 };
    while (count != ball) {
        random = rand() % range + 1;
        if (rotto[random] == 0) {
            rotto[random] = 1;
            ary[count++] = random;
        }
    }
}
void SortNumber(int ary[]) {
    for (int j = ball-1; j >= 0; j--) {
        int sel = 0;
        for (int k = 0; k <= j; k++) {
            if (ary[sel] < ary[k]) sel = k;
        }
        swap(int, ary[sel], ary[j]);
    }
}
void NumberPrint(int ary[]) {
    for (int j = 0; j < ball; j++) {
        printf("%4d ", ary[j]);
    }
    printf("\n");
}
```

Day 13 - Contents

1. **변수**
   - 전역변수, 지역변수
   - auto , static , extern, register 변수
2. **예제 코드**

Day13 - 1. 변수

변수

정의

- 자료형 : 어떤 종류의 자료를 다룰 것인가
- 유지시간 : 변수가 어디에서 사용 될 것인가
  언제까지 필요한가
  어떻게 초기화될 것인가

모든 변수는 제한된 사용영역과 생성~소멸 기간(life time)을 갖고 있다.

전역변수

- 함수 밖이나 외부파일에서 선언되어
  프로그램 전체에 걸쳐 사용될 수 있는 변수
- 전역변수는 0으로 자동 초기화

지역변수

- 특정 범위 내에서만 통용되는 변수로서
  선언된 블록이나 함수 내에서만 사용 ({ } 블록)
- 지역변수는 초기화를 하지 않으면 쓰레기 값(garbage)

Day13 - 1. 변수

변수

기억클래스
  자동변수(auto), 정적변수(static),
  외부변수(extern), 레지스터변수(register)

자동변수(auto) <생략 가능하며, 보통 생략>
 - 함수 안에서 선언되고 그 함수 안에서만 사용할 수 있는 유형(지역변수)
 - 함수가 실행되면 스택(stack)메모리에 할당되고,
   함수가 종료되면 메모리에서 해제
 - 자동변수는 항상 쓰레기(garbage)값을 갖기 때문에 초기화 필요.

static 변수
 - 함수 안에서 선언되면 지역변수와 같다.
 - 컴파일 시 데이터 영역에 메모리가 할당되어
   프로그램이 종료될 때까지 그 값을 유지한다는 특성
 - 초기 값 지정이 없으면 0으로 자동 초기화
   프로그램 실행 시 단 한번만 자동 초기화

Day13 - 1. 변수

변수

기억클래스
  자동변수(auto), 정적변수(static),
  외부변수(extern), 레지스터변수(register)

외부변수(extern)
- 함수의 외부에 선언되어 전역변수와 같다.
- 초기값은 0이고 프로그램이 종료될 때까지 유지된다.
- 다른 파일에서 외부변수로 선언된 변수의 값을 참조가능
  (전역변수, 정적변수 불가)
- 기억 영역의 할당은 정적변수와 같다.

레지스터변수(register)
- 선언된 함수내의 프로그램에만 영향을 미치는 지역변수 특징
- 함수가 실행중에만 사용, 종료되면 자동적으로 소멸 자동 변수와 같음
- 기억 장소가 레지스터이므로 실행 속도가 제일 빠름
- 레지스터 배정이 가능한 개수만큼 할당, 나머지는 자동 변수로 할당

Day13 - 1. 변수

변수

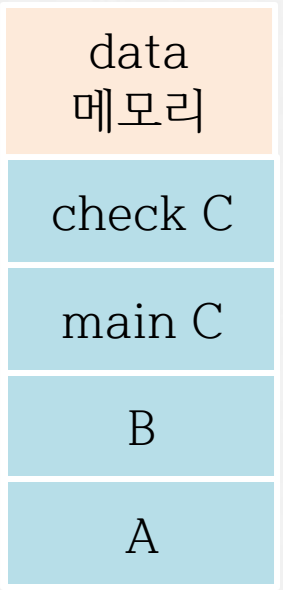
기억클래스

자동변수(auto), 정적변수(static),
외부변수(extern), 레지스터변수(register)

<table>
<tr><th>선언장소</th><th>기억클래스</th><th>지정자</th><th>기억장소</th><th>생존기간</th><th>유효범위</th></tr>
<tr><td rowspan="2">함수의 외부</td><td rowspan="2">외부변수</td><td>정의:없음</td><td rowspan="4">메모리
(정적영역)</td><td rowspan="3">영구적</td><td>프로그램전역</td></tr>
<tr><td>선언:extern</td><td rowspan="2">선언된 파일</td></tr>
<tr><td>외부정적변수</td><td>static</td></tr>
<tr><td rowspan="3">함수의 내부</td><td>내부정적변수</td><td>static</td><td rowspan="3">선언된 함수
또는 블럭</td></tr>
<tr><td>레지스터변수</td><td>register</td><td>레지스터</td><td rowspan="2">일시적</td></tr>
<tr><td>자동변수</td><td>auto</td><td>메모리(스택)</td></tr>
</table>

Day13 - 1. 변수

```
/*다른 파일에 외부 변수 선언
//int ex_a = 10;
//int ex_b = 20;
//int ex_c = 30;
//int ex_d = 40;
//int ex_e = 50;
//int ex_f = 60;
*/
extern int ex_a;
extern int ex_b;
extern int ex_c;
extern int ex_d;
extern int ex_e;
extern int ex_f;
#include <stdio.h>
int A;
static int B = 30;
void check() {
    static int C=10;
    int i = 20; A++; ++B; ++C; ex_d++; ex_e++; ex_f++;
    printf("check\n");
    printf("i = %d, A=%d, B=%d, C=%d\n", i, A, B, C);
    printf("ex_a = %d, ex_b = %d, ex_c = %d, ", ex_a, ex_b, ex_c);
    printf("ex_d = %d, ex_e = %d, ex_f = %d\n", ex_d, ex_e, ex_f);
}
int main() {
    int i = 10;
    static int C = 40;
    if (i > 0) {
        int value1 = 100;
        for (int k = 0; k < 10; k++) {
            printf("k = %d , value = %d\n", k, value1++);
        }
        int k = 20;
        printf("%d ", k);
    }
    int value1 = 20;
    printf("%d \n", value1);
    {
        int value11 = 11, value12 = 12, value13 = 13, value14 = 14, value15 =
15;
        printf("%d %d %d %d %d \n", value11, value12, value13, value14,
value15);
    }
    int value11 = 21, value12 = 22, value13 = 23, value14 = 24, value15 = 25;
    printf("%d %d %d %d %d \n\n", value11, value12, value13, value14,
value15);
    printf("main\n");
    A++, ++B, ++C, ex_a++, ex_b++, ex_c++;
    printf("i = %d, A=%d, B=%d C=%d \n", i, A, B, C);
    printf("ex_a = %d, ex_b = %d, ex_c = %d, ", ex_a, ex_b, ex_c);
    printf("ex_d = %d, ex_e = %d, ex_f = %d\n", ex_d, ex_e, ex_f);
    check();
    check();
    check();
    printf("main\n");
    A++, ++B, ++C, ex_a++, ex_b++, ex_c++;
    printf("i = %d, A=%d, B=%d C=%d \n", i, A, B, C);
    printf("ex_a = %d, ex_b = %d, ex_c = %d, ", ex_a, ex_b, ex_c);
    printf("ex_d = %d, ex_e = %d, ex_f = %d\n", ex_d, ex_e, ex_f);
    return 0;
}
```



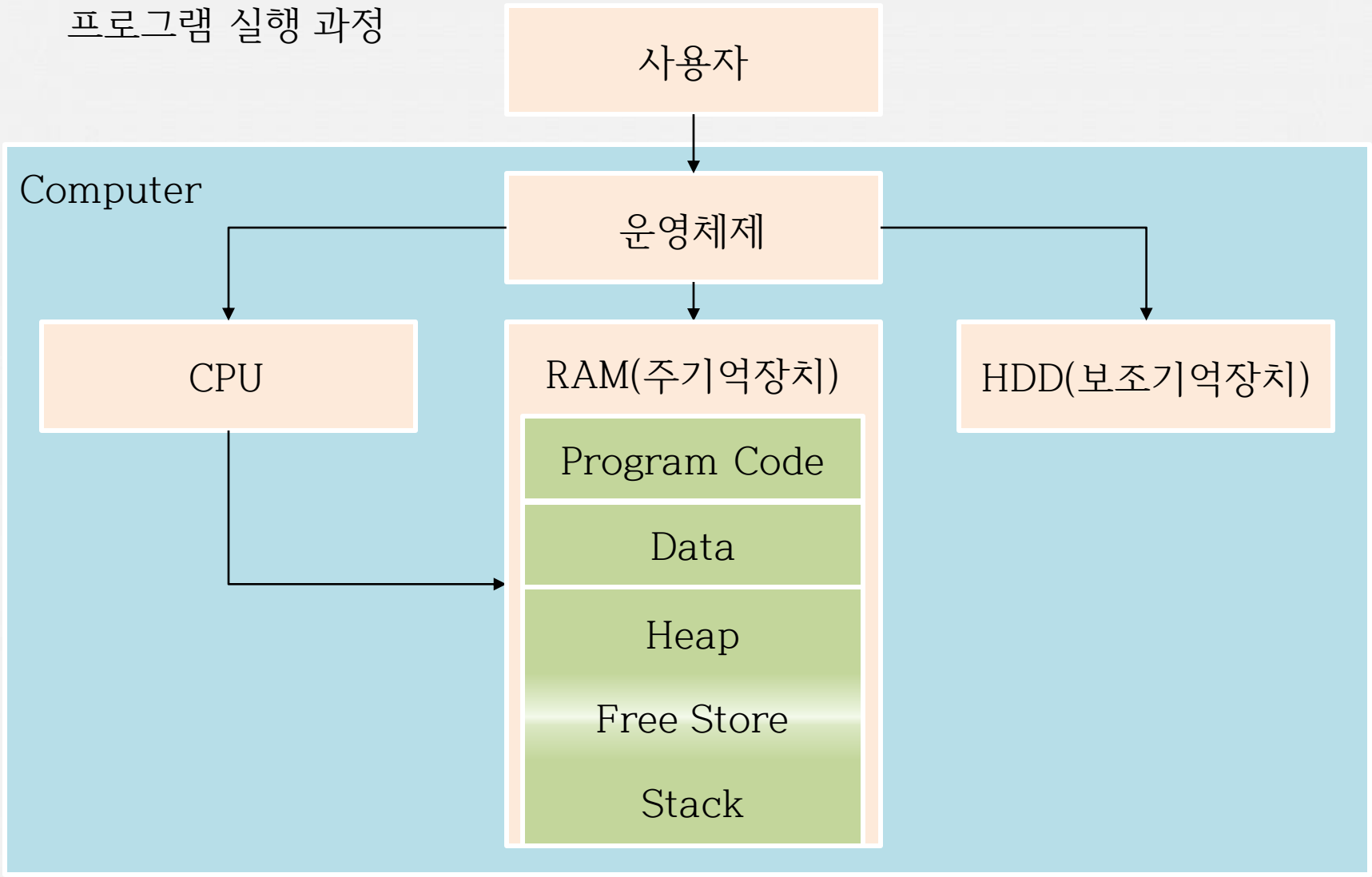
Day 14 - Contents

1. **포인터**
   - 메모리의 구조
   - 포인터 선언과 초기화
   - 포인터 연산
2. **예제 코드**
   - 포인터 연산 예제

Day14 - 1. 포인터

메모리의 구조

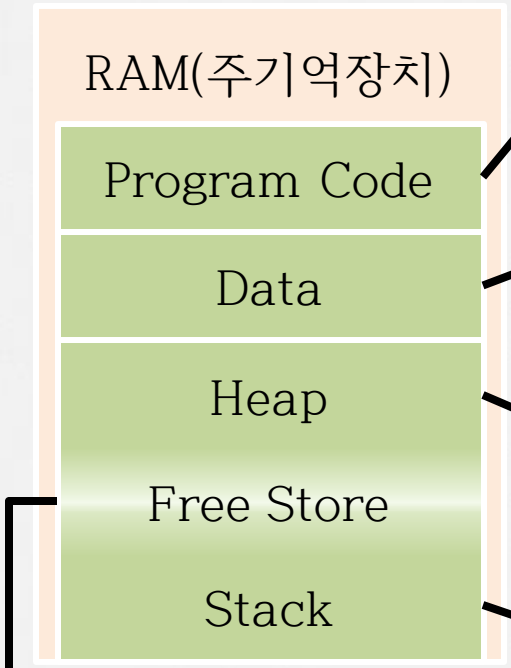
프로그램 실행 과정



Day14 - 1. 포인터

메모리의 구조

메모리 영역



Code 영역 : 순차적인 명령문(제어문, 함수, 상수, …)의 저장 공간
CPU는 코드 영역에 명령어를 하나씩 가져가서 처리

Data 영역 전역변수, 정적 변수,… 저장 공간
프로그램 실행 전에 선언, 프로그램이 끝날 때 까지 살아있는 변수들

Heap 영역 : 사용자의 동적 할당 (프로그램 실행 중 메모리 할당)
보통 낮은 주소 번지에서 높은 주소 번지로 할당

Stack 영역 : 지역 변수, 매개 변수, ({ } 안에 변수들, …)
함수가 종료되면 해당 함수에 할당된 변수들을 메모리에서 해제
보통 높은 주소 번지에서 낮은 주소 번지로 할당

Free Store : Heap or Stack 메모리 모두 할당 가능
Stack 영역의 메모리가 늘어나면 Heap 영역의 메모리가 줄어들고
Heap 영역의 메모리가 늘어나면 Stack 영역의 메모리가 줄어든다.

Day14 - 1. 포인터

메모리의 구조

메모리 영역

	메모리 주소	변수	값
Code	0x0000000000	…	…
	0x0000000001	…	…
	…	…	…
	0xFFFB0EE620	g	103
	0xFFFB0EE621	f	102
	0xFFFB0EE622	e	101
	0xFFFB0EE623	d	100
	0xFFFB0EE624	c	99
	0xFFFB0EE625	b	98
	0xFFFB0EE626	a	97
	…	…	…
	0xFFFFFFFFE	…	…
Stack	0xFFFFFFFFF	…	…

Day14 - 1. 포인터

포인터 선언과 초기화

포인터의 정의

메모리의 주소를 저장하는 변수

메모리 주소를 가리키는 변수로 포인터 변수라고 한다.

포인터 선언 - * (asterisk)연산자 사용

형식

자료형 * 포인터 변수 이름; //int * score;

자료형의 주소를 저장 할 수 있는 포인터 변수 선언

포인터 초기화 - & (ampersand) 연산자 사용

형식

포인터 변수 이름 = &변수; // int * score = &a;

&(변수)를 이용하면 변수의 주소를 알 수 있다.

메모리 주소	변수	값
…	…	…
0x7ffffb0ed245	pc	0x7ffffb0ee612
…	…	…

*pc →

메모리 주소	변수	값
…	…	…
0x7ffffb0ee612	c	A
…	…	…

Day14 - 1. 포인터

포인터 선언과 초기화

예시
int a=10;
int *pa=&a;

포인터 변수가 가리키는 변수의 자료 타입이 같아야 한다.
시작 주소 + 자료형 크기로 메모리 읽기
int * (시작 주소 + sizeof(int))

1 2 3 4

메모리 주소	변수	값
…	…	…
0x7ffffb0ee000	pa	0x7ffffb0ee628
…	…	…

*pa →

메모리 주소	변수	값
…	…	…
0x7ffffb0ee628	a	10
…	…	…

// 'A' = 65
char c='A';
char *pc=&c;

포인터 변수가 가리키는 변수의 자료 타입이 같아야 한다.
시작 주소 + 자료형 크기로 메모리 읽기
char * (시작 주소 + sizeof(char))

1

메모리 주소	변수	값
…	…	…
0x7ffffb0ed245	pc	0x7ffffb0ee612
…	…	…

*pc →

메모리 주소	변수	값
…	…	…
0x7ffffb0ee612	c	A
…	…	…

Day14 - 1. 포인터

포인터 선언과 초기화

* (asterisk)연산자 사용

형식

*(포인터 변수 이름)

포인터 변수 이름은 포인터가 가리키는 메모리 주소의 값을 참조한다.

포인터를 이용한 메모리 접근방식을 간접접근(Indirect Access)

연산자	용도	형식	예시
*	곱하기	변수 or 상수	a*b; x*3; 5*5;
	포인터 선언	자료형 * 변수명	char * cp; int * ip; double * dp;
	간접접근 (역참조)	*포인터 변수 이름	*cp; *ip; *dp;

Day14 - 1. 포인터

포인터 선언과 초기화

```
#include <stdio.h>
int main() {
  int a = 365;
  char c = 'A';
  int* p = &a;
  char* pc = &c;
  printf(" a 데이터 : %d \n", a);
  printf("*p 데이터 : %d \n\n", *p);

  printf("  c 데이터 : %d \n", c);
  printf("*pc 데이터 : %d \n\n", *pc);

  *p = *p + 10; *pc = *pc + 10;

  printf(" a 데이터 : %d \n", a);
  printf("*p 데이터 : %d \n\n", *p);

  printf("  c 데이터 : %d \n", c);
  printf("*pc 데이터 : %d \n\n", *pc);

  printf(" &a 주소 : %p \n", &a);
  printf("  p   값 : %p \n", p);
  printf(" &p 주소 : %p \n\n", &p);

  printf(" &c 주소 : %p \n", &c);
  printf(" pc   값 : %p \n", pc);
  printf("&pc 주소 : %p \n\n", &pc);
  return 0;
}
```

Day14 - 1. 포인터

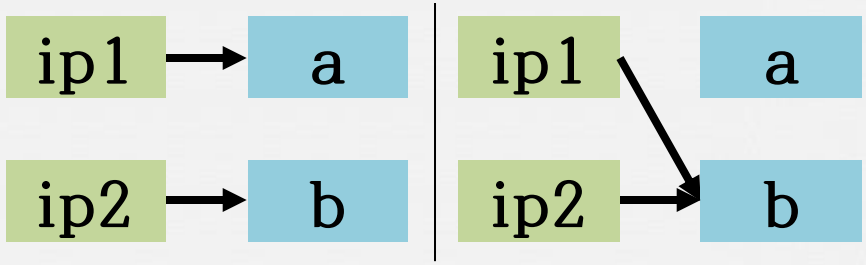
포인터 선언과 초기화

```
#include <stdio.h>
int main()
{
    int a, * ap = 0; //포인터 안전한 초기화
    char c, * cp = NULL; //포인터 안전한 초기화
    float e, * fp;
    /*
    //*fp = 200;
    //컴파일 에러!
    //fp의 쓰레기 값 주소가 어디를 가리키는지 알 수 없다.
    */
    a = 45; c = 'A'; e = 2.548; // 변수 초기화
    ap = &a; cp = &c; fp = &e; // 포인터 변수 초기화
    printf("%.15f\n", e);
    printf("a + c + e = %.2f\n", *ap + *cp + *fp);
    printf("a - c - e = %.2f\n", *ap - *cp - *fp);
    printf("a * c * e = %.2f\n", *ap * *cp * *fp);
    printf("a * c / e = %.2f\n", *ap * *cp / *fp);

    //ap = &c;    cp = &e;    fp = &a;
    // 포인터 변수와 자료 타입을 다르게 넣은 경우
    printf("*ap = %d | *cp = %d | *fp = %.2f\n", *ap, *cp, *fp);
    return 0;
}
```

Day14 - 1. 포인터

포인터 연산

- 같은 형을 갖는 포인터는 대입 가능
  int a=10, b=20;
  int * ip1=&a, *ip2=&b;
  ip1=ip2;



- 포인터 변수를 정수로 더하거나 빼기 가능
  ip1 + 1 x (자료형 크기); ip2 + 2 x (자료형 크기);
  ip1 – 3 x (자료형 크기); ip2 – 4 x (자료형 크기);

- 같은 배열을 가리키는 포인터의 빼기와 비교 가능 (덧셈 불가)
  int a[10]={1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
  int * ip1=&a[2], *ip2=&a[8];
  ip2 – ip1; (빼기 가능), ip2 > ip1 (비교 가능)

- 0(NULL) 과 비교 또는 0의 대입 가능
  ip1 = 0; ip2=NULL;

- 이외 모든 연산은 불가!

Day14 - 1. 포인터

```
#include <stdio.h>
int main() {//     p         q
    int a[10] = { 1, 3, 5, 7, 9, 11, 13, 15, 17, 19 }, b[10] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
    float c[10] = { 1.5, 5.5, 9.5, 13.5, 17.5, 21.5, 25.5, 29.5, 33.5, 37.5 };
    int* p, * q; //  fp
    float* fp;
    p = a; q = &a[3]; fp = c;
    printf("| a[0] = %d | *++p = %d | a[3] =%d | *++q = %d |\n\n", a[0], *++p, a[3], *++q);//전위 후위 차이 있음!
    printf("| c[0] = %.2f | *fp = %.2f | *fp + 2 = %.2f | *(fp + 2) = %.2f |\n\n", c[0], *fp, *fp + 2, *(fp + 2));
    printf("| *p + 2 = %d | *(p + 2) = %d | *q - 2 =%d | *(q - 2) = %d |\n\n", *p + 2, *(p + 2), *q - 2, *(q - 2));
    printf("| *(p - 3) = %d | *(q + 13) = %d |\n\n", *(p - 3), *(q + 13));
    printf("q : %p  \t  q - p = %3d  \n",   q, q - p);
    printf("p : %p  \t  p - q = %3d  \n\n", p, p - q);
    if (q > p + 2/* 증가값 < 3 : (참), >= 3 (거짓) */) printf("q : %p\np : %p\t참\n\n",  q, p + 2);
    else printf("q : %p\np : %p\t거짓\n\n",  q, p + 3);
    q = 0;     p = NULL;
    printf("q : %p \n", q);
    printf("p : %p \n", p);
    p = a;     fp = c;
    for (int i = 0; i < 25; i++) printf("   p+i 값 : %d \n", *(p + i));
    for (int i = 0; i < 15; i++) printf("   q+i 값 : %f \n", *(fp + i));
    //add=p+q; 연산 불가
    //p=fp; 연산 불가 or 역참조 불가
    return 0;
}
```

Day 15 - Contents

1. **포인터의 활용 1**
   - Call by value, Call by reference
   - 포인터와 배열
2. **예제 코드**

Day15 - 1. 포인터의 활용1

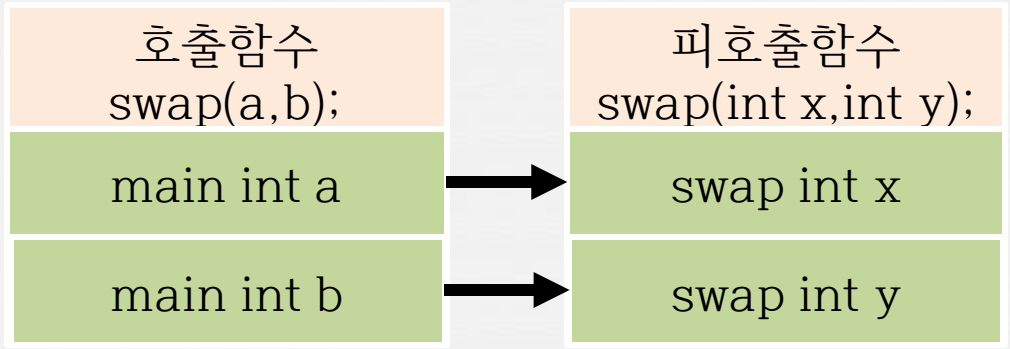
Call by value, Call by reference

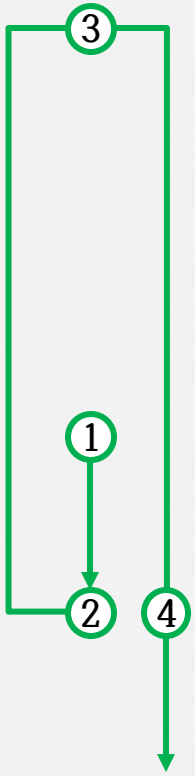
호출방식

- Call by value

C언어는 Call by value 기반으로 호출함수의 변수 값을 복사하는 전달 방식
호출 함수의 변수(실인수)는 영향을 받지 않는다.
연산 처리 결과값을 변경 할 경우 대입 연산 필요.

Call by value
복사






```
#include <stdio.h>
void swap(int x, int y) {
  int z;
  printf("\nswap call start\n");
  printf("x %d y %d\n", x, y);
  //swap 전
  z = x, x = y, y = z;
  printf("x %d y %d\n", x, y);
  //swap 후
  printf("swap call end\n\n");
}
int main() {
  int a = 10, b = 20;
  printf("A %d B %d\n", a, b);
  swap(a, b);
  printf("A %d B %d\n", a, b);
  return 0;
}
```

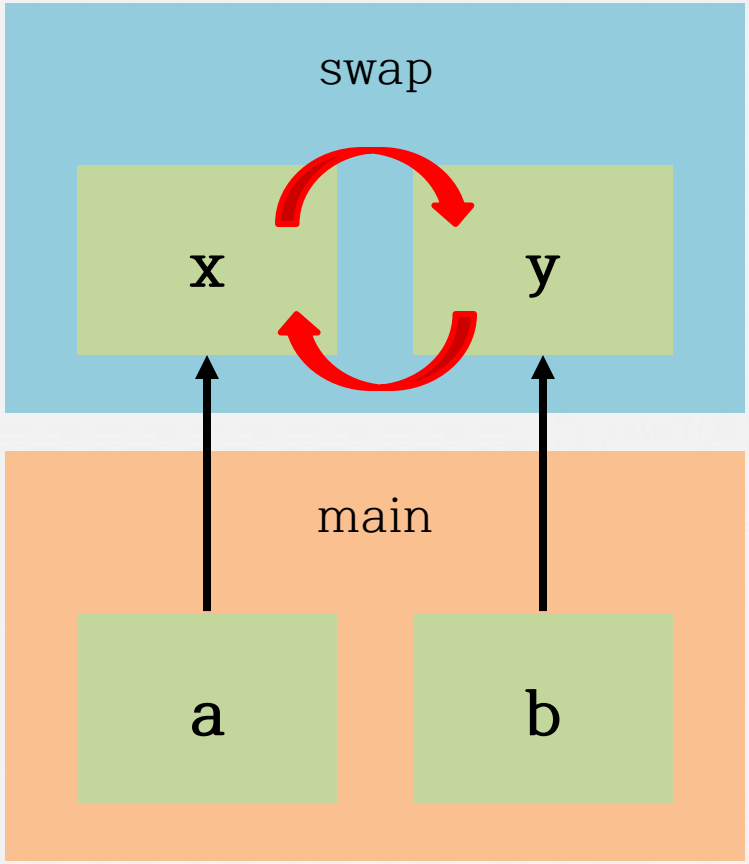
Day15 - 1. 포인터의 활용1

Call by value, Call by reference

호출방식

- Call by value

```
#include <stdio.h>
void swap(int x, int y) {
  int z;
  printf("\nswap call start\n");
  printf("x %d y %d\n", x, y); //swap 전
  z = x, x = y, y = z;
  printf("x %d y %d\n", x, y); //swap 후
  printf("swap call end\n\n");
}
int main() {
  int a = 10, b = 20;
  printf("A %d B %d\n", a, b);
  swap(a, b);
  printf("A %d B %d\n", a, b);
  return 0;
}
```



Day15 - 1. 포인터의 활용1

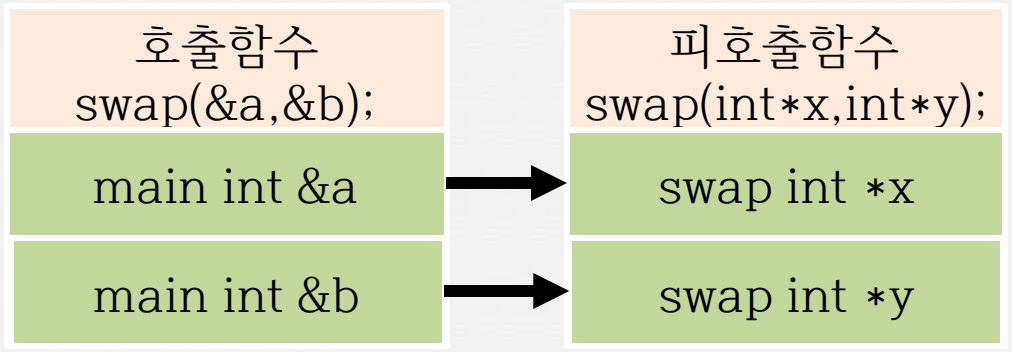
Call by value, Call by reference

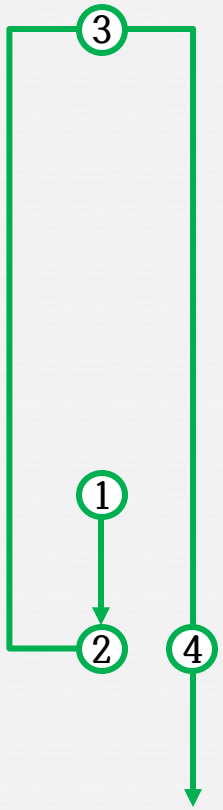
호출방식

- Call by reference

호출함수 변수에 주소를 넘겨주는 방식
호출함수 변수에 간접 접근할 수 있다.

Call by reference
주소 복사






```
#include <stdio.h>
void swap(int* x, int* y) {
  int z;
  printf("\nswap call start\n");
  printf("x %p y %p\n", x, y);
   //포인터 주소 값
  printf("x %d y %d\n", *x, *y);
   //포인터 가리키는 값 swap 전
  z = *x, * x = *y, * y = z;
  printf("x %d y %d\n", *x, *y);
   //포인터 가리키는 값 swap 후
  printf("swap call end\n\n");
}
int main() {
  int a = 10, b = 20;
  printf("A %d B %d\n", a, b);
  printf("a %p b %p\n", &a, &b);
  swap(&a, &b);
  printf("A %d B %d\n", a, b);
  printf("a %p b %p\n", &a, &b);
  return 0;
}
```

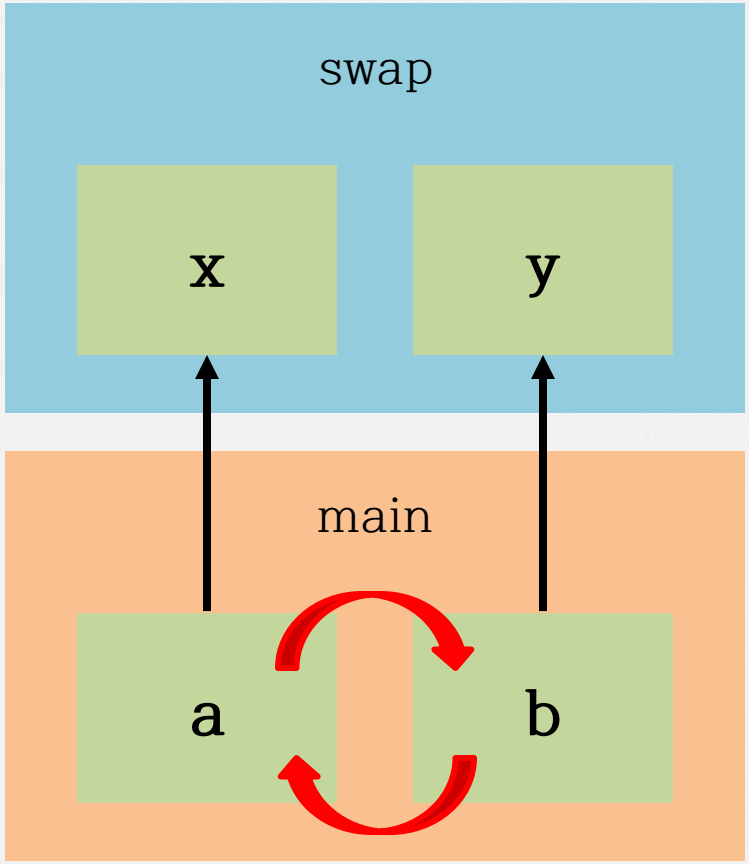
Day15 - 1. 포인터의 활용1

Call by value, Call by reference

호출방식

- Call by reference

```
#include <stdio.h>
void swap(int* x, int* y) { // int * x = &a, int * y = &b;
  int z;
  printf("\nswap call start\n");
  printf("x %p y %p\n", x, y);
   //포인터 주소 값
  printf("x %d y %d\n", *x, *y);
   //포인터 가리키는 값 swap 전
  z = *x, * x = *y, * y = z;
  printf("x %d y %d\n", *x, *y);
   //포인터 가리키는 값 swap 후
  printf("swap call end\n\n");
}
int main() {
  int a = 10, b = 20;
  printf("A %d B %d\n", a, b);
  printf("a %p b %p\n", &a, &b);
  swap(&a, &b);
  printf("A %d B %d\n", a, b);
  printf("a %p b %p\n", &a, &b);
  return 0;
}
```



Day15 - 1. 포인터의 활용1

포인터와 1차원 배열

포인터의 자료 타입은
가리키는 변수 (배열) or 포인터의 자료 타입과 같아야 한다.

예시) int a[5]= {1, 2, 3, 4, 5};

- 배열 이름(a)은 첫 번째 a[0]의 주소 값 (상수 포인터)
- 배열 이름(a)은 정수형 자료 타입 집합의 시작이다.
- 배열 각 요소의 자료 타입은 정수형(int)이다. { int , int , int , int , int }
- *(a+2) == a[2] 값이 같다

메모리 주소	배열 순서	값
(배열명이 주소) ary	int ary[0]	1
ary+1	int ary[1]	2
ary+2	int ary[2]	3
ary+3	int ary[3]	4
ary+4	int ary[4]	5
…	…	…

Day15 - 1. 포인터의 활용1

```
int a[5]={1,2,3,4,5};
int *p=a;
```

	메모리 주소	배열 순서	요소 값
P ->	0x7ffffb0ee620	a[0]	1
	0x7ffffb0ee624	a[1]	2
	0x7ffffb0ee628	a[2]	3
	0x7ffffb0ee62C	a[3]	4
	0x7ffffb0ee632	a[4]	5

주소 값 표현			
&a[0]	a	&p[0]	p
&a[1]	a+1	&p[1]	p+1
&a[2]	a+2	&p[2]	p+2
&a[3]	a+3	&p[3]	p+3
&a[4]	a+4	&p[4]	p+4

요소 값 표현				값
p[0]	*p	*a	a[0]	1
p[1]	*(p+1)	*(a+1)	a[1]	2
p[2]	*(p+2)	*(a+2)	a[2]	3
p[3]	*(p+3)	*(a+3)	a[3]	4
p[4]	*(p+4)	*(a+4)	a[4]	5

Day15 - 1. 포인터의 활용1

```
#include <stdio.h>
int main()
{
    int* p, a[5] = { 10, 20, 30, 40, 50 };
    //배열 이름은 배열의 시작주소 값이다.
    p = a;
    printf("1. %p\n", a);
    printf("2. %d\n", *p);
    printf("3. %d\n", *(p + 1));
    printf("4. %d\n", p[2]);
    printf("5. %d\n\n", *(a + 3));
    p += 1; // p = p + 1
    //a += 2; 배열 이름은 상수 포인터로 변경 불가능
    printf("6. %p\n", p);
    printf("7. %d\n", *(p - 1));
    printf("8. %d\n", *p);
    printf("9. %d\n", *(p + 1));
    printf("10. %d\n", a[3]);
    printf("11. %d\n", p[3]);
    printf("12. %d\n", p[-1]);
    printf("13. %d\n", *p + 2);
    return 0;
}
```

Day15 - 1. 포인터의 활용1

```
#include <stdio.h>
int main()
{
    int a[] = { 10, 20, 30, 40, 50, 60 };
    int i;
    int* ptr = a;
    for (i = 0; i < 5; i++)printf("%d ", *(ptr++));
    //*ptr과 ptr++이 결합된 형태
    printf("\n");
    ptr = a;
    for (i = 0; i < 5; i++)printf("%d ", *(++ptr));
    //++ptr과 *ptr이 결합된 형태
    printf("\n");
    ptr = a;
    for (i = 0; i < 5; i++)printf("%d ", ++ * (ptr++));
    //++a[i]과 같은 형태
    printf("\n");
    ptr = a;
    for (i = 0; i < 5; i++)printf("%d ", (*(ptr++))++);
    // a[i]++와 같은 형태
    printf("\n");
    for (i = 0; i < 6; i++)printf("%d ", a[i]);
    //배열 최종 값 확인
    printf("\n");
    return 0;
}
```

Day15 - 1. 포인터의 활용1

포인터와 2차원 배열

2차원 배열을 행렬 구조가 아닌 각 주소에 순차적으로 값을 저장

	ary[][0]	ary[][1]	ary[][2]
ary[0][]	ary[0][0]	ary[0][1]	ary[0][2]
ary[1][]	ary[1][0]	ary[1][1]	ary[1][2]
ary[2][]	ary[2][0]	ary[2][1]	ary[2][2]

주소 1칸 차이

메모리 주소	배열 순서	예제 값
(배열명이 주소) ary	int ary[0][0]	1
ary+1	int ary[0][1]	2
ary+2	int ary[0][2]	3
ary+3	int ary[1][0]	4
ary+4	int ary[1][1]	5
ary+5	int ary[1][2]	6
ary+6	int ary[2][0]	7
ary+7	int ary[2][1]	8
ary+8	int ary[2][2]	9
…	…	…

Day15 - 1. 포인터의 활용1

포인터와 2차원 배열

```
////포인터와 2차원 배열
#include <stdio.h>
#define max 3
int main() {
    int a[max + 1][max + 1] = { 0 }, k = 0;
    int* p_a = a[0]; // &a[0][0];
    printf("\n2차원 배열 값 입력\n");
    for (int i = 0; i < max; i++) {
        for (int j = 0; j < max; j++) {
            a[i][j] = ++k;
            printf("%5d", a[i][j]);
        }
        printf("\n");
    }
    printf("\n2차원 배열 값 출력 (배열 접근)\n");
    for (int i = 0; i < max; i++) {
        for (int j = 0; j < max; j++) {
            printf("%5d", a[i][j]);
        }
        printf("\n");
    }
    printf("\n2차원 배열 값 출력 (포인터 접근)\n");
    for (int i = 0; i < max+1; i++) {
        for (int j = 0; j < max+1; j++) {
            printf("%5d", *p_a++);
        }
        printf("\n");
    }
    printf("\n");
    return 0;
}
```

```
2차원 배열 값 입력
    1    2    3
    4    5    6
    7    8    9
2차원 배열 값 출력 (배열 접근)
    1    2    3
    4    5    6
    7    8    9
2차원 배열 값 출력 (포인터 접근)
    1    2    3    0
    4    5    6    0
    7    8    9    0
    0    0    0    0
```

Day16 - 1. 포인터의 활용2

포인터와 배열의 관계

포인터
- 메모리 주소를 저장하는 변수
- 간접 접근 , 역참조를 사용해 메모리 주소의 값을 읽고 쓸 수 있다.

배열
- 같은 자료형을 여러 개 묶은 그룹화
- 연속된 메모리 주소를 가진다.

	포인터와 배열의 관계
a[i] == *(a+i)	1차원 배열
a[i]	a배열의 i번 칸의 값을 접근
*(a+i)	a(주소 값) + i * (자료형 크기) 주소가 가지고 있는 값에 접근
a[i][j] == ???	2차원 배열
a[i][j]	a배열의 i번 줄 j번 칸의 값을 접근
???	a(주소 값) + ???(i, j의 식으로 표현) 주소가 가지고 있는 값에 접근

Day15 - 1. 포인터의 활용1

포인터와 2차원 배열

```
#include <stdio.h>
#define max 10
#define line 0
#define room 0
int main() {
  int a[max + 1][max + 1] = { 0 }, k = 0;
  int* p_a = a[0]; // &a[0][0];
  printf("\n2차원 배열 값 입력\n");
  for (int i = 0; i < max; i++) {
    for (int j = 0; j < max; j++) {
      a[i][j] = ++k;
    }
  }
  printf("\n2차원 배열 값 출력 (배열 접근)\n");
  for (int i = line; i < max; i++) {
    for (int j = room; j < max; j++) {
      printf("%5d", a[i][j]);
    }
    printf("\n");
  }
  printf("\n2차원 배열 값 출력 (포인터 접근)\n");
  for (int i = line; i < max; i++) {
    for (int j = room; j < max; j++) {
      printf("%5d", *(p_a + i * (max + 1) + j));
    }
    printf("\n");
  }
  printf("\n");
  return 0;
}
```

```
2차원 배열 값 출력 (배열 접근)
    1    2    3
    4    5    6
    7    8    9
2차원 배열 값 출력 (포인터 접근)
    1    2    3
    4    5    6
    7    8    9
```

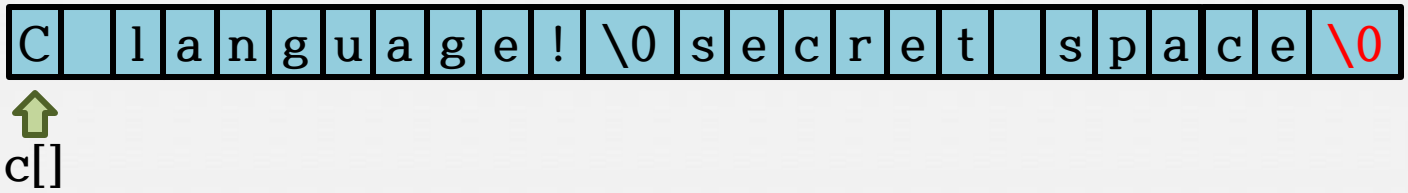
Day15 - 1. 포인터의 활용1

포인터 배열

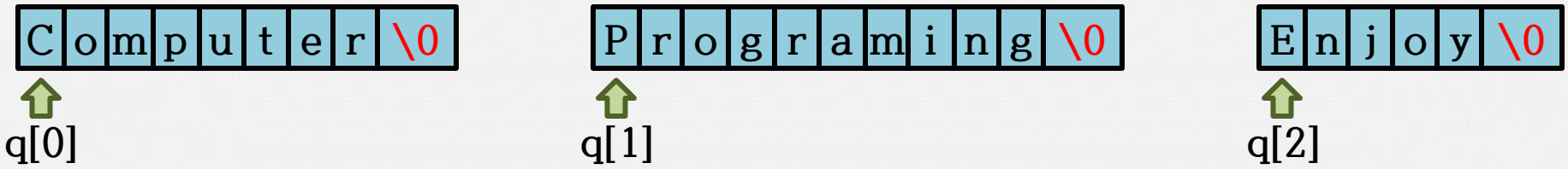
배열의 요소가 포인터로 이루어져 있다.

문자형 배열, char c[] = "C language!\0 secret space"; \0 (Null문자 생성)

- 문자형 메모리 26 byte 배열, c 배열 선언 & 초기화
- 각 문자 별 1byte 공간 할당, 대입



char* q[3] = { "Computer","Programing","Enjoy" }; { char*, char*, char* }



메모리 주소	배열 순서	예제 값
(배열명이 주소) q	char* q[0]	"Computer\0"
q+1	char* q[1]	"Programing\0"
q+2	char* q[2]	"Enjoy\0"
…	…	…

Day15 - 1. 포인터의 활용1

```
#include <stdio.h>
#include <string.h> // strlen() : 문자열 길이 확인 함수
int main()
{
  char c[] = "C language!\0 secret space";
  char* a = "array and pointer";
  char* q[3] = { "Computer", "Programing", "Enjoy" };
  char* p = c, d[20];
  int i = 0, k = 0;
  while (*(p + i) != '\0') {
    printf("%c", *(p + i));
    i++;
  }
  printf("\n\n");
  while (*p != '\0') {
    printf("%s\n", p);
    p++;
  }
  printf("\n\n");
  printf("%c %c \n", *(a + 4), a[4]);
  printf("pointer : %s \n", a);
  printf("  array : ");
  for (int i = 0; i < strlen(a); i++) printf("%c", a[i]);
  printf("\n\n");
  for (int i = 0; i < 3; i++) printf("%s\n", *(q + i));
  printf("\n\n");
  printf("[문자열 입력 20개]\n");
  scanf_s("%s", d, (int)sizeof(d));
  p = d;
  printf("입력된 문자열 %s\n\n", p);
  return 0;
}
```

Day 16 - Contents

1. **포인터의 활용 2**
   - 포인터와 배열, 함수
2. **예제 코드**

Day16 - 1. 포인터의 활용2

```c
//배열 방식 표현
#include <stdio.h>
#include <stdlib.h> // rand() , srand()
#include <time.h> // time()
#define MAX 10
#define rand_range 5000
void input(int p[], int cnt) { // int p[] == int *p;
  srand((unsigned)time(NULL));
  printf("sizeof : %llu, %llu, %llu, %llu\n\n", sizeof(p),
sizeof(int*), sizeof(p) / sizeof(p[0]), sizeof(p[0]));
  printf("%d개 난수 생성\n\n", cnt);
  for (int i = 0; i < cnt; i++) {
    printf("생성 %3d : ", i + 1);
    p[i] = rand() % rand_range + 1;
    printf(" %4d\n", p[i]);
  }
}
int num_max(int p[], int cnt) { // int p[] == int *p;
  int i, max = p[0];
  for (i = 1; i < cnt; i++) if (max < p[i]) max = p[i];
  return max;
}
int num_min(int p[], int cnt) { // int p[] == int *p;
  int i, min = p[0];
  for (i = 1; i < cnt; i++) if (min > p[i]) min = p[i];
  return min;
}
int main() {
  int num[MAX];
  input(num, MAX);
  printf("최대값 : %d\n", num_max(num, MAX));
  printf("최소값 : %d\n", num_min(num, MAX));
  return 0;
}
```

```c
//포인터 방식 표현
#include <stdio.h>
#include <stdlib.h> // rand() , srand()
#include <time.h> // time()
#define MAX 10
#define rand_range 5000
void input(int* p, int cnt) {
    srand(time(NULL));
    printf("%d개 난수 생성\n\n", cnt);
    for (int i = 0; i < cnt; i++) {
        printf("생성 %3d : ", i + 1);
        *p = rand() % rand_range + 1;
        printf(" %4d\n", *p++);
    }
}
int num_max(int* p, int cnt) {
    int i, max = *p;
    for (i = 1; i < cnt; i++) if (max < *++p) max = *p;
    return max;
}
int num_min(int* p, int cnt) {
    int i, min = *p;
    for (i = 1; i < cnt; i++) if (min > *++p)  min = *p;
    return min;
}
int main() {
    int num[MAX];
    input(num, MAX);
    printf("최대값 :[ %5d ]\n", num_max(num, MAX));
    printf("최소값 :[ %5d ]\n", num_min(num, MAX));
    return 0;
}
```

int *p와 int p[]는 동일한 표현이다.
함수의 매개변수로 배열의 인자가 전달되는 것을 직관적으로 이해 할 수 있는 int p[] 사용
배열을 함수의 인자로 전달하려면 모든 값을 복사 해야 한다.
그러나 C언어는 매개변수로 배열의 선언을 허용하지 않아 배열을 통째로 복사하는 방법은 없다. **대신 주소 값 전달 방식**

Day16 - 1. 포인터의 활용2

배열.ver 작성

```
void random(int ary[], int size);
int FindMax(int ary[], int size);
int FindMin(int ary[], int size);
void SequentialSort(int ary[], int size);
void SelectionSort(int ary[], int size);
void Print(int ary[], int size);
```

포인터 .ver 작성

```
void random(int * ary, int size);
int FindMax(int * ary, int size);
int FindMin(int * ary, int size);
void SequentialSort(int * ary, int size);
void SelectionSort(int * ary, int size);
void Print(int * ary, int size);
```

최대값 구하기

1차원 배열 10칸 메모리 공간 할당

각 방에 랜덤 변수 생성 (1 ~ 500)

기준 값([0]번) 1개를 설정, 각 방의 값을 비교하며 큰 값 저장 [최대값]
기준 값([0]번) 1개를 설정, 각 방의 값을 비교하며 작은 값 저장 [최소값]

10개의 배열 데이터를 정렬하기
순차 정렬, 선택 정렬
정렬 후 결과 출력

Day16 - 1. 포인터의 활용2

```
#include <stdio.h>
#include <stdlib.h>// srand(), rand()
#include <time.h> // time()
#define range 500
#define swap(type,a,b) {type temp=a;a=b;b=temp;}
#define count(ary) (sizeof(ary)/sizeof(ary[0]))
void random(int ary[], int size);
int FindMax(int ary[], int size);
int FindMin(int ary[], int size);
void SequentialSort(int ary[], int size);
void SelectionSort(int ary[], int size);
void Print(int ary[], int size);
int main() {
    srand((unsigned int)time(NULL));
    int value[10] = { 0 };

    random(value, count(value));
    printf("최대값 : %5d \n", FindMax(value, count(value)));
    printf("최소값 : %5d \n", FindMin(value, count(value)));
    SequentialSort(value, count(value));

    random(value, count(value));
    printf("최대값 : %5d \n", FindMax(value, count(value)));
    printf("최소값 : %5d \n", FindMin(value, count(value)));
    SelectionSort(value, count(value));

    return 0;}
void random(int ary[], int size) {
    for (int i = 0; i < size; i++) {
        ary[i] = rand() % range + 1;
        printf("value[%02d] = %5d \n", i, ary[i]);
    }
}
int FindMax(int ary[], int size) {
    int Max = ary[0];
    for (int i = 1; i < size; i++) {
        if (Max < ary[i]) Max = ary[i];
    }
    return Max;
}
int FindMin(int ary[], int size) {
    int Min = ary[0];
    for (int i = 1; i < size; i++) {
        if (Min > ary[i]) Min = ary[i];
    }
    return Min;
}
void SequentialSort(int ary[], int size) {//순차정렬
    for (int i = 0; i < size; i++){
        for (int j = i + 1; j < size; j++) {
            if (ary[i] > ary[j]) swap(int, ary[i], ary[j]);
        }
        Print(ary, size);
    }
}
void SelectionSort(int ary[], int size) {//선택정렬
    for (int i = size - 1; i >= 0; i--) {
        int MAX = 0;
        for (int j = 0; j <= i; j++) {
            if (ary[MAX] < ary[j]) MAX = j;
        }
        swap(int, ary[MAX], ary[i]);
        Print(ary, size);
    }
}
void Print(int ary[], int size)
{
    for (int j = 0; j < size; j++) {
        printf("%3d ", ary[j]);
    }
    printf("\n");
}
```

Day16 - 1. 포인터의 활용2

```
#include <stdio.h>
#include <stdlib.h>// srand(),rand()
#include <time.h> // time()
#define range 500
#define swap(type,a,b) {type temp=a;a=b;b=temp;}
#define count(ary) (sizeof(ary)/sizeof(ary[0]))
void random(int * ary, int size);
int FindMax(int * ary, int size);
int FindMin(int * ary, int size);
void SequentialSort(int * ary, int size);
void SelectionSort(int * ary, int size);
void Print(int * ary, int size);
int main() {
    srand((unsigned int)time(NULL));
    int value[10] = { 0 };

    random(value, count(value));
    printf("최대값 : %5d \n", FindMax(value, count(value)));
    printf("최소값 : %5d \n", FindMin(value, count(value)));
    SequentialSort(value, count(value));

    random(value, count(value));
    printf("최대값 : %5d \n", FindMax(value, count(value)));
    printf("최소값 : %5d \n", FindMin(value, count(value)));
    SelectionSort(value, count(value));

    return 0;
}
void random(int * ary, int size) {
    for (int i = 0; i < size; i++) {
        *ary = rand() % range + 1; // ary[i] == *(ary+i)
        printf("value[%02d] = %5d \n", i, *ary++);
    }
}
int FindMax(int * ary, int size)
{
    int Max = *(ary++);
    for (int i = 1; i < size; i++,ary++) {
        if (Max < *ary) Max = *ary;
    }
    return Max;
}
int FindMin(int * ary, int size)
{
    int Min = *(ary++);
    for (int i = 1; i < size; i++, ary++) {
        if (Min > *ary) Min = *ary;
    }
    return Min;
}
void SequentialSort(int * ary, int size) {//순차정렬
    int* pivot = ary;
    for (int i = 0; i < size; i++, pivot++) {
        int* compare = pivot +1;
        for (int j = i + 1; j < size; j++, compare++){
            if (*pivot > *compare) swap(int, *pivot, *compare);
        }
        Print(ary, size);
    }
}
void SelectionSort(int * ary, int size) {//선택정렬
    for (int i = size - 1; i >= 0; i--) {
        int MAX = 0;
        for (int j = 0; j <= i; j++) {
            if (*(ary+MAX) < *(ary+j)) MAX = j;
        }
        swap(int, *(ary + MAX), *(ary + i));
        Print(ary, size);
    }
}
void Print(int * ary, int size) {
    for (int j = 0; j < size; j++, ary++) {
        printf("%3d ", *ary);
    }
    printf("\n");
}
```

Day16 - 1. 포인터의 활용2

배열.ver 작성

```
void random(int ary[], int size, int* odd, int* even);
void SequentialSort(int ary[], int start, int end);
void SelectionSort(int ary[], int start, int end);
void Print(int ary[], int start, int end);
```

포인터 .ver 작성

```
void random(int * ary, int size, int* odd, int* even);
void SequentialSort(int * ary, int start, int end);
void SelectionSort(int * ary, int start, int end);
void Print(int * ary, int start, int end);
```

홀수, 짝수 개수 세기

1차원 배열 100칸 메모리 공간 할당
배열 칸 당 랜덤 변수 생성 (1 ~ 1000)
각 칸의 값이 홀수 & 짝수 판별 -> 홀수, 짝수 수량 파악

홀수 값을 오름차순 정렬 (순차정렬)
짝수 값을 오름차순 정렬 (선택정렬)

결과 출력

Day16 - 1. 포인터의 활용2

```
#include <stdio.h>
#include <stdlib.h>// srand(), rand()
#include <time.h> // time()
#define COUNT 100
#define RANGE 1000
#define count(ary) (sizeof(ary)/sizeof(ary[0]))
#define swap(type,a,b) {type temp=a;a=b;b=temp;}
void random(int ary[], int size, int* odd, int* even);
void SequentialSort(int ary[], int start, int end);
void SelectionSort(int ary[], int start, int end);
void Print(int ary[], int start, int end);
int main() {
    srand((unsigned int)time(NULL));
    int value[COUNT] = { 0 }, even, odd;

    random(value, count(value), &odd, &even);
    Print(value, 0, count(value));
    SequentialSort(value, 0, odd);
    Print(value, 0, count(value));
    SelectionSort(value, even, count(value));
    Print(value, 0, count(value));

    printf("\t[홀수 목록]\n");
    Print(value, 0, odd);
    printf("\t[짝수 목록]\n");
    Print(value, even, count(value));

    return 0;
}
void random(int ary[], int size, int* odd, int* even)
{
    *odd = 0, * even = size;
    for (int i = 0; i < size; i++) {
        int number = rand() % RANGE + 1;
        if (number % 2 == 0) ary[--(*even)] = number;
        else ary[(*odd)++] = number;
    }
}
void SequentialSort(int ary[], int start, int end)
{//순차정렬
    for (int i = start; i < end; i++) {
        for (int j = i + 1; j < end; j++) {
            if (ary[i] > ary[j]) swap(int, ary[i], ary[j]);
        }
    }
}
void SelectionSort(int ary[], int start, int end)
{//선택정렬
    for (int i = end - 1; i >= start; i--) {
        int MAX = start;
        for (int j = start; j <= i; j++) {
            if (ary[MAX] < ary[j]) MAX = j;
        }
        swap(int, ary[MAX], ary[i]);
    }
}
void Print(int ary[], int start, int end) {
    for (int i = start; i < end; i++) {
        printf("value[%05d] = %5d \n", i, ary[i]);
    }
    printf("\n\n");
}
```

Day16 - 1. 포인터의 활용2

```
#include <stdio.h>
#include <stdlib.h>// srand(), rand()
#include <time.h> // time()
#define COUNT 100
#define RANGE 1000
#define count(ary) (sizeof(ary)/sizeof(ary[0]))
#define swap(type,a,b) {type temp=a;a=b;b=temp;}
void random(int * ary, int size, int* odd, int* even);
void SequentialSort(int * ary, int start, int end);
void SelectionSort(int * ary, int start, int end);
void Print(int * ary, int start, int end);
int main() {
    srand((unsigned int)time(NULL));
    int value[COUNT] = { 0 }, even, odd;
    random(value, count(value), &odd, &even);
    Print(value, 0, count(value));

    SequentialSort(value, 0, odd);
    SelectionSort(value, even, count(value));

    printf("\t[홀수 목록]\n");
    Print(value, 0, odd);
    printf("\t[짝수 목록]\n");
    Print(value, even, count(value));
    printf("\t[전체 목록]\n");
    Print(value, 0, count(value));

    return 0;
}
void random(int * ary, int size, int* odd, int* even) {
    *odd = 0, * even = size;}
    for (int i = 0; i < size; i++) {
        int number = rand() % RANGE + 1;
        if (number % 2 == 0) *(ary + --(*even)) = number;
        else *(ary + (*odd)++) = number;
    }
}
void SequentialSort(int * ary, int start, int end) {// 순차정렬
    int* pivot = ary+ start;
    for (int i = start; i < end; i++, pivot++) {
        int* compare = pivot + 1;
        for (int j = i + 1; j < end; j++, compare++) {
            if (*pivot > *compare) swap(int, *pivot, *compare);
        }
    }
}
void SelectionSort(int * ary, int start, int end) {// 선택정렬
    for (int i = end - 1; i >= start; i--) {
        int MAX = start;
        for (int j = start; j <= i; j++) {
            if (*(ary + MAX) < *(ary + j)) MAX = j;
        }
        swap(int, *(ary + MAX), *(ary + i));
    }
}
void Print(int * ary, int start, int end) {
    for (int i = start; i < end; i++) {
        printf("value[%05d] = %5d \n", i, *(ary + i));
    }
    printf("\n\n");
}
```

Day16 - 1. 포인터의 활용2

배열.ver 작성

```
int TotalNumber(void);
void create_value_check(int ary[]);
void Create_arr_check(int ary[]);
void SortNumber(int ary[]);
void NumberPrint(int ary[]);
```

포인터 .ver 작성

```
int TotalNumber(void);
void Create_value_check(int * ary);
void Create_arr_check(int * ary);
void SortNumber(int * ary);
void NumberPrint(int * ary);
```

로또 번호 생성기

- 로또 번호를 생성할 개수 저장 (최종 N 개 번호 생성)
- 1개 번호 생성
  * 1 ~ 45의 숫자 중 6개 랜덤 생성
  * 중복 숫자 검사 (이전에 나온 값 확인 or 숫자 체크 배열)
- 번호 정렬
  순차 or 선택 정렬
- 번호 출력

Day16 - 1. 포인터의 활용2

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#define swap(type,a,b) {type t=a;a=b;b=t;}
#define ball 6
#define range 45
////함수의 원형(선언부)
int TotalNumber(void);
void create_value_check(int ary[]);
void Create_arr_check(int ary[]);
void SortNumber(int ary[]);
void NumberPrint(int ary[]);
int main() {
    srand((unsigned int)time(NULL));
    int selec[ball] = { 0 };
    int number;

    number = TotalNumber();

    for (int i = 0; i < number; i++) {
        create_value_check(selec);
        //Create_arr_check(selec);
        SortNumber(selec);
        printf("%04d : ", i + 1);
        NumberPrint(selec);
    }

    return 0;
}
////함수의 정의
int TotalNumber(void) {
    int n;
    printf(" [ 로또 번호 생성기 ] \n");
    printf("몇개를 만들고 싶나요? : ");
    scanf_s("%d", &n);
    while (getchar() != '\n');
    return n;
}
void create_value_check(int ary[])
{
    for (int i = 0; i < ball; ) {
        int check = 0;
        ary[i] = rand() % range + 1;
        for (int j = 0; j < i; j++) {
            if (ary[i] == ary[j]) {
                check = 1;
                break;
            }
        }
        if (check == 1) continue;
        i++;
    }
}
void Create_arr_check(int ary[]) {
    int count = 0, random, rotto[46]
= { 0 };
    while (count != ball) {
        random = rand() % range + 1;
        if (rotto[random] == 0) {
            rotto[random] = 1;
            ary[count++] = random;
        }
    }
}
void SortNumber(int ary[]) {
    for (int j = ball-1; j >= 0; j--) {
        int sel = 0;
        for (int k = 0; k <= j; k++) {
            if (ary[sel] < ary[k]) sel = k;
        }
        swap(int, ary[sel], ary[j]);
    }
}
void NumberPrint(int ary[]) {
    for (int j = 0; j < ball; j++) {
        printf("%4d ", ary[j]);
    }
    printf("\n");
}
```

Day16 - 1. 포인터의 활용2

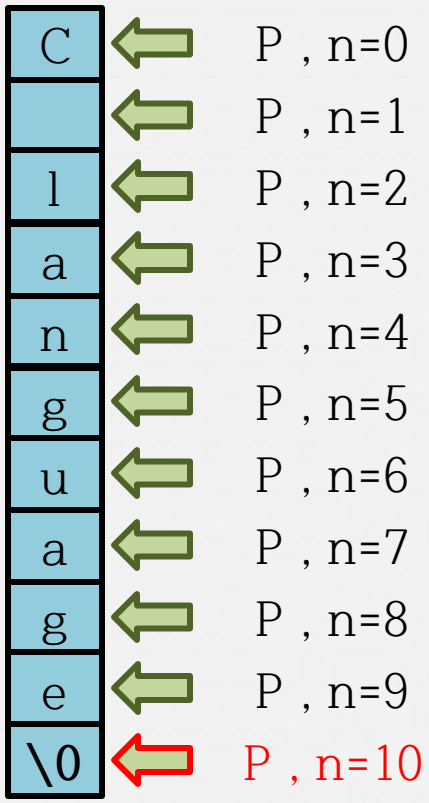
```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#define swap(type,a,b) {type t=a;a=b;b=t;}
#define ball 6
#define range 45
////함수의 원형(선언부)
int TotalNumber(void);
void Create_value_check(int * ary);
void Create_arr_check(int * ary);
void SortNumber(int * ary);
void NumberPrint(int * ary);
int main() {
    srand((unsigned int)time(NULL));
    int selec[ball] = { 0 };
    int number;

    number = TotalNumber();

    for (int i = 0; i < number; i++) {
        //Create_value_check(selec);
        Create_arr_check(selec);
        SortNumber(selec);
        printf("%04d : ", i + 1);
        NumberPrint(selec);
    }

    return 0;
}
////함수의 정의
int TotalNumber(void) {
    int n;
    printf(" [ 로또 번호 생성기 ] \n");
    printf("몇개를 만들고 싶나요? : ");
    scanf_s("%d", &n);
    while (getchar() != '\n');
    return n;
}
void Create_value_check(int * ary)
{
    int* pivot = ary;
    for (int i = 0; i < ball; ) {
        int check = 0;
        *pivot = rand() % range + 1;
        for (int j = 0; j < i; j++) {
            if (*pivot == *(ary+j)) {
                check = 1;
                break;
            }
        }
        if (check == 1) continue;
        i++;
        pivot++;
    }
}
void Create_arr_check(int * ary) {
    int count = 0, random, rotto[46] = { 0 };
    while (count != ball) {
        random = rand() % range + 1;
        if (rotto[random] == 0) {
            rotto[random] = 1;
            *(ary++) = random;
            count++;
        }
    }
}
void SortNumber(int * ary) {
    for (int j = ball - 1; j >= 0; j--) {
        int sel = 0;
        for (int k = 0; k <= j; k++) {
            if (*(ary+sel) < *(ary+k)) sel = k;
        }
        swap(int, *(ary + sel), *(ary + j));
    }
}
void NumberPrint(int * ary) {
    for (int j = 0; j < ball; j++,ary++) {
        printf("%4d ", *ary);
    }
    printf("\n");
}
```

Day16 - 1. 포인터의 활용2

포인터와 함수

C언어는 Call by value 기반이므로
Call by reference 를 구현하기 위해 함수의 매개변수로 포인터를 사용.
[매개변수로 배열의 선언을 허용하지 않고 포인터(주소)로 전달]
메모리 주소를 사용해 데이터 읽기, 쓰기 가능 (간접접근, 역참조)

```
#include <stdio.h>
int str_len(char* p) {
    int n = 0;
    while (*p++)n++;
    return n;
}
int main() {
    char c[15] = "C language", d[50];
    int len;
    len = str_len(c);
    printf("문자열 길이 : %d\n", len);
    printf("문자열 입력 : ");
    scanf_s("%s", d, (int)sizeof(d));
    len = str_len(d);
    printf("문자열 길이 : %d\n", len);
    return 0;
}
```

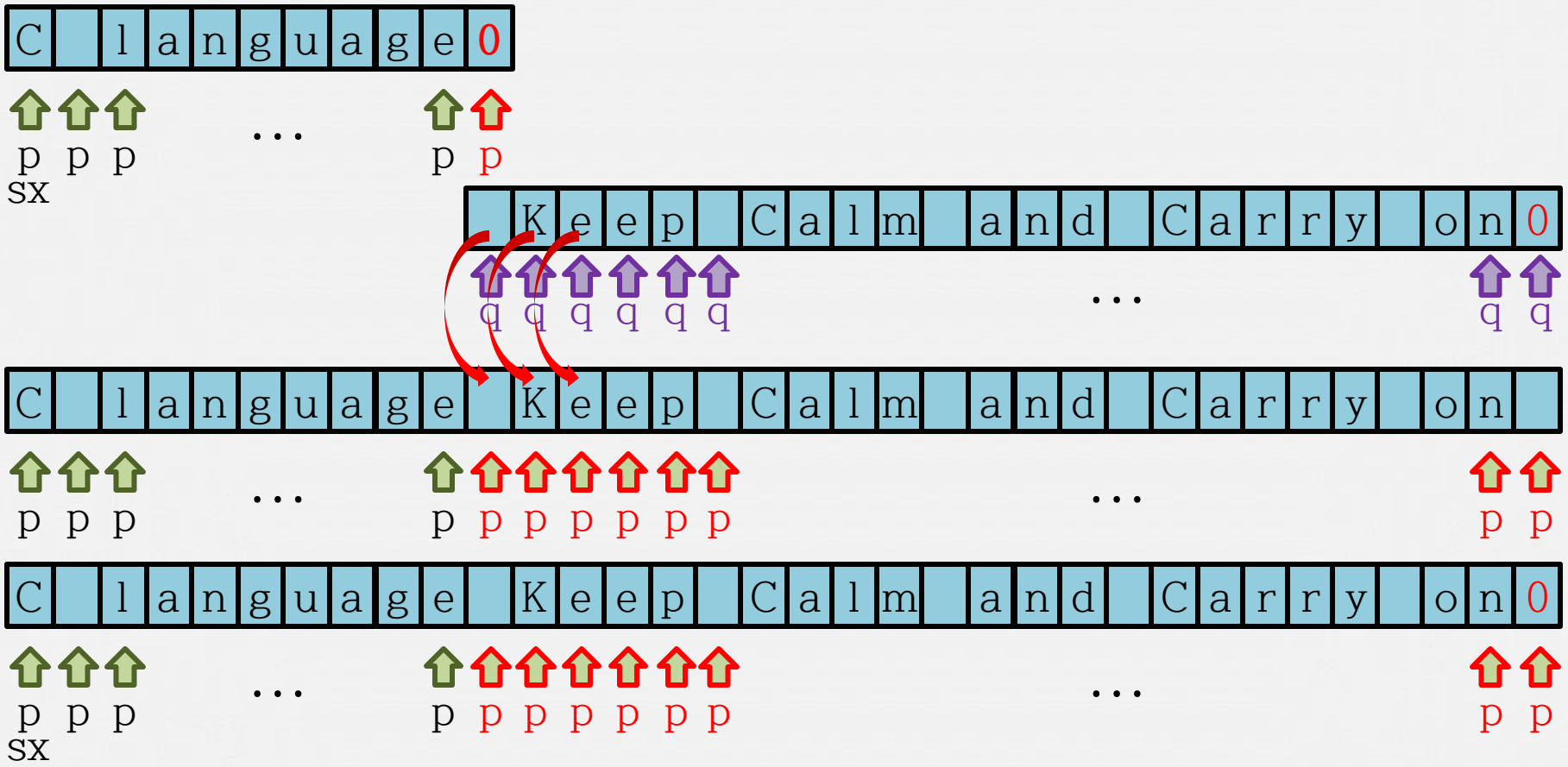


Day16 - 1. 포인터의 활용2

```
#include <stdio.h>
char* str_cat(char* p, char* q) {
    char* sx = p;
    while (*p)p++; // p가 가리키는 곳을 문자열의 \0 문자 위치로 (문자열 끝으로) 변경
    while (*q)*p++ = *q++; // 이어 붙이기
    *p = '\0'; // 문자열 마지막 주소에 \0 값 넣기
    return sx; // 문자열 첫 주소 반환
}
int main() {
    char a[100] = "C language", * b = (char*)" Keep Calm and Carry On"; //(const char)
    char c[100], d[50];
    //str_cat(a, b);
    printf("*a + *b = %s\n\n", str_cat(a, b));
    printf("1 문자열 입력 : ");
    scanf_s("%s", c, (int)sizeof(c));
    while (getchar() != '\n');
    printf("2 문자열 입력 : ");
    scanf_s("%s", d, (int)sizeof(d));
    while (getchar() != '\n');
    printf("*c + *d = %s\n\n", str_cat(c, d));
    return 0;
}
```

Day16 - 1. 포인터의 활용2

```
char* str_cat(char* p, char* q) {
    char* sx = p;
    while (*p)p++; // p가 가리키는 곳을 문자열의 \0 문자 위치로 (문자열 끝으로) 변경
    while (*q)*p++ = *q++; // 이어 붙이기
    *p = '\0'; // 문자열 마지막 주소에 \0 값 넣기
    return sx; // 문자열 첫 주소 반환
}
```



Day16 - 1. 포인터의 활용2

<string.h> strupr(), strlwr(), strrev() 구현

char* str_upr(char* str)
문자열에 포함된 소문자를 모두 대문자로 변환하는 함수

AbCdEfg123!@# \0 -> ABCDEFG123!@# \0

char* str_lwr(char* str)
문자열에 포함된 대문자를 모두 소문자로 변환하는 함수

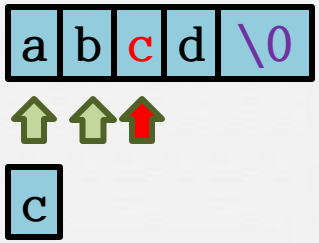
AbCdEfg123!@# \0 -> abcdefg123!@# \0

char* str_rev(char* str)
문자열의 순서를 바꾸는 함수

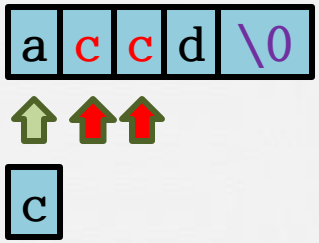
hello \0 -> olleh \0

Day16 - 1. 포인터의 활용2

<string.h> strcmp(), strchr() 구현
char* str_chr(char* str1, int ch); [함수의 원형]
문자열 내에 일치하는 문자가 있는지 검사하는 함수
성공 : 문자의 주소를 반환
실패 : NULL (없음)

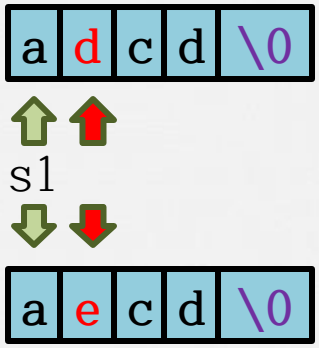


(사용자 정의 함수)
int str_cnt(char* str1, int ch) ; [함수의 원형]
문자열의 일치하는 문자의 수량 검사하는 함수
성공 : 포함된 문자의 수량 반환
실패 : 0



2개

int str_cmp(char* str1, char* str2); [함수의 원형]
두 개의 문자열을 아스키 코드 값으로 비교
- str1 < str2 인 경우 -1 반환
- str1 > str2 인 경우 1 반환
- str1 == str2 인 경우 0 반환



Day16 - 1. 포인터의 활용2

```
//
//char* str_upr(char* str)
//문자열에 포함된 소문자를 모두 대문자로 변환하는 함수
//AbCdEfg123!@# \0 -> ABCDEFG123!@# \0
//
//char* str_lwr(char* str)
//문자열에 포함된 대문자를 모두 소문자로 변환하는 함수
//AbCdEfg123!@# \0 -> abcdefg123!@# \0
//
//char* str_rev(char* str)
//문자열의 순서를 바꾸는 함수
//hello \0->olleh \0
//
#include <stdio.h>
char* strupr(char* str) {
    char* p = str;
    while (*p) {
        if (96 < *p && *p < 122) {
            *p = *p - 32;
        }
        p++;
    }
    return str;
}
char* strlwr(char* str) {
    char* p = str;
    while (*p) {
        if (64 < *p && *p < 90) {
            *p = *p + 32;
        }
        p++;
    }
    return str;
}
char* strrev(char* str) {
    int len = 0, st, end;
    char* p = str;
    while (*p++)len++;
    st = 0;
    end = len - 1;
    while (st < end) {
        char temp = *(str + st);
        *(str + st) = *(str + end);
        *(str + end) = temp;
        st++;
        end--;
    }
    return str;
}
int main() {
    char arr[] = "AbCdEfg123!@# Test";
    printf("  원본 : %s \n", arr);
    printf("  역순 : %s \n", strrev(arr));
    printf("대문자 : %s \n", strupr(arr));
    printf("소문자 : %s \n", strlwr(arr));
    return 0;
}
```

Day16 - 1. 포인터의 활용2

```
/*
<string.h> strcmp(), strchr() 구현
int str_cmp(char* str1, char* str2); [함수의 원형]
두 개의 문자열을 아스키 코드 값으로 비교
str1 < str2 인 경우 -1 반환
str1 > str2 인 경우 1 반환
str1 == str2 인 경우 0 반환
*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define swap(type,val1,val2) {type temp=val1; val1=val2; val2=temp;}
int str_cmp(char* str1, char* str2) {
  int result = 0;
  if (str1 == NULL) return -1;
  else if (str2 == NULL) return 1;
  else {
    while (1) {
      if (*str1 == '\0') break; // 0 == '\0'
      if (*str2 == '\0') break; // 0 == '\0'
      if (*str1 != *str2) break;
      str1++;
      str2++;
    }
    result = *str1 - *str2;
  }
  if (result > 0) return 1;
  else if (result < 0) return -1;
  else return 0;
}
```

```
/*
char* str_chr(char* str1, int ch); [함수의 원형]
문자열 내에 일치하는 문자가 있는지 검사하는 함수
성공 : 문자의 주소를 반환
실패 : NULL (없음)
*/
char* str_chr(char* str1, int ch) {
  if (str1 == NULL) return NULL;
  while (*str1 != '\0') {
    if (*str1 == ch) break;
    str1++;
  }
  return str1;
}



/*
(사용자 정의 함수)
int str_cnt(char* str1, int ch) ; [함수의 원형]
문자열의 일치하는 문자의 수량 검사하는 함수
성공 : 포함된 문자의 수량 반환
실패 : 0
에러 : -1
*/
int str_cnt(char* str1, int ch) {
  int cnt = 0;
  if (str1 == NULL) return -1;
  while (*str1 != '\0') {
    if (*str1 == ch) cnt++;
    str1++;
  }
  return cnt;
}
```

```
int main()
{
  char arr1[] = "strawberry", arr2[] = "strong man";
  char random[5][11] = { 0 };
  char* sort[5] = { 0 }, * temp = NULL;
  printf("문자열 출력 1 : %s\n", arr1);
  printf("문자열 출력 2 : %s\n", arr2);
  printf("문자열 비교  : %d\n", str_cmp(arr1, arr2));
  printf("문자  찾기  : %s\n", str_chr(arr1, 'b'));
  printf("문자  수량  : %d\n", str_cnt(arr1, 'r'));
  printf("\n\n");
  for (int i = 0; i < 5; i++) {
    for (int j = 0; j < 10; j++) {
      random[i][j] = rand() % 3 + 97;
    }
    random[i][10] = '\0';
    sort[i] = random[i];
  }
  for (int i = 0; i < 5; i++) {
    printf("문자열 출력 %d : %s\n", i, sort[i]);
  }
  printf("\n\n");
  for (int i = 0; i < 4; i++) {
    for (int j = i + 1; j < 5; j++) {
      if (str_cmp(sort[i], sort[j]) > 0) {
        swap(char*, sort[i], sort[j]);
      }
    }
  }
  for (int i = 0; i < 5; i++) {
    printf("문자열 출력 %d : %s\n", i, sort[i]);
  }
  printf("\n\n");
  return 0;
}
```

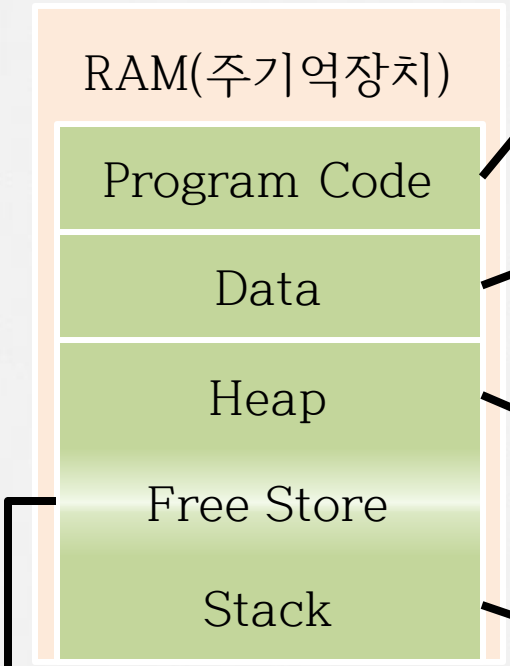
Day 17 - Contents

1. **메모리 할당 및 해제**
   - 메모리의 구조
   - 동적할당
2. **예제 코드**
   -효율적인 메모리 사용코드(이중 포인터)

Day17 - 1. 메모리 할당 및 해제

메모리의 구조

메모리 영역



Code 영역 : 순차적인 명령문(제어문, 함수, 상수, …)의 저장 공간
CPU는 코드 영역에 명령어를 하나씩 가져가서 처리

Data 영역 전역변수, 정적 변수,… 저장 공간
프로그램 실행 전에 선언, 프로그램이 끝날 때 까지 살아있는 변수들

Heap 영역 : 사용자의 동적 할당 (프로그램 실행 중 메모리 할당)
보통 낮은 주소 번지에서 높은 주소 번지로 할당

Stack 영역 : 지역 변수, 매개 변수, ({ } 안에 변수들, …)
함수가 종료되면 해당 함수에 할당된 변수들을 메모리에서 해제
보통 높은 주소 번지에서 낮은 주소 번지로 할당

Free Store : Heap or Stack 메모리 모두 할당 가능
Stack 영역의 메모리가 늘어나면 Heap 영역의 메모리가 줄어들고
Heap 영역의 메모리가 늘어나면 Stack 영역의 메모리가 줄어든다.

Day17 - 1. 메모리 할당 및 해제

메모리의 구조

메모리 영역

	메모리 주소	변수	값
Code	0x0000000000	…	…
	0x0000000001	…	…
	…	…	…
	0xFFFB0EE620	g	103
	0xFFFB0EE621	f	102
	0xFFFB0EE622	e	101
	0xFFFB0EE623	d	100
	0xFFFB0EE624	c	99
	0xFFFB0EE625	b	98
	0xFFFB0EE626	a	97
	…	…	…
	0xFFFFFFFFE	…	…
Stack	0xFFFFFFFFF	…	…

Day17 - 1. 메모리 할당 및 해제

동적 할당

한정된 자원 메모리를 효율적으로 사용하기 위한 방법 -> 동적 할당의 필요성
프로그램 실행 중 수요에 맞게 메모리를 할당, 해제하는 방법이 동적 할당.

형식
(void *)malloc(size_t s);
//s byte 메모리를 할당, 성공 : 메모리 시작주소 (void *) 반환, 실패 : NULL 반환
(void) free(void *p); //p가 가리키는 메모리 해제

배열의 포인터
int * score1;
score1 = (int*)malloc(sizeof(int) * n); // int score1[n];
//sizeof(int) * n byte의 메모리를 할당, 메모리의 시작 주소(void *)를 반환 한다.
//시작 주소(void *)를 정수형 포인터(int *)로 형 변환 후 score 대입
free(score1); //score1이 가리키는 메모리 해제

포인터 배열의 포인터
int ** score2;
score2 = (int**)malloc(sizeof(int*) * n);// int* score2[n];
free(score2); //score2가 가리키는 메모리 해제

Day17 - 2. 예제 코드

//N개 데이터 평균 분산 구하기

```
#include <stdio.h>
#include <stdlib.h> //malloc(), free()
int main() {
    int i, n;
    int* score; //동적 할당 받을 포인터 변수
    float sum, average, variance;
    printf("몇 명의 성적을 입력 하나요? ");
    scanf_s("%d", &n);
    score = (int*)malloc(sizeof(int) * n);
    //int(4byte)*n의 메모리 할당 - 메모리 시작주소 반환(void *) - (void * -> int *) 형 변환
    //int score[n]; 같은 의미 ( 증감연산자 사용X )
    for (i = 0; i < n; i++) {
        printf("학생 %2d : ", i + 1);
        scanf_s("%d", &score[i]); //score +i 같은 표현
    }
    for (sum = 0.0, i = 0; i < n; i++) sum += score[i];
    average = sum / n;
    for (sum = 0.0, i = 0; i < n; i++)sum += (score[i] - average) * (score[i] - average);
    variance = sum / n;
    printf("평균과 분산은 %f, %f 이다\n", average, variance);
    free(score);// 동적 할당 받은 메모리 해제
    return 0;
}
```

Day17 - 2. 예제 코드

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct score {
    int kor, eng, mat, total;
    char* memo; //문자열 주소를 저장
}score;
int main() {
    char buf[1024];
    score* s;
    int number, i;
    printf("학생 수 입력 : ");
    scanf_s("%d", &number);
    s = (score*)malloc(sizeof(score) * number);
    // 구조체 n개 메모리 할당, 그 시작 주소를 반환
    // score s[n];
    for (i = 0; i < number; i++) {
        printf( " \n%d번째 입력\n " , i + 1);
        printf( " 국어 : " );
        scanf_s( " %d " , &s[i].kor);
        printf( " 영어 : " );
        scanf_s( " %d " , &s[i].eng);
        printf( " 수학 : " );
        scanf_s( " %d " , &s[i].mat);
        while (getchar() != ' \n ' );
        printf( " 메모 : " );
        scanf_s( " %s " , buf, (int)sizeof(buf));
        while (getchar() != ' \n ' );
        s[i].memo = (char*)malloc(strlen(buf) + 1);
        //buf에 입력된 문자열 길이 + 1 (\0)만큼 메모리 할당
        // 할당 받은 메모리의 시작 주소를 반환
        // char s[i].memo[buf+1];
        strcpy_s(s[i].memo, (strlen(buf) + 1), buf);
        s[i].total = s[i].kor + s[i].eng + s[i].mat;
    }
    printf("\n\n *** 학생 평균 ***\n");
    for (i = 0; i < number; i++) {
        printf("%d번. %.2f점\n", i + 1, (double)s[i].total / 3);
        printf("메모 : %s\n", s[i].memo);
    }
    printf("\n");
    for (i = 0; i < number; i++) {
        free(s[i].memo); // 구조체 배열 크기만큼 해제
    }
    free(s); //s가 가리키는 메모리 해제
    return 0;
}
```

Day17 - 2. 예제 코드

한 학년 학생의 성적 데이터 관리 프로그램 작성

학년 총 분반 수 입력

1반의 학생수 입력
1반의 학생수 만큼 성적 입력 // 랜덤함수 사용

2반의 학생수 입력
2반의 학생수 만큼 성적 입력 // 랜덤함수 사용

3반의 학생수 입력
3반의 학생수 만큼 성적 입력 // 랜덤함수 사용

…반의 학생수 입력
…반의 학생수 만큼 성적 입력 // 랜덤함수 사용

결과 출력

반	학생수	평균
1반	20	52.12
2반	19	54.04
3반	22	57.56
4반	24	51.41
5반	17	58.27
6반	21	53.39
…		
n반	20	55.70

Day17 - 2. 예제 코드

메모리를 가리키는 포인터	배열								
no_student = (int*)malloc(no_class * sizeof(int));//int no_student[no_class]; 분반 별 학생 수 값 배열									
분반 별 학생 수 (int *)	int	int	…	…	…	…	…	…	int
ave = (float*)malloc(no_class * sizeof(float));//float ave[no_class]; 분반 별 평균 값 배열									
분반 별 평균값 (float *)	float	float	…	…	…	…	…	…	float
var = (float*)malloc(no_class * sizeof(float));//float var[no_class]; 분반 별 분산 값 배열									
분반 별 분산값 (float *)	float	float	…	…	…	…	…	…	float

포인터를 가리키는 포인터	score = (int**)malloc(no_class * sizeof(int*));
	//int * score[no_class];
	//분반 별 성적을 가리키는 포인터 배열								

int** == score	score[i] = (int*)malloc(no_student[i] * sizeof(int));
	//int score[i][no_student]
	//분반 당 학생 인원수의 성적 배열								

1반 학생 성적(int *) s[0]	int	int	…	…	…	…	…	…	int
2반 학생 성적(int *) s[1]	int	int	…	…	…	…	…	…	int
…	int	int	…	…	…	…	…	…	int
n반 학생 성적(int *)s[n-1]	int	int	…	…	…	…	…	…	int

Day17 - 2. 예제 코드

```
#include <stdio.h>
#include <stdlib.h> //malloc(), free(), srand(), rand()
#include <time.h> //time()
void read_score(); // 값 쓰기
void compute_stat(); // 계산
void print_result(); // 출력
void error(char*); // 예외처리
void release_memory(); //동적 할당 해제
int no_class; // 분반 값 변수
int* no_student; // 분반 별 학생수를 가리키는 포인터
float* ave, * var; // 분반 별 평균값, 분산값 가리키는 포인터
int** score; // 분반 별 성적을 가리키는 포인터 배열
float tot_ave, tot_var; // 전체 평균 수치
int main() {
  read_score(); // 성적 입력
  compute_stat(); // 평균값, 분산값 계산
  print_result(); // 출력
  return 0;
}
void read_score() {
  srand((unsigned)time(NULL));
  printf("분반 수를 입력하세요.:");
  scanf_s("%d", &no_class);

  no_student = (int*)malloc(no_class * sizeof(int));
  //int no_student[no_class]; 분반 별 학생 수 값 배열
  ave = (float*)malloc(no_class * sizeof(float));
  //float ave[no_class];  분반 별 평균 값 배열
  var = (float*)malloc(no_class * sizeof(float));
  //float var[no_class];  분반 별 분산 값 배열
  score = (int**)malloc(no_class * sizeof(int*));
  //int * score[no_class]; 분반 별 성적을 가리키는 포인터 배열

  if (no_student == NULL || ave == NULL || var == NULL || score == NULL)
  { error("메모리 부족"); }

  for (int i = 0; i < no_class; i++) {
    printf(" %d 반의 학생 수를 입력하세요.", i + 1);
    scanf_s(" %d", &no_student[i]);

    score[i] = (int*)malloc(no_student[i] * sizeof(int));
    //int score[i][no_student] 분반당 학생 인원수의 성적 배열
    if (score[i] == NULL) { error("메모리 부족"); }

    printf(" %d 반 %d명 학생의 성적을 입력하세요.\n", i + 1, no_student[i]);
    for (int j = 0; j < no_student[i]; j++) {
      score[i][j] = rand() % 100 + 1;
      printf("%4d\n", score[i][j]);
    }
  }
}
void compute_stat() {
  int i, j;
  float sum, total_ave = 0.0, total_var=0.0;
  for (i = 0; i < no_class; i++) {
    for (sum = 0.0, j = 0; j < no_student[i]; j++) {
      sum += score[i][j];
    }
    ave[i] = sum / no_student[i];
    total_ave += ave[i];
    for (sum = 0.0, j = 0; j < no_student[i]; j++) {
      sum += (score[i][j] - ave[i]) * (score[i][j] - ave[i]);
    }
    var[i] = sum / no_student[i];
    total_var += var[i];
  }
  tot_ave = total_ave / no_class;
  tot_var = total_var / no_class;
}
void print_result() {
  printf("%3s %7s %9s %9s\n", "분반", "학생수", "평균", "분산");
  for (int i = 0; i < no_class; i++) {
    printf("%3d %7d %11.2f %11.2f\n", i + 1, no_student[i], ave[i], var[i]);
  }
  printf("%11s %8.2f %11.2f\n","전체 평균 수치",tot_ave, tot_var);
  release_memory();
}
void release_memory() {
  for (int i = 0; i < no_class; i++) free(score[i]);
  free(score);
  free(no_student);
  free(ave);
  free(var);
}
void error(char* message) {
  printf("ERR : %s \n", message);
  exit(1);
}
```

Day 18 - Contents

1. **구조체**
   - 구조체 선언(struct, union, enum, typedef)
   - 구조체와 배열
   - 구조체와 포인터
2. **예제 코드**
   - 구조체 선언, 입력 받기
   - 과제

Day18 - 1. 구조체

구조체 선언(struct, typedef, union, enum)

구조체의 정의

두 개 이상의 변수(자료형)를 묶음으로 새로운 자료형으로 재정의하는 것

<배열 : 같은 자료형의 그룹, 구조체 : 다양한 자료형의 재정의>

형식	예시
struct 구조체명	struct human
{	{
자료형 멤버 1;	char name[10];
자료형 멤버 2;	int age;
…;	};
};	

구조체 선언

```
struct 구조체명 구조체 변수명; // struct human man, woman;
```

구조체 멤버의 참조

```
구조체 변수명 . 멤버명; // man.age;
```

구조체 초기화

```
struct 구조체명 구조체 변수명 = {값1, 값2, …};
// struct human man = { "Michelin", 21 };
```

Day18 - 1. 구조체

```
#include <stdio.h>
#include <string.h>
struct student {
  int  no; // 학번 멤버 변수
  char name[20]; // 이름 멤버 변수
  int  age; // 나이 멤버 변수
  char phone[20]; // 전화번호 멤버 변수
};int main(){
  struct student st1, st2, st3;
  //구조체 변수 선언
  st1.no = 1;
  strcpy_s(st1.name, sizeof(st1.name), "James Robert");
  //st1.name = "James Robert"; 에러!
  st1.age = 35;
  strcpy_s(st1.phone, sizeof(st1.phone), "010-1234-5678");

  st2.no = 2;
  strcpy_s(st2.name, sizeof(st2.name), "William Michael");
  st2.age = 24;
  strcpy_s(st2.phone, sizeof(st2.phone), "010-5678-1234");

  st3.no = 3;
  strcpy_s(st3.name, sizeof(st3.name), "Joshua David");
  st3.age = 20;
  strcpy_s(st3.phone, sizeof(st3.phone), "010-4321-8765");
  printf("%5d  %20s %4d %20s\n", st1.no, st1.name, st1.age,
st1.phone);
  printf("%5d  %20s %4d %20s\n", st2.no, st2.name, st2.age,
st2.phone);
  printf("%5d  %20s %4d %20s\n", st3.no, st3.name, st3.age,
st3.phone);
  return 0;
}
```

결과

```
    1        James Robert   35        010-1234-5678
    2     William Michael   24        010-5678-1234
    3       Joshua David    20        010-4321-8765
```

구조체		
St1	no	1
	name	James Robert
	age	35
	phone	010-1234-5678
St2	no	2
	name	William Michael
	age	24
	phone	010-5678-1234
St3	no	3
	name	Joshua David
	age	20
	phone	010-4321-8765

Day18 - 1. 구조체

구조체 선언(struct, typedef, union, enum)

typedef A B : A 자료 타입의 또 다른 이름 B

형식

typedef 자료형 재정의 명칭

```
typedef unsigned char uchar; //재정의
typedef int * intptr; //재정의
uchar i,j; // 선언
intptr p; // 선언

typedef struct student{
int no;
char name[20];
int score;
int rank;
} STU; //타입 재정의

STU st1,st2,st3; // 선언
```

Day18 - 2. 예제 코드

```c
#include <stdio.h>
typedef struct student {
  int no;
  char name[20];
  int score;
  int rank;
} STU;

int main() {
  STU st;
  printf("번호를 입력하세요 :");
  scanf_s("%d", &st.no);
  printf("이름을 입력하세요 :");
  scanf_s("%s", st.name, (int)sizeof(st.name));
  printf("점수를 입력하세요 :");
  scanf_s("%d", &st.score);
  printf("순위를 입력하세요 :");
  scanf_s("%d", &st.rank);
  printf("\n");
  printf("============================\n");
  printf("번호\t이름\t점수\t순위\n");
  printf("%4d\t%s\t%4d\t%4d\n", st.no, st.name, st.score, st.rank);
  printf("============================\n");
  return 0;
}
```

공용체 선언(struct, typedef, union, enum)

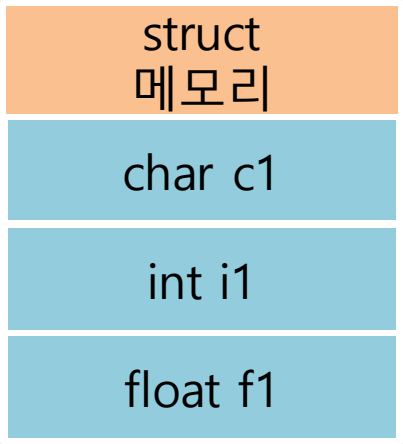
공용체(union) 정의
여러 가지 자료형의 데이터를 하나의 기억 장소에서 공유하며 저장
가장 큰 자료형 1개의 공간만 할당 된다.

형식
```
union 공용체명
{
    자료형 멤버 1;
    자료형 멤버 2;
    …;
};
```

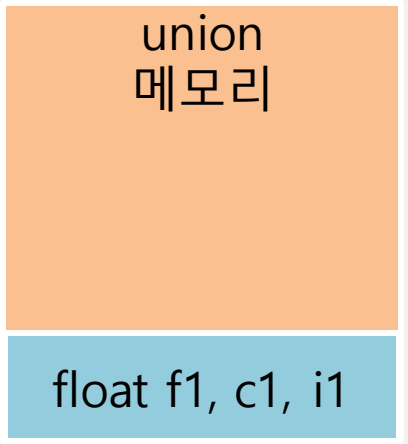
예시
```
union int_or_float
{
    int a;
    float b;
};
```

구조체와 공용체의 차이

```
struct tag{
 char c1;
 int i1;
 float f1;
};
```




```
union tag{
 char c1;
 int i1;
 float f1;
};
```



Day18 - 2. 예제 코드

```
#include <stdio.h>
typedef union Unum{
  double d;
  int i;
} Unum;

typedef struct Snum{
  double d;
  int i;
} Snum;

int main() {
  Unum u_num;
  Snum s_num;
  printf("Union 정수  입력 :");
  scanf_s("%d", &u_num.i);
  printf("Union 정수       : %10d \t \t union  whole size : %3d\n", u_num.i, (int)sizeof(u_num));
  printf("Union 실수  입력 :");
  scanf_s("%lf", &u_num.d);
  printf("Union 정수 (손실): %10d \t \t union    int size : %3d\n", u_num.i, (int)sizeof(u_num.i));
  printf("Union 실수       : %10.2lf \t \t union double size : %3d\n\n", u_num.d, (int)sizeof(u_num.d));

  printf("Struct 정수 입력 :");
  scanf_s("%d", &s_num.i);
  printf("Struct 정수      : %10d \t \t struct  whole size : %3d\n", s_num.i, (int)sizeof(s_num));
  printf("Struct 실수 입력 :");
  scanf_s("%lf", &s_num.d);
  printf("Struct 정수      : %10d \t \t struct    int size : %3d\n", s_num.i, (int)sizeof(s_num.i));
  printf("Struct 실수      : %10.2lf \t \t struct double size : %3d\n\n", s_num.d, (int)sizeof(s_num.d));
  return 0;
}
```

Day18 - 1. 구조체

구조체 선언(struct, union, enum, typedef)

열거형(enum) 정의
문자열을 상수화 시켜 나열하며 정의
프로그램의 가독성 ↑

형식

enum 열거형 이름 { 문자열상수1, 문자열상수2, …};
enum week {SUN, MON, TUE, WED, THU, FRI, SAT};
<SUN = 0부터 시작>

```
#include <stdio.h>
typedef enum week1 { SUN, MON, TUE, WED, THU, FRI, SAT } WEEK;
enum week2 { SU=2, MO, TU, WE=10, TH, FR, SA };
int main() {
    WEEK day1[7] = { SUN, MON, TUE, WED, THU, FRI, SAT };
    enum week2 day2[7] = { SU, MO, TU, WE, TH, FR, SA };
    for(int i=0;i<7;i++){
        printf("week1 : %d \tweek2 : %d \n", day1[i], day2[i]);
    }
    return 0;
}
```

Day18 - 1. 구조체

구조체와 배열

구조체도 재정의된 자료 타입. 배열의 형태로 사용 가능

```
#include <stdio.h>
#include <stdlib.h> //system() : 화면 지우기
#define MAX 3
typedef struct student {
    int no;
    char name[20];
    char phone[20];
} STU;
int main() {
    STU st[MAX]; //STU 구조체 배열 선언
    for (int i = 0; i < MAX; i++) {
        system("cls");
        printf("\n\t\t*** 학생정보 입력 ***\n");
        st[i].no = i + 1; //자동일련번호
        printf("번    호 : %03d", st[i].no);
        printf("\n이    름 : ");
        scanf_s("%s", st[i].name,(int)sizeof(st[i].name));
        while (getchar() != '\n'); //입력 버퍼 비우기
        printf("전화번호 : ");
        scanf_s("%s", st[i].phone, (int)sizeof(st[i].phone));
        while (getchar() != '\n');
    }
    system("cls");
    printf("\n\t\t*** 출 력 결 과 ***\n");
    for (int i = 0; i < MAX; i++) {
        printf("\t%d. 이    름 : %s\n\t   전화번호 : %s\n\n\n", st[i].no, st[i].name, st[i].phone);
    }
    return 0;
}
```

구조체		
st[0]	no	…
	name	…
	phone	…
st[1]	no	…
	name	…
	phone	…
st[2]	no	…
	name	…
	phone	…
…	…	…
	…	…
	…	…

Day18 - 1. 구조체

구조체와 포인터

구조체 포인터를 선언한 경우 이를 참조할 경우
'->' , '*' **연산자**(간접 멤버 접근 연산자)를 이용한다

형식
struct 구조체명 * 포인터 변수명; //human * hpt; (typedef 사용)

```
struct person{
 int no;
 char name[20];
 int score;
 int rank;
 };
 struct person per1;
 struct person * pt;
 pt = &per1;
```

구조체 포인터의 멤버참조
포인터 변수명 -> 멤버명;
pt->name; == (*pt).name == per1.name

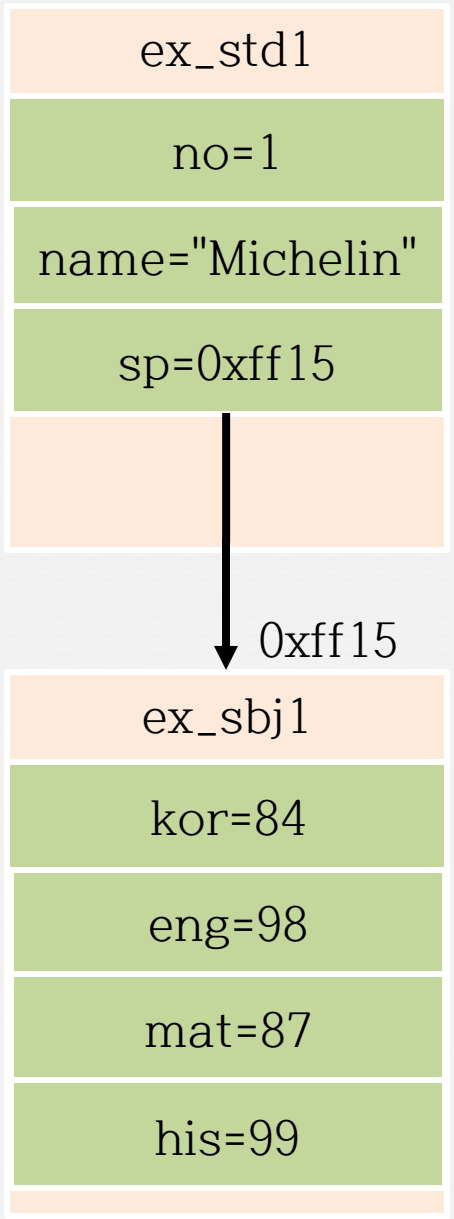
Day18 - 2. 예제 코드

간접 멤버 접근		
포인터		구조체
*p	p->	st.
(*p).kor	p->kor	st.kor
(*p).eng	p->eng	st.eng
(*p).mat	p->mat	st.mat
(*p).sum	p->sum	st.sum

```
#include <stdio.h>
typedef struct score {
  int kor, eng, mat, sum;
}SCO;
int main() {
  SCO st = { 100, 100, 100 };
  SCO* p;
  p = &st;
  printf("%d %d %d %d\n", st.kor, st.eng, st.mat, st.sum);
  printf("%d %d %d %d\n", st.kor, st.eng, st.mat, st.kor + st.eng + st.mat);//sum == 0
  printf("%d %d %d %d\n", p->kor, p->eng, p->mat, p->kor + p->eng + p->mat);//sum == 0
  printf("%d %d %d %d\n", (*p).kor, (*p).eng, (*p).mat, (*p).kor + (*p).eng + (*p).mat);//sum == 0
  printf("%d %d %d %d\n", st.kor, st.eng, st.mat, st.sum);
  p->sum = p->kor + p->eng + p->mat;//sum == 300
  printf("1. sum = %d\n", st.sum);
  printf("2. sum = %d\n", p->sum);
  printf("3. sum = %d\n", (*p).sum);
  return 0;
}
```

Day18 - 2. 예제 코드

```
#include <stdio.h>
typedef struct subject {
  int kor, eng, mat, his;
}subject;
typedef struct student {
  int no;
  char name[20];
  subject* sp;
  //구조체의 멤버로 다른 구조체 포인터 선언
}student;
int main() {
  student ex_std1 = { 1, "michelin" };
  subject ex_sbj1 = { 84, 98, 87, 99 };
  ex_std1.sp = &ex_sbj1;
  printf("번호 : %d\n", ex_std1.no);
  printf("이름 : %s\n", ex_std1.name);
  printf("====================\n");
  printf("과목  점수 \n");
  printf("====================\n");
  printf("국어 : %3d\n", ex_std1.sp->kor);
  printf("영어 : %3d\n", ex_std1.sp->eng);
  printf("수학 : %3d\n", ex_std1.sp->mat);
  printf("국사 : %3d\n", ex_std1.sp->his);
  printf("====================\n");
  return 0;
}
```



Day 19 - Contents

1. **구조체 활용**
   - 구조체와 함수
   - 중첩 구조체
2. **예제 코드**
   - 도서관리프로그램

Day19 - 1. 구조체 활용

구조체와 함수

구조체변수의 연산

구조체 변수는 대입 연산만 가능. (person A, B; A = B ;)

이외 연산은 함수를 만들어 구조체 변수의 멤버접근으로 연산!

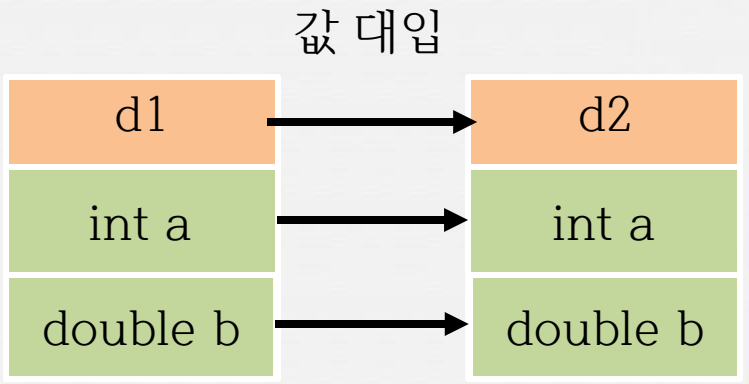
구조체도 자료 타입이므로 함수의 매개변수 or 리턴 가능.

구조체 변수를 함수 호출 시 전달하는 방법

- 값에 의한 전달(Call by Value)
- 레퍼런스에 의한 전달(Call by Reference)

Day19 - 1. 구조체 활용

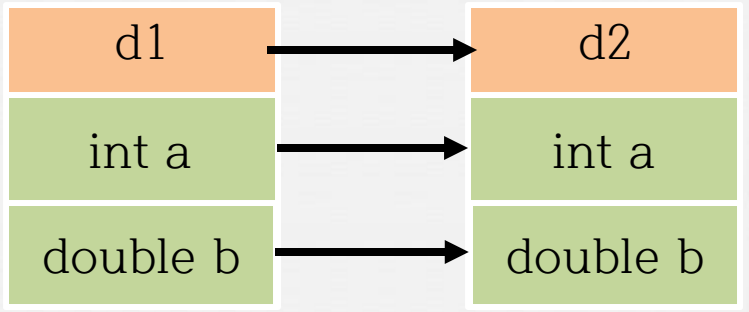
```
#include <stdio.h>
typedef struct data{
  int a;
  double b;
}DATA;
int main(){
  DATA d1 = {10, 10.5};
  DATA d2;
  d2 = d1;
  printf("d1 = a = %d, b = %.2f\n", d1.a, d1.b);
  printf("d2 = a = %d, b = %.2f\n", d2.a, d2.b);
}
```



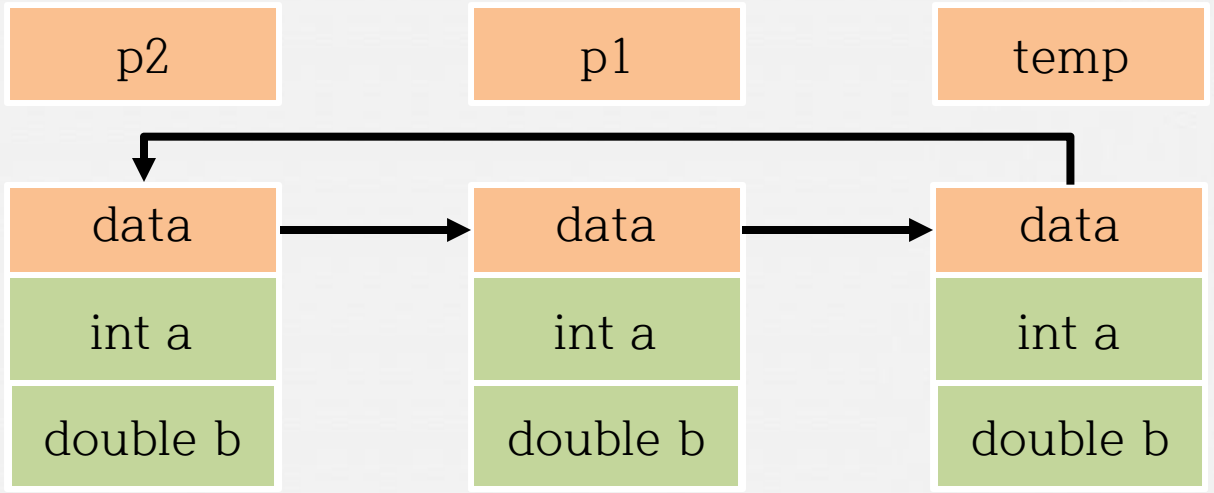
Day19 - 1. 구조체 활용

```
#include <stdio.h>
typedef struct data{
  int v1;
  int v2;
}data;
void swapVal(data d1, data d2){
  data temp;
  temp = d1;
  d1 = d2;
  d2 = temp;
}
void swapRef(data *p1, data *p2){
  data temp;
  temp = *p1;
  *p1 = *p2;
  *p2 = temp;
}
```

Call by Value
값 복사



Call by Reference
주소 복사

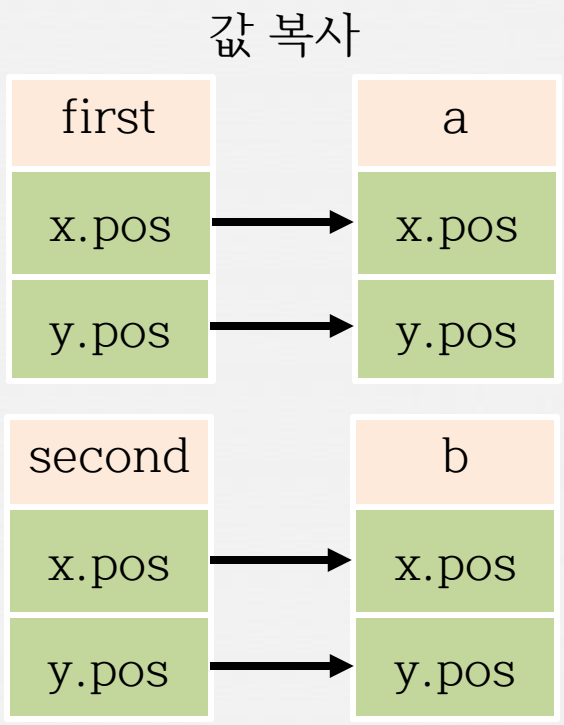


Day19 - 1. 구조체 활용

```
#include <stdio.h>
typedef struct data {
  int v1;
  int v2;
}data;
void swapVal(data d1, data d2) {
  data temp;
  temp = d1;  d1 = d2;  d2 = temp;
}
void swapRef(data* p1, data* p2) {
  data temp;
  temp = *p1;  *p1 = *p2;  *p2 = temp;
}
int main() {
  data d1 = { 10, 20 };
  data d2 = { 100, 200 };
  printf("swapVal 호출 전 d1 = %d %d, d2 = %d %d\n", d1.v1, d1.v2, d2.v1, d2.v2);
  swapVal(d1, d2);
  printf("swapVal 호출 후 d1 = %d %d, d2 = %d %d\n", d1.v1, d1.v2, d2.v1, d2.v2);
  printf("swapRef 호출 전 d1 = %d %d, d2 = %d %d\n", d1.v1, d1.v2, d2.v1, d2.v2);
  swapRef(&d1, &d2);
  printf("swapRef 호출 후 d1 = %d %d,  d2 = %d %d\n", d1.v1, d1.v2, d2.v1, d2.v2);
  return 0;
}
```

Day19 - 2. 예제 코드

```
#include <stdio.h>
//x, y좌표를 표현하는 구조체
typedef struct point {
  int xpos;
  int ypos;     void PointAdd2(point * result, point a, point b) {
}point;         }
point PointAdd(point a, point b) {
  point result;
  result.xpos = a.xpos + b.xpos;
  result.ypos = a.ypos + b.ypos;
  return result;
}
int main(void) {
  point first;
  point second;
  point result;
  printf("첫번째 x, y좌표를 입력해 주세요 : ");
  scanf_s("%d %d", &first.xpos, &first.ypos);
  printf("두번째 x, y좌표를 입력해 주세요 : ");
  scanf_s("%d %d", &second.xpos, &second.ypos);
  result = PointAdd(first, second);
  printf("첫번째와 두번째 좌표를 더한 값 : %d  %d \n\n", result.xpos, result.ypos);
  return 0;
}
```



Day19 - 2. 예제 코드

```
#include <stdio.h>
//x, y좌표를 표현하는 구조체
typedef struct point {
  int xpos;
  int ypos;
}point;
void PointAdd2(point * result, point a, point b){
  result->xpos = a.xpos + b.xpos; // (*result).xpos = a.xpos + b.xpos;
  result->ypos = a.ypos + b.ypos; // (*result).ypos = a.ypos + b.ypos;
}
int main(void) {
  point first;
  point second;
  point result;
  printf("첫번째 x, y좌표를 입력해 주세요 : ");
  scanf_s("%d %d", &first.xpos, &first.ypos);
  printf("두번째 x, y좌표를 입력해 주세요 : ");
  scanf_s("%d %d", &second.xpos, &second.ypos);
  PointAdd2(&result, first, second);
  printf("첫번째와 두번째 좌표를 더한 값 : %d  %d \n\n", result.xpos, result.ypos);
  return 0;
}
```

Day19 - 1. 구조체 활용

중첩 구조체

구조체 자료형을 포함하는 구조체의 재정의

```
struct book{//책 구조체
  int code;
  struct b_data b_info;
  struct p_data * p_info;
};
```

```
struct b_data{//서적
char genre[20];
char book_name[20];
char original[20];
char author[20];
char translator[20];
int price;
int stock;
float rank;
};
```

```
struct p_data{//출판
char publisher_name[20];
int edition;
int date[3];
int page;
char address[20];
char phone[20];
char e_mail[20];
char fax[20];
};
```

도서 관리 프로그램 (구조체 3)

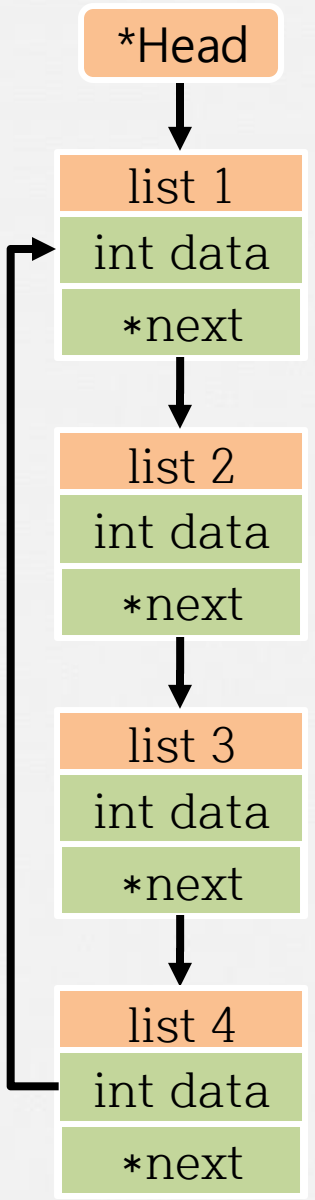
번호	서적정보 (구조체 1)							
	장르	제목	원제	저자	번역가	가격	재고량	평점
	출판사 정보 (구조체 2)							
	출판사	판본	출판년도	쪽수	주소	전화번호	E-mail	Fax

Day19 - 1. 구조체 활용

자기 참조 구조체

포인터를 활용해 자기 자신을 가리키는 구조체

```
#include <stdio.h>
//struct list {
//  int data;
//  struct list next; // 에러
//};
typedef struct list {
  int data;
  struct list* next;
} list;
int main() {
  list list1, list2, list3, list4, * head = &list1;
  list1.data = 111111;  list2.data = 222222;
  list3.data = 333333;  list4.data = 444444;
  list1.next = &list2;  list2.next = &list3;  list3.next = &list4;  list4.next = &list1;
  while (head) {
    printf("%d\n", head->data);
    head = head->next;
  }
  return 0;
}
```



Day19 - 1. 구조체 활용

도서관리 프로그램 작성

```
typedef struct _book
{
    int no;// 책 일련번호
    char title[30]; //도서명
    char author[20]; //작가
    int price; //가격
    int cnt; //수량
    int sale; //매출액(가격 * 수량)
}BOOK;
```

```
printf("\n *** 도서관리 메뉴 ***\n\n");
printf("  1. 입력\n");
printf("  2. 출력\n");
printf("  3. 정렬\n");
printf("  4. 검색\n");
printf("  5. 종료\n\n");
printf("  선택 : [ ]\b\b");
scanf_s("%d", &n);
```

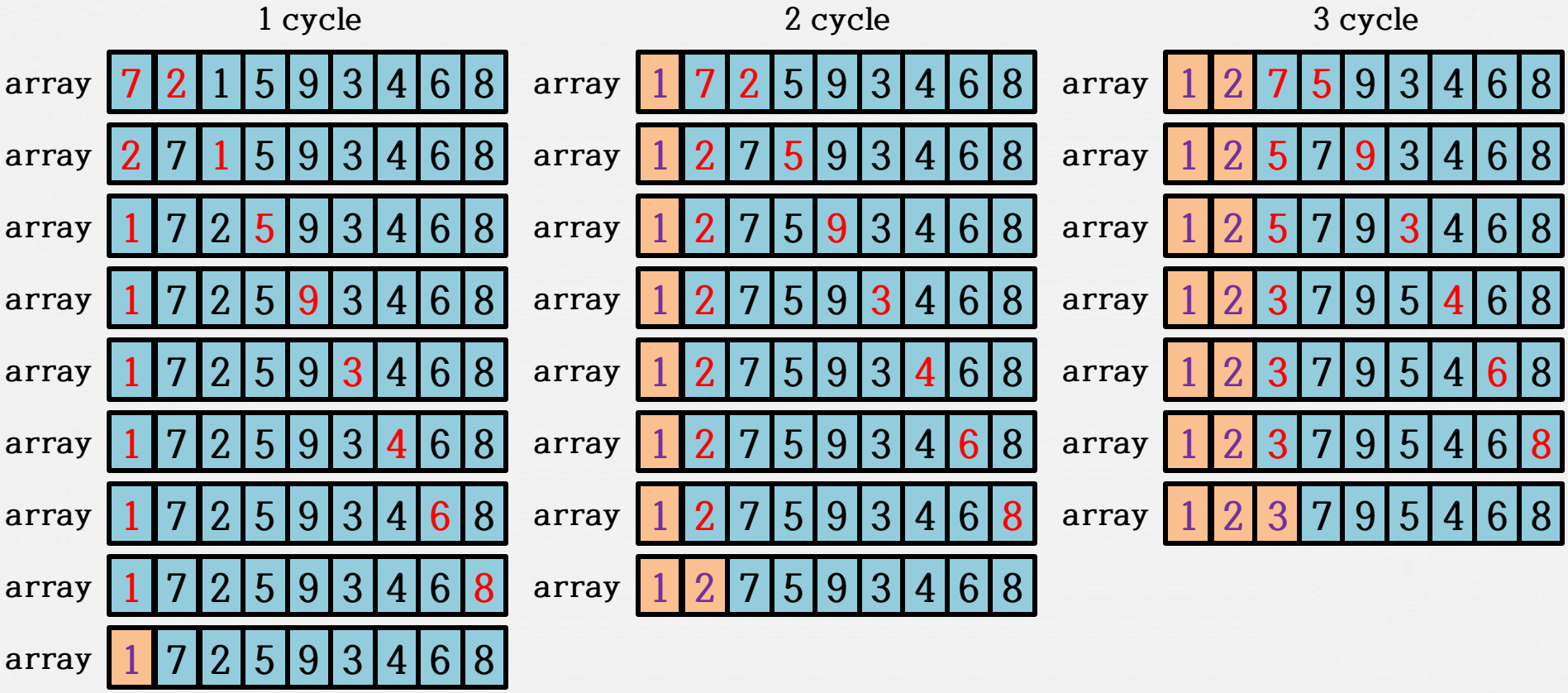
```
void input(BOOK* p);  //구조체 데이터 입력함수
void output(BOOK* p); //출력함수
void sort(BOOK* p);   //작가명 a ~ z 정렬함수
void search(BOOK* p); //책이름 검색함수
```

Day19 - 2. 예제코드

```
#include <stdio.h>
#include <stdlib.h> //system(), exit()
#include <conio.h>  //_getch()
#include <string.h> //strcmp()
#define MAX 3


typedef struct _book
{
    int no; // 책 일련번호
    char title[30];    //도서명
    char author[20];   //작가
    int price;         //가격
    int cnt;           //수량
    int sale;          //매출액(가격 * 수량)
}BOOK;


void input(BOOK* p);  //입력함수
void output(BOOK* p); //출력함수
void sort(BOOK* p);   //정렬함수
void search(BOOK* p); //검색함수


int main() {
    int n;
    BOOK bo[MAX];
    while (1) {//무한루프   for(;;) 무한루프
        system("cls");//화면 지우기
        printf("\n *** 도서관리 메뉴 ***\n\n");
        printf("  1. 입력\n");
        printf("  2. 출력\n");
        printf("  3. 정렬\n");
        printf("  4. 검색\n");
        printf("  5. 종료\n\n");
        printf("  선택 : [ ]\b\b");
        scanf_s("%d", &n);

        switch (n) {
        case 1:
            input(bo);
            break;
        case 2:
            output(bo);
            break;
        case 3:
            sort(bo);
            break;
        case 4:
            search(bo);
            break;
        case 5:
            printf("\n\n\t\t프로그램을 종료합니다.\n");
            exit(0); //프로그램 강제종료
        }//end switch
        printf("\n\n\t\t아무키나 누르면 메뉴로 돌아갑니다.");
        _getch(); //입력함수(에코기능X, Enter필요없음) 대기 목적
    }//end while
    return 0;
}//end main


void input(BOOK* p) {
    for (int i = 0; i < MAX; i++) {
        system("cls");
        printf("\n\n*** 도서 정보 입력 ***\n\n");
        p[i].no = i + 1; //일련번호 자동할당
        printf("[%d번째 입력]\n", p[i].no);
        printf("도서명 : ");
        scanf_s("%s", p[i].title, (int)sizeof(p[i].title));
        printf("작  가 : ");
        scanf_s("%s", p[i].author, (int)sizeof(p[i].author));
        printf("가  격 : ");
        scanf_s("%d", &p[i].price);
        printf("수  량 : ");
        scanf_s("%d", &p[i].cnt);
        p[i].sale = p[i].price * p[i].cnt;   //매출액 = 가격 * 수량
    }
}//end input

void output(BOOK* p) {
    system("cls");
    printf("\n\n\t\t*** 도서 정보 출력 ***\n\n");
    printf("%5s %27s %20s %7s %5s %8s\n", "번호", "도서명", "
작가", "가격", "수량", "매출액");
    printf("-----------------------------------------
--------------------------------\n");
    for (int i = 0; i < MAX; i++) {
        printf("  %03d %27s %20s %7d %5d %8d\n", p[i].no,
p[i].title, p[i].author, p[i].price, p[i].cnt, p[i].sale);
    }
}//end output

void sort(BOOK* p) {
    BOOK temp;
    for (int i = 0; i < MAX - 1; i++) { //순차정렬
        for (int j = i + 1; j < MAX; j++) {//최소값 찾기
            //if( p[i].sale < p[j].sale ) //매출액(내림차순)
            if (strcmp(p[i].title, p[j].title) > 0) { //도서명(오름차순)
                //구조체 전체 데이터를 바꾸기
                temp = p[i];
                p[i] = p[j];
                p[j] = temp;
            }
        }
    }
    printf("\n\n\t\t도서명 기준 오름차순 정렬 완료~!!\n");
}//end sort


void search(BOOK* p) {
    char str[30];
    int flag = 0; //검색여부 확인변수
    system("cls");
    printf("\n\n검색 도서명 입력 : ");
    scanf_s("%s", str, (int)sizeof(str));
    printf("\n\n\t\t*** 도서 검색 정보 출력 ***\n\n");
    printf("%5s %27s %20s %7s %5s %8s\n", "번호", "도서명", "
작가", "가격", "수량", "매출액");
    printf("-----------------------------------------
--------------------------------\n");
    for (int i = 0; i < MAX; i++) {
        if (strcmp(str, p[i].title) == 0) {//도서명이 일치한다면
            printf("  %03d %27s %20s %7d %5d %8d\n", p[i].no,
p[i].title, p[i].author, p[i].price, p[i].cnt, p[i].sale);
            flag = 1; //검색 되었을 때 flag값을 1로 변경
        }
    }//end for
    if (flag == 0) {
        printf("\n\n\t\t검색 도서가 존재하지 않습니다.\n");
    }
}//end search
```

Day19 - 2. 예제코드

순차정렬
맨 앞에서부터 제일 작은 원소를 배치하게 만들어 나가는 알고리즘
배치할 자리에 있는 원소를 뒤쪽에 있는 원소들과 비교
작은 것을 발견하면 배치할 위치의 원소와 교환
위 과정을 n-1번 반복, 정렬 수행
1회전을 수행하고 나면 가장 작은 값의 자료가 맨 앞에 위치한다.
다음 회전에서는 두 번째 자료를 가지고 비교 마찬가지로 3회전에서는 세 번째 자료를 정렬, …

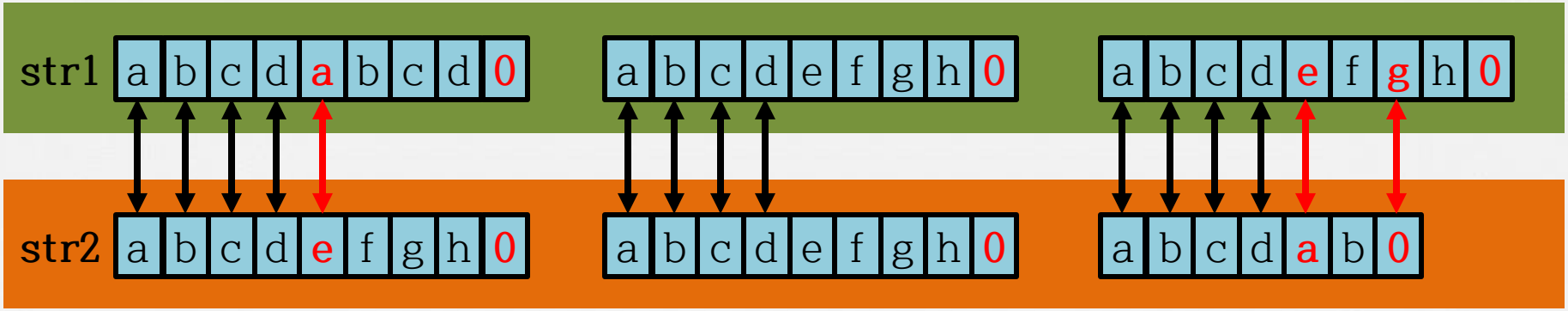


Day19 - 2. 예제코드

strcmp() 문자열 비교 함수
int strcmp(const char* str1, const char* str2)
int strncmp(const char* str1, const char* str2, size_t n) //N개의 문자열만 비교
매개변수로 들어온 두 개의 문자열(str1, str2)을 비교 하여
문자열이 완전히 같다면 0을 반환, 다르면 음수 혹은 양수를 반환하는 함수

(1) str1 < str2 인 경우에는 음수 반환
(2) str1 > str2 인 경우에는 양수 반환
(3) str1 == str2 인 경우에는 0을 반환

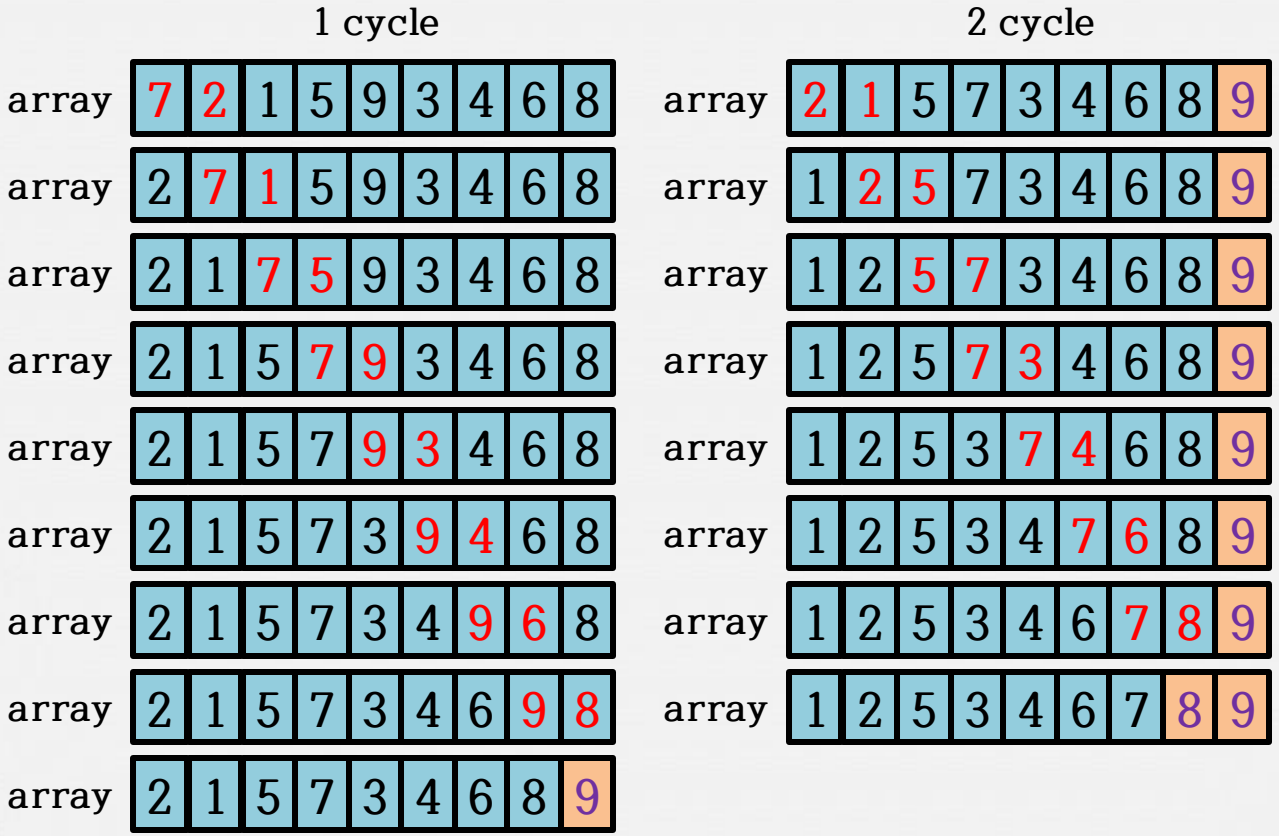
- 대소문자 구분
- 각 문자 별 정수 값이 매칭[아스키 코드 표] 대소 비교 가능



return -1;　　return 0;　　return 1;

Day19 - 2. 예제코드

버블정렬
첫 번째 자료와 두 번째 자료를 비교해 더 큰 값을 오른쪽으로 이동 [오름차순 or 작은 값(내림차순)]
두 번째 자료와 세 번째 자료 비교, 세 번째와 네 번째, … (n-1), n번째 자료를 비교 후 교환
1회전을 수행하고 나면 가장 큰 자료가 맨 뒤로 이동
2회전에서는 맨 끝에 있는 자료는 정렬에서 제외 (n-1번 반복)
2회전을 수행하고 나면 끝에서 두 번째 자료까지는 정렬에서 제외(n-2번 반복)
1회전 수행할 때마다 정렬에서 제외되는 데이터가 하나씩 늘어난다.



Day 20 - Contents

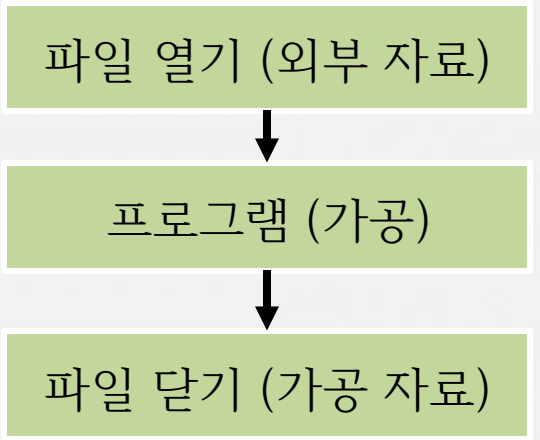
1. **파일 입출력**
   - FILE 포인터
   - 파일 열기, 닫기
   - 문자열 입출력
2. **예제 코드**
   - 단일 문자 fputc(), fgetc()
   - 문자열 fputs(), fgets()
   - 서식문자 fprintf(), fscanf()

Day20 - 1. 파일 입출력

FILE 포인터

파일 입, 출력 정의
보조기억장치에 파일(file) 자료를 입, 출력한다.
데이터 입 . 출력 모드
1. 텍스트 모드
 - 텍스트 기반의 데이터 처리용도
   프로그램상에서 파일로 읽고 쓰는 경우, 데이터 변환 이후 입 . 출력

2. 2진 모드
 - 영상, 사진…등 손실 or 변환이 없어야 하는 경우
   데이터 변환 없이 데이터 입 . 출력



FILE(구조체)
파일에 대한 여러 가지 정보를 가진 구조체
입출력 함수들은 모두 이 구조체의 내용을 참조 수행.

반드시 파일 구조체 포인터를 선언한 후,
파일 입출력을 하기 전에 입출력 함수에게 포인터(주소)를 전달.

Day20 - 1. 파일 입출력

파일 열기, 닫기

파일 열기 fopen_s(FILE** const char* _filename, const char* _mode);

```
FILE* fp=NULL; //예시
if ( fopen_s(&fp, "out.txt", "w") == 0 ) { //fopen_s 성공 시 0 반환
//파일 열기 성공 가공 코드 작성
  fclose(fp); // "out.txt" 파일 닫기
 }
```

파일 닫기 fclose(FILE *); 버퍼에 남아있는 데이터를 파일로 완전히 출력
내부적으로 생성했던 FILE 구조체를 해제

사용 모드	의미
r (읽기 전용)	파일 불러오기(읽기), 해당 파일 필요
w(쓰기 전용)	새로 생성, 이름이 같으면 덮어쓰기
a(추가 전용)	새로 생성, 이름이 같으면 내용 뒤에 추가
r+(갱신용)	읽고 쓰기용, 새로 생성, 이름이 같으면 덮어쓰기
w+(갱신용)	읽고 쓰기용, 새로 생성, 이름이 같으면 덮어쓰기
a+(갱신용)	파일 맨 끝부터 작업이 필요할 경우 사용 새로 생성, 이름이 같으면 데이터 끝부터 추가

Day20 - 1. 파일 입출력

EOF (End Of File)

C언어에서 운영체제와 상관없이 파일의 끝을 알리는 반환값(-1)을 발생.
키보드를 통한 입력을 받을 때에도 입력의 끝을 알려주는 방법이 필요

EOF 발생

- 대부분의 유닉스(UNIX) 시스템에서는 라인의 시작 위치에서 (Enter)
  (어떠한 문자도 없는 상태에서)

- Ctrl+D
- Ctrl + Z (console)

-> Enter

Day20 - 1. 파일 입출력

문자열 입출력

fputc
파일로 단일 문자를 출력하는 함수

형식
int fputc(int c,FILE *stream);
출력한 문자를 반환, 에러 발생할 경우 EOF(-1) 반환

fgetc()
파일로부터 단일 문자 입력 받는 함수

형식
int fgetc(FILE *);
파일로부터 한 문자씩 읽고 그 문자를 반환.
파일의 끝을 만나거나 에러가 발생하면 EOF(-1) 반환.

Day20 - 1. 파일 입출력

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h> //_getch()
int main() {
  FILE* fp = NULL;
  //파일 포인터 선언 & 초기화
  int ch;
  if (fopen_s(&fp, "out.txt", "w") == 0) {
  //fopen_s 성공 시 0 반환, 쓰기모드
     printf("문자 입력(종료:0)\n");
     do {//단일 문자 입력
        ch = _getch();
        printf("입력된 문자 : %c\n", ch);
        fputc(ch, fp);
     //스트림에 문자 출력
     } while (ch != '0');
     printf("\n \t 프로그램 종료\n");
     fclose(fp); //파일 포인터 닫기
  }
  else printf("File Open Fail\n");
  //파일 생성 실패
```

```
  if (fopen_s(&fp, "out.txt", "r") == 0) {
    //fopen_s 성공 시 0 반환, 읽기모드
    printf("파일 내용\n");
    while ((ch = fgetc(fp)) != EOF) {
//텍스트 파일의 끝을 만나면 EOF 값 반환
      if (ch != '\r') printf("%c", ch);
 // enter = \r 입력(커서 줄 앞으로 이동)
      else printf("\n");
    }
    fclose(fp);//파일 포인터 닫기
  }
  else printf("File Open Fail\n");
  //파일 생성 실패
  return 0;
}
```

Day20 - 1. 파일 입출력

문자열 입출력

fputs()
파일로 문자열을 널문자('\0') 이전까지 출력. <개행 문자('\n')도 출력>
문자열은 반드시 널문자로 끝나야 한다.
출력된 마지막 문자 리턴, 에러는 EOF(-1) 리턴

형식
int fputs(문자열, 파일포인터);
int fputs(char const* string, FILE* _Stream);

fgets()
문자열을 읽어와 char형 배열에 저장하는 함수
개행문자('\n')까지 or 배열의 길이만큼 읽는다.
문자열보다 배열의 길이가 더 길면 개행문자('\n')까지 배열에 저장
행 단위로 읽는 경우 주로 사용된다.
문자열이 저장된 포인터를 리턴. 문자열이 없는 빈 파일은 NULL 리턴

형식
char* fgets(문자열 변수, 입력 최대 문자수, 파일포인터);
char* fgets(char* _Buffer, int _MaxCount, FILE* _Stream);

Day20 - 1. 파일 입출력

```
#include <stdio.h>
#include <stdlib.h>
int main() {
  FILE* fp = NULL;
  //파일 포인터 선언 & 초기화
  char str[1024];
  int ch;
  int i = 0;

  if (fopen_s(&fp, "out2.txt", "w") == 0) {
    //fopen_s 성공 시 0 반환, 쓰기모드
    printf("문자열 입력 (종료 : Enter + Ctrl + Z + Enter)\n");
    while ((ch = getchar()) != EOF) {
 // int getchar(void) unsigned char로 받은 문자를 int로
변환해서 리턴, 에러시 EOF 리턴
 //입력 버퍼의 문자열에서 문자 1개씩 ch에 저장
EOF(Ctrl+Z) 비교 후 루프 실행
 //앞에 다른 문자가 없이 Ctrl + z 키를 누르면 EOF 값 발생
      str[i++] = ch;
    }//입력한 문자열 str에 저장
    str[i] = '\0';
    fputs(str, fp);// \0 까지 str 문자열 파일 출력
    printf("out2.txt 파일에 저장완료\n");
    fclose(fp);//파일 포인터 닫기
  }
  else printf("File Open Fail\n"); //파일 생성 실패
  //out2.txt 문자열 출력

  if (fopen_s(&fp, "out2.txt", "r") == 0) {
    //fopen_s 성공 시 0 반환, 파일생성 - 읽기모드
    while (fgets(str, 1024, fp) != NULL) {
      //fgets 문자열을 읽고 char형 배열에 저장하는 함수
      // 1 행 단위로 str 배열에 문자열 저장
      //문자열이 없는 마지막은 NULL 리턴
      printf("%s", str);
    }
    fclose(fp);//파일 포인터 닫기
  }
  else printf("File Open Fail\n"); //파일 생성 실패
  //out2.txt 문자열 입력
  return 0;
}
```

Day20 - 1. 파일 입출력

문자열 입출력

fprintf()
파일에 서식화된 자료를 기록하기 위해 사용되는 함수

형식
int fprintf(파일포인터, 문자열…);
int fprintf(FILE* _Stream, char const* string);

Day20 - 1. 파일 입출력

```
#include <stdio.h>
#include <stdlib.h>
#define STU_SIZE 3
typedef struct student {
  int no;
  char name[20];
  int kor, eng, mat;
}STU;
int main() {
  STU sp[STU_SIZE];
  FILE* fp = NULL; //파일 포인터 선언 & 초기화
  int i;
  char file_name[80];
  printf("\n * 파일명 입력 (.txt 확장자 입력!) : ");
  scanf_s("%s", file_name, sizeof(file_name) - 1);
  // \0 (NULL) 문자 공간 1 제외
  while (getchar() != '\n');
  for (i = 0; i < STU_SIZE; i++) {
    sp[i].no = i + 1;    //일련변호
    printf("\n번호 : [%03d]\n", sp[i].no);
    printf("이름 : ");
    scanf_s("%s", sp[i].name, sizeof(sp[i].name) - 1);
    while (getchar() != '\n');
    printf("국어 : ");
    scanf_s("%d", &sp[i].kor);
    printf("영어 : ");
    scanf_s("%d", &sp[i].eng);
    printf("수학 : ");
    scanf_s("%d", &sp[i].mat);
    while (getchar() != '\n');
  }
  if (fopen_s(&fp, file_name, "w") == 0) {
    for (i = 0; i < STU_SIZE; i++) {
      fprintf(fp, "%d %s %d %d %d\n", sp[i].no, sp[i].name,
sp[i].kor, sp[i].eng, sp[i].mat);
    }
    printf("\n\n\t\t %s 파일에 입력완료되었습니다...\n",
file_name);
    fclose(fp);                //파일 포인터 닫기
  }
  else printf("File Open Fail\n"); //파일 생성 실패
  return 0;
}
```

Day20 - 1. 파일 입출력

문자열 입출력

fscanf()
파일에서 서식화된 자료를 입력 받기 위해 사용되는 함수

형식
int fscanf(파일포인터, 문자열…);
int fscanf(FILE* _Stream, char const* string);

int feof(FILE* _Stream)
(feof : File End Of Flag)
파일의 끝에서(EOF) 0이 아닌 값을 리턴
파일의 끝이 아니면 0를 리턴

예시 : while (!feof(fp)){ }
파일의 마지막 데이터가 아닌 경우 0을 반환
파일의 끝에 도달했는지 여부를 확인할 때 사용

Day20 - 1. 파일 입출력

```
#include <stdio.h>
#include <stdlib.h>
#define STU_SIZE 20
typedef struct student {
    int no;
    char name[20];
    int kor, eng, mat;
}STU;
int main() {
    STU sp[STU_SIZE];
    FILE* fp = NULL; //파일 포인터 선언 & 초기화
    int i = 0, j;
    char file_name[80];
    printf("\n * 파일명 입력 (.txt 확장자 입력!) : ");
    //성적 데이터 파일 불러오기
    scanf_s("%s", file_name, sizeof(file_name) - 1);
    // \0 (NULL) 문자 공간 1 제외
    while (getchar() != '\n');
    if (0 == fopen_s(&fp, file_name, "r")) {
    //fopen_s 성공 시 0 반환, 파일생성 - 읽기모드
        while (!feof(fp)) {
    //파일의 끝에 도달했는지 여부를 확인할 때 사용,
    //파일의 마지막 데이터가 아닌 경우 0을 반환
            fscanf_s(fp, "%d %s %d %d %d\n", &sp[i].no,
sp[i].name, (int)sizeof(sp[i].name), &sp[i].kor, &sp[i].eng,
&sp[i].mat);
            i++;
        }
        fclose(fp);//파일 포인터 닫기
    }
    else printf("File Open Fail\n"); //파일 생성 실패
    printf("\n\n%7s%10s%8s%8s%8s\n", "번호", "이름", "국어",
"영어", "수학");
    printf("
————————————————————————————————————————
————————————————————————————————\n");
    for (j = 0; j < i; j++) {
        printf("%7d%10s%8d%8d%8d\n", sp[j].no, sp[j].name,
            sp[j].kor, sp[j].eng, sp[j].mat);
    }
    return 0;
}
```

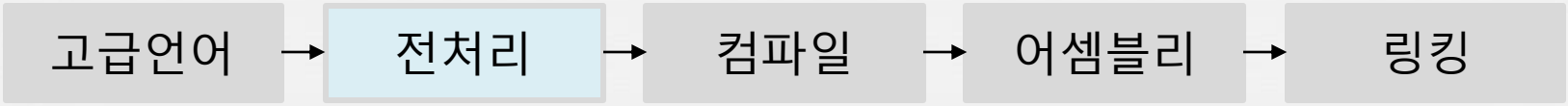
Contents

1. 전처리 지시자

매크로 및 조건부 컴파일 지시자

전처리 정의

컴파일-고급언어를 컴퓨터가 실행 가능하도록 변환 과정



Preprocessor : #include, #define 처리

#include <filename> or "filename"
<filename> 의 경우 특정 경로에서 헤더파일 헤더파일 폴더에서 가져오기

"filename" 의 경우 소스파일과 같은 폴더에서 1차 검색
- <filename> 와 동일한 경로에서 2차 검색

#define token expression // token을 expression으로 대치
유효 범위 : 정의된 라인부터 파일의 마지막 라인까지 (다른 파일에서 접근X)

1. 전처리 지시자

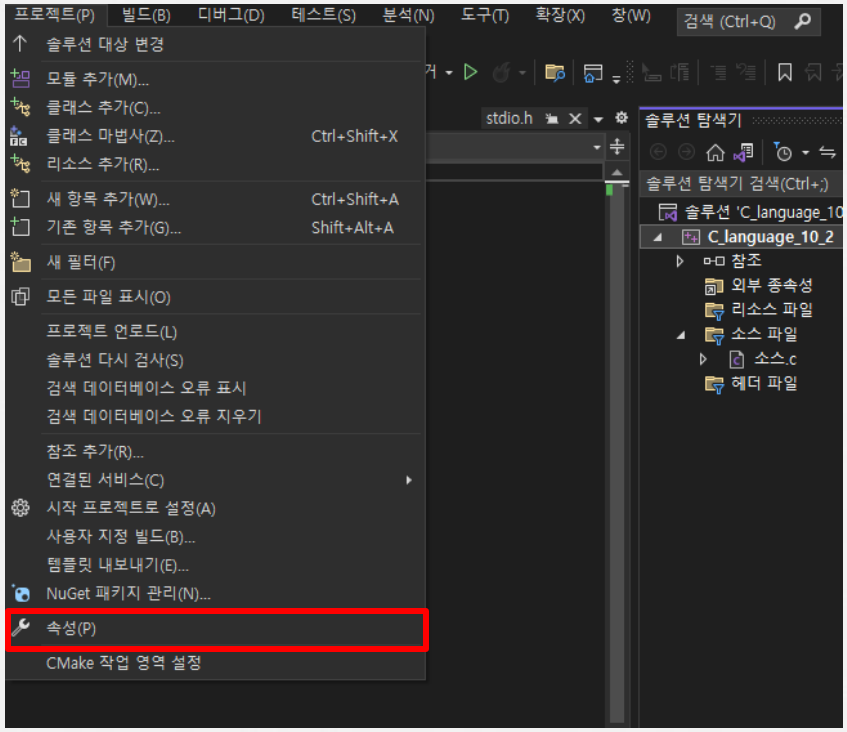
매크로 및 조건부 컴파일 지시자

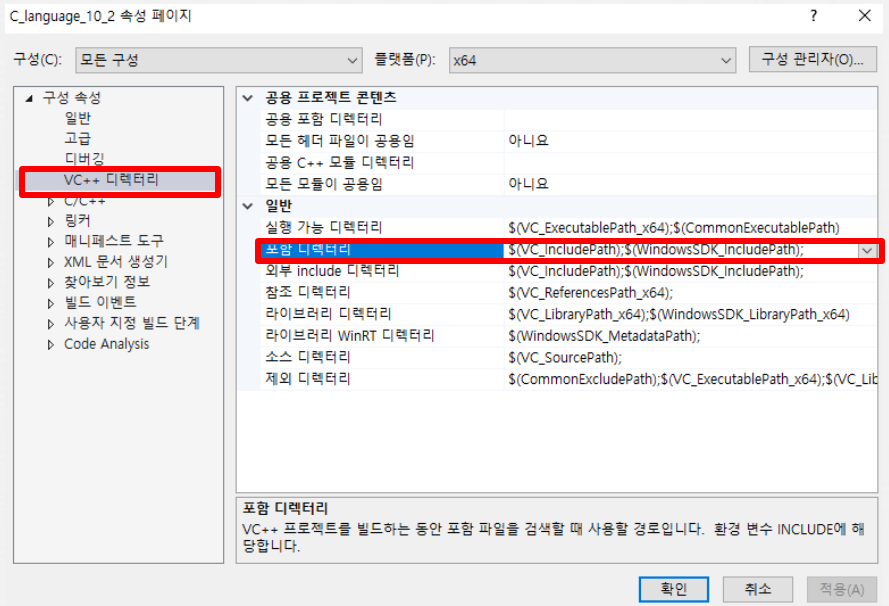
#include <filename> or "filename"

<filename> 의 경우 특정 경로에서 헤더파일 헤더파일 폴더에서 가져오기

"filename" 의 경우 소스파일과 같은 폴더에서 1차 검색

- <filename> 와 동일한 경로에서 2차 검색





1. 전처리 지시자

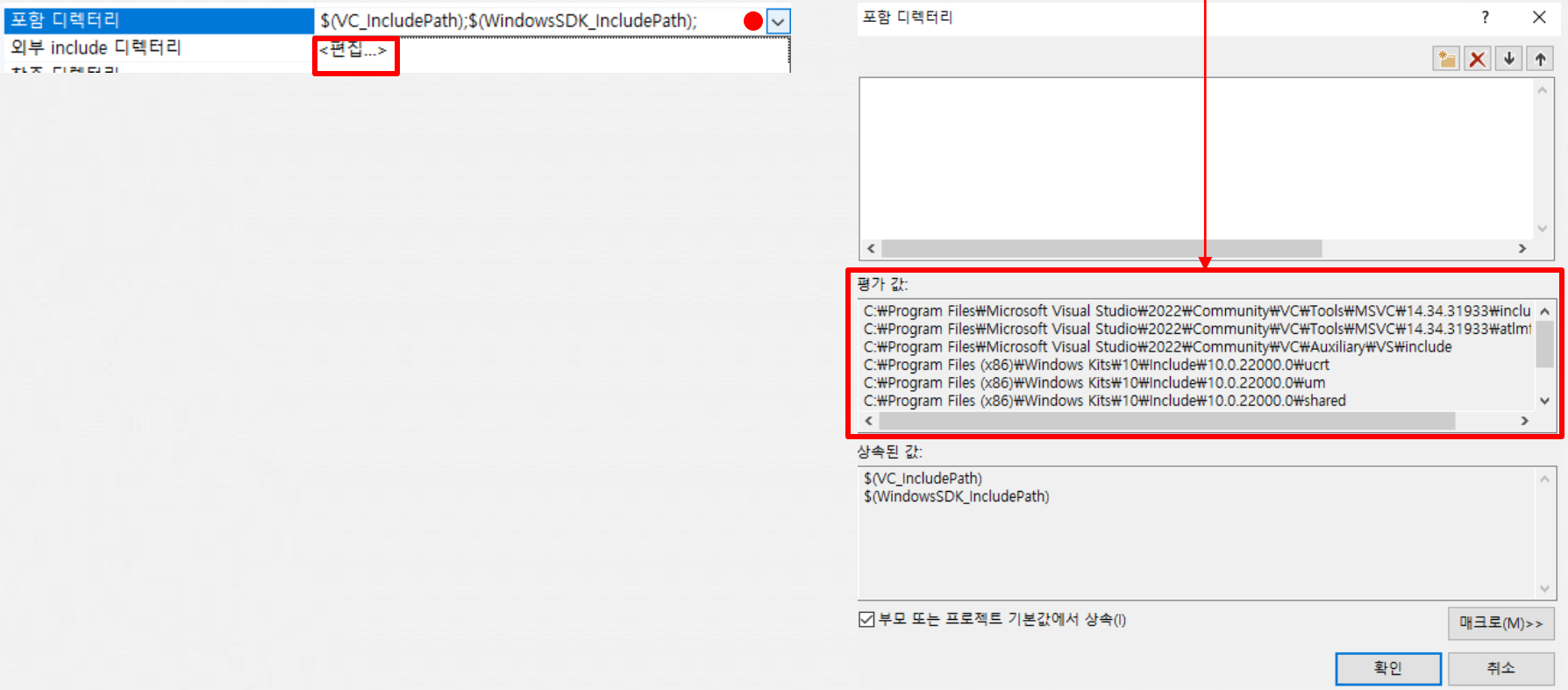
매크로 및 조건부 컴파일 지시자

#include <filename> or "filename"

<filename> 의 경우 특정 경로에서 헤더파일 헤더파일 폴더에서 가져오기

"filename" 의 경우 소스파일과 같은 폴더에서 1차 검색

- <filename> 와 동일한 경로에서 2차 검색



1. 전처리 지시자

매크로 및 조건부 컴파일 지시자

#ifdef _Header_PSK : _Header_PSK 선언되어 있다면 컴파일 포함
#ifndef _Header_PSK : _Header_PSK 선언되지 않았다면 컴파일 포함
전처리 과정에서 조건에 따라 컴파일 대상에 포함시키거나 제외시키는 역할
조건식은 매크로 상수의 존재 여부, 값에 대한 평가식, … 등

#if
매크로의 값이나, 여러 가지 조건을 결합하여 컴파일 여부를 결정

```
#if 조건1
코드1          // 조건1을 만족하면 코드1을 컴파일
#elif 조건2
코드2          // 조건 2가 만족되면 코드2를 컴파일
#else
코드3          // 둘 다 맞지 않으면 코드 3을 컴파일
#endif
```

- 상등, 비교 연산자 사용이 가능하다. ==, != , >, <, >=, <=
- 비교 대상은 정수 상수 가능 (실수, 문자열 불가)
- 논리 연산자 사용 가능

2. 헤더파일 만들기

헤더파일, 소스파일

헤더파일 목록에서 추가 -> 새 항목 -> 헤더파일 선택

파일 이름 header.h 파일 위치는 소스파일과 같은 폴더에 있어야 합니다.

소스 파일에서 #include "헤더파일명.h" 선언

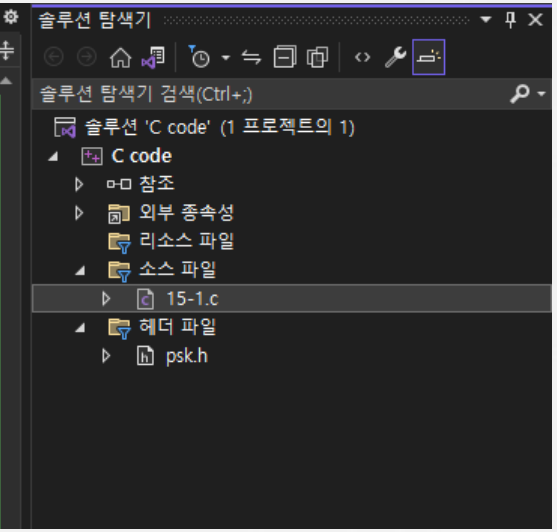
psk.h

```c
//헤더 파일
//include guard
#ifndef _psk_H
// PSK_H 선언되지 않았다면
# define _psk_H
// PSK_H 선언
#include<stdio.h>
# define PI 3.141592
# define SQRT(X) X*X

typedef struct _Person {
    char name[10];
    int age;
    char address[128];
}Person;

typedef long int file_size_t;
extern long long arr_len;

void hello_display() {
    printf("hello world₩n");
}

#endif
```

15-1.c

```c
//소스파일
#define A  1 // 0 or 1
#include<stdio.h>
#include"psk.h"
int main() {
    file_size_t a;
    Person p1;
#if A == 0
    printf("define A == 0₩n");
#elif A == 1
    printf("define A == 1₩n");
#endif
    hello_display();
    return 0;
}
```

솔루션 탐색기



2. 헤더파일 만들기

```
//헤더 파일
//include guard
#ifndef _psk_H
// PSK_H 선언되지 않았다면
# define _psk_H
// PSK_H 선언
#include<stdio.h>
# define PI 3.141592
# define SQRT(X) X*X
typedef struct _Person{
char name[10];
int age;
char address[128];
}Person;
typedef long int file_size_t;
extern long long arr_len;
void hello_display() {
  printf("hello world\n");
}
#endif
```

```
//소스파일
#define A  1 // 0 or 1
#include<stdio.h>
#include"psk.h"
int main() {
  file_size_t a;
  Person p1;
#if A == 0
  printf("define A == 0\n");
#elif A == 1
  printf("define A == 1\n");
#endif
  hello_display();
  return 0;
}
```

3. F12 or ALT + F12 함수 확인하기

#include에서 제공되는 표준 함수 확인 // F12

헤더파일	분류	함수 예
stdio.h	입, 출력	printf, scanf, getchar, …
	파일	fopen, fclose, fprintf, …
conio.h	콘솔 입, 출력	putch, cputs, cprintf, …
string.h	문자열 처리	strcat, strcmp, strcpy, …
math.h	수학	sqrt, sin, cos, tan, log, exp, pow, …
ctype.h	문자 형태 판별	isalpha, isdigit, islower, …
	문자 변환	tolower, toupper, …
stdlib.h	수치 변환	atoi, atoa
	난수 관련	rand, srand
	정렬, 검색	qsqrt, lfind, …
time.h	날짜, 시간	clock, ctime, localtime, mktime, …
…	…	…

https://www.ibm.com/docs/ko/i/7.3?topic=extensions-standard-c-library-functions-table-by-name

4. 디버깅

디버깅 방법 <VS Code>

비교 사용자의 생각 != 프로그램 실행 순서

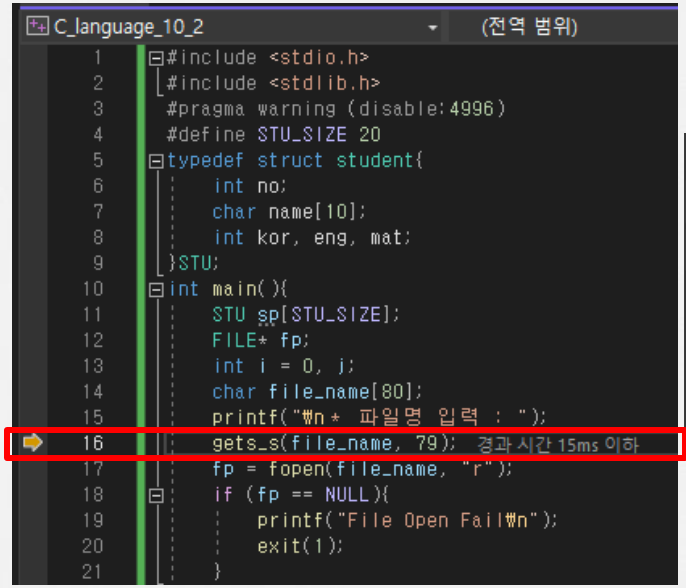
프로그램 순서도 작성	코드 실행 순서 확인
문법 오류	에러, 경고 확인
논리 오류	예외 처리 확인
함수의 기능	인수, 반환 확인
데이터 변화	데이터 값의 변화

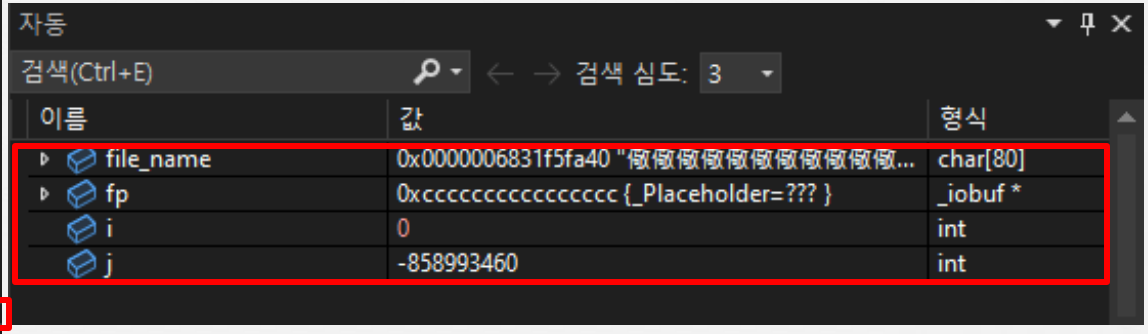
4. 디버깅

디버깅 방법 <VS Code>

한 줄 실행 (F11)

코드 한 줄 씩 실행하면서 데이터 값의 변화와 프로그램이 처리되는 과정을 확인





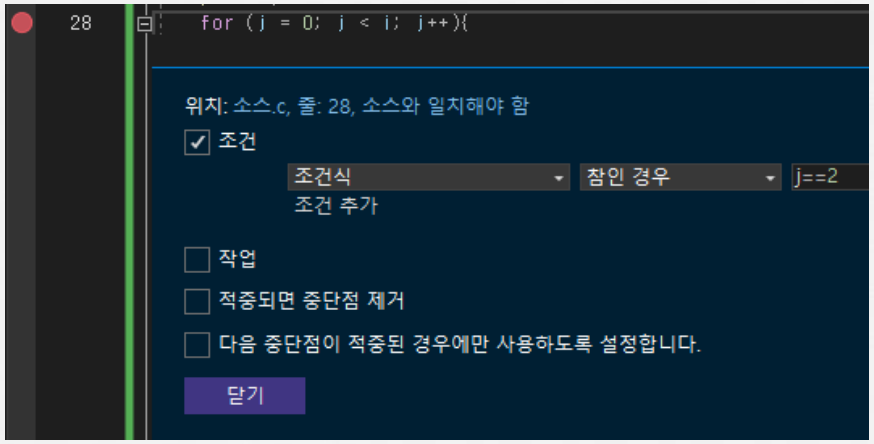
4. 디버깅

디버깅 방법 <VS Code>

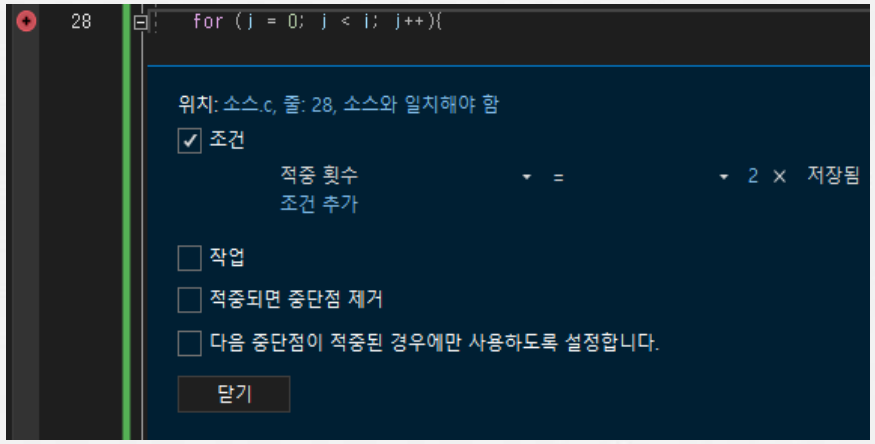
중단점 사용 (F9)
반복 횟수나 규모가 커지는 프로젝트에서 진행 할 때 사용

조건식
Bool 조건 (논리식 : 참 or 거짓)으로 중단점 확인

적중 횟수
코드의 루프가 특정 횟수를 반복한 후 잘못된 동작을 의심되는 경우,
확인 하고 싶은 횟수만큼 적중된 후 실행을 중지



조건식

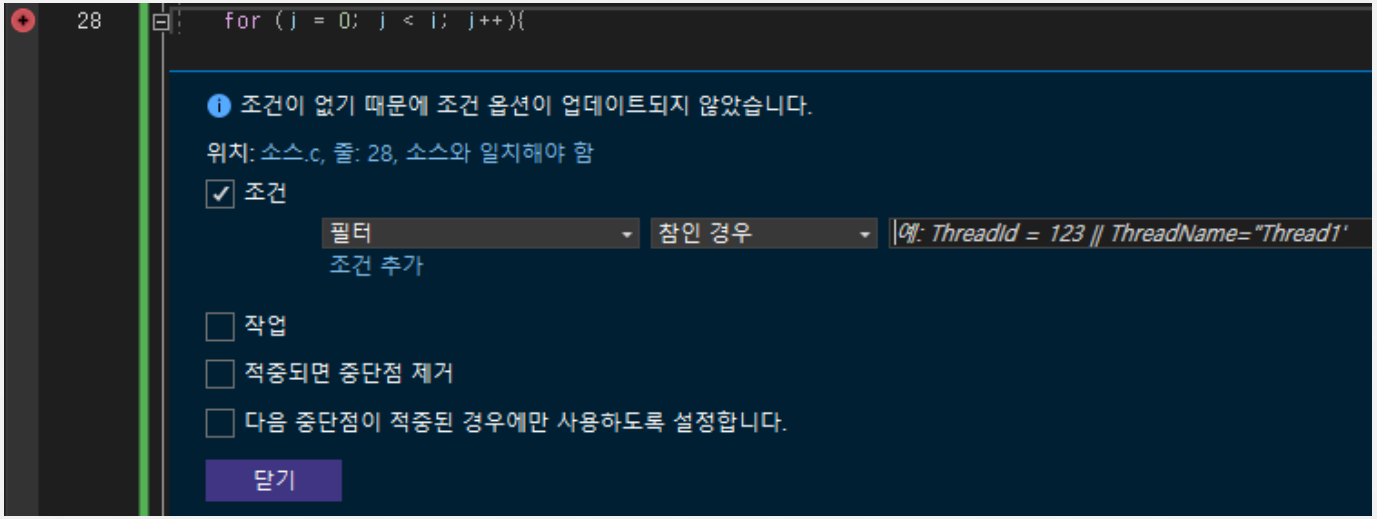


적중 횟수

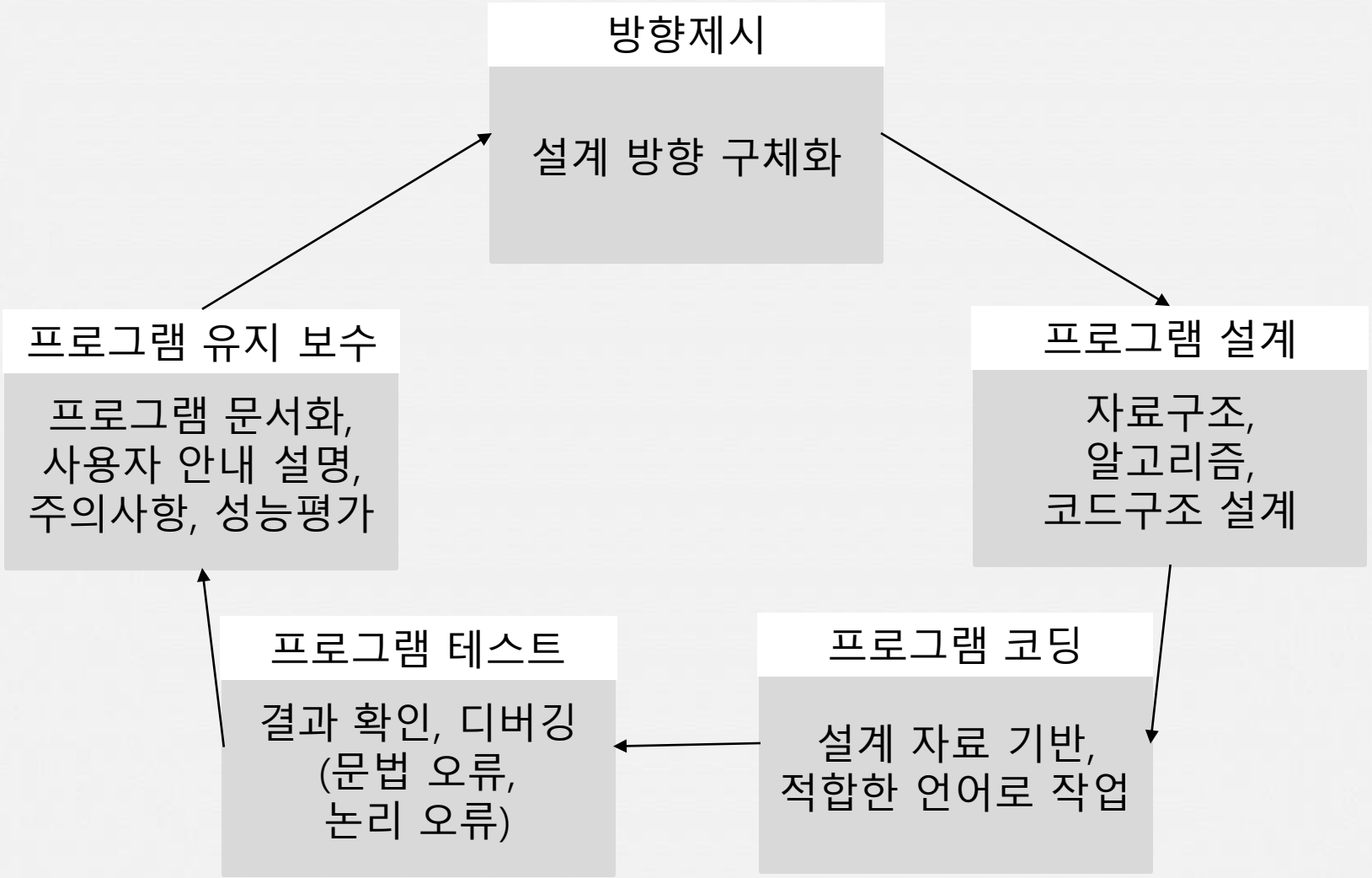
4. 디버깅

디버깅 방법 <VS Code>

중단점 사용 (F9)
필터 조건 (MachineName, ProcessID및 ProcessName, ThreadID 및 ThreadName)
지정된 디바이스, 지정된 프로세스, 스레드에서만 발생하도록 중단점을 제한
MachineName
데이터베이스 엔진 인스턴스를 실행 중인 컴퓨터
ProcessID및 ProcessName
데이터베이스 엔진 인스턴스를 실행 중인 운영 체제 프로세스
ThreadID 및 ThreadName
데이터베이스 엔진 인스턴스에서
Transact-SQL 일괄 처리, 프로시저 또는 함수를 실행 중인 운영 체제 스레드



4. 프로그램 설계 방법


방향제시
설계 방향 구체화
프로그램 설계
자료구조,
알고리즘,
코드구조 설계
프로그램 코딩
설계 자료 기반,
적합한 언어로 작업
프로그램 테스트
결과 확인, 디버깅
(문법 오류,
논리 오류)
프로그램 유지 보수
프로그램 문서화,
사용자 안내 설명,
주의사항, 성능평가