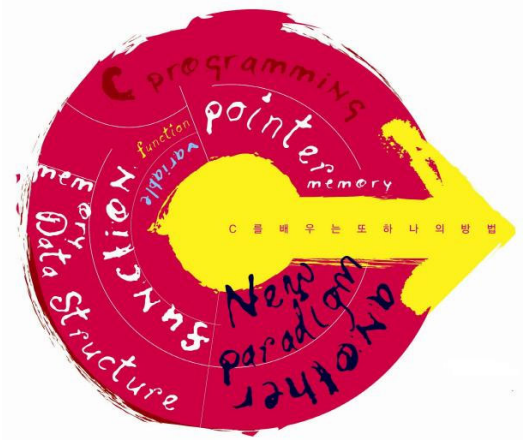
Clickseo.com

C Programming

C 표준 라이브러리

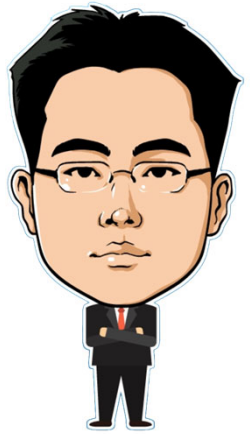
(C Standard Library)



Seo, Doo-Ok

Clickseo.com

clickseo@gmail.com



목 차

- 표준 입출력: <stdio.h>

백문이불여일타(百聞而不如一打)

- 표준 라이브러리: <stdlib.h>
- 문자와 문자열 처리
- 날짜 및 시간 관련 함수: <time.h>
- 수학 관련 함수: <math.h>

C 표준 라이브러리 (1/2)

- **표준 라이브러리 함수(Standard Library Functions)**
  - C/C++ 언어에서 프로그래머의 편의성을 도모하기 위해 프로그램 언어 개발자들에 의해 작성되어 포함된 함수
    - C 표준 라이브러리(C standard library)
      - ISO C 라이브러리
      - C POSIX 라이브러리 와 거의 동시에 개발
    - glibc(The GNU C Library)
  - #include
    - 표준 라이브러리 함수를 불러 들이기 위해 전 처리 구문 영역에 사용
    - 예: #include <stdio.h>
      - 대표적인 표준 라이브러리 함수인 **scanf** 와 **printf** 계열 함수를 사용하기 위한 헤더 파일

C 표준 라이브러리 (2/2)

C 표준 라이브러리(C Standard Library)

헤더 파일

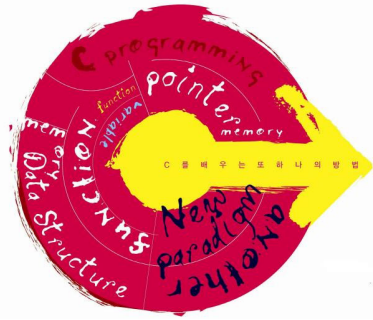
- 하나 이상의 함수 원형 선언과 자료형의 정의 그리고 다양한 매크로들을 포함
- **현재 총 29개의 헤더 파일을 제공**
- **1995년, 규범 별첨 1(NA1)에서 3개의 헤더 파일 추가**
  - iso646.h, wchar.h 그리고 wctype.h
- **1999년, C99 버전에서 새롭게 6개의 헤더 파일 추가**
  - complex.h, fenv.h, inttypes.h, stdbool.h, stdint.h 그리고 tgmath.h
- **2011년, C11 버전에서 5개의 헤더 파일 추가**
  - stdalign.h, stdatomic.h, stdnoreturn.h, threads.h 그리고 uchar.h

표준 입출력

- **표준 입출력:** <stdio.h>

백문이불여일타(百聞而不如一打)

- 표준 라이브러리: <stdlib.h>
- 문자와 문자열 처리
- 날짜 및 시간 관련 함수: <time.h>
- 수학 관련 함수: <math.h>



표준 입출력 (1/6)

- **매크로(Macro): 변수 및 상수, 매크로 함수**

```
#include <stdio.h>

extern  FILE  __iob[FOPEN_MAX]
#define  stdin        (&__iob[0])          // 표준 입력 스트림
#define  stdout       (&__iob[1])          // 표준 출력 스트림
#define  stderr       (&__iob[2])          // 프로그램의 오류 메시지
                      // 또는 다른 예외적인 내용을 출력하기 위한 출력 스트림

#define  EOF          (-1)
#define  NULL         ((void  *) 0)

#define  BUFSIZ       ...          // 256KB, 512KB 또는 4096KB

#define  FOPEN_MAX    ...          // 동시에 개방할  수 있는 스트림 개수(최소 8 이상)
#define  FILENAME_MAX ...          // 개방할 수 있는 파일 이름의 최대 길이

#define  SEEK_SET  0
#define  SEEK_CUR  1
#define  SEEK_END  2
```

표준 입출력 (2/6)

- 형식화된 입출력(Formatted I/O) 함수 (1/2)

```
#include <stdio.h>
int scanf(const char* format, ...);                          // until C99
int fscanf(FILE* stream, const char* format, ...);           // until C99
int sscanf(char* buffer, const char* format, ...);           // until C99

int scanf(const char* restrict format, ...);                 // since C99
int fscanf(FILE* stream, const char* restrict format, ...);  // since C99
int sscanf(char* buffer, const char* restrict format, ...);  // since C99

int scanf_s(const char* restrict format, ...);               // since C11
int fscanf_s(FILE* stream, const char* restrict format, ...);// since C11
int sscanf_s(char* buffer, const char* restrict format, ...);// since C11
```

반환(성공): 입력에 성공한 필드 개수를 반환
(실패): EOF 값을 반환

표준 입출력 (3/6)

- 형식화된 입출력(Formatted I/O) 함수 (2/2)

```
#include <stdio.h>
int printf(const char* format, ...);                                    // until C99
int fprintf(FILE *stream, const char *format, ...);                     // until C99
int sprintf(char *buffer , const char *format, ...);                    // until C99
int snprintf(char *buffer , int buf_siz, const char *format, ...);      // until C99

int printf(const char* restrict format, ...);                           // since C99
int fprintf(FILE *stream, const char* restrict format, ...);            // since C99
int sprintf(char *buffer , const char* restrict format, ...);           // since C99
int snprintf(char *buffer , int buf_siz, const char* restrict format, ...);  // since C99

int printf_s(const char *restrict format, ...);                         // since C11
int fprintf_s(FILE *stream, const char* restrict format, ...);          // since C11
int sprint_s(char *buffer , const char* restrict format, ...);          // since C11
int snprintf_s(char *buffer , int buf_siz, const char* restrict format, ...); // since C11
```

반환(성공): 출력한 바이트 수를 반환
(실패): EOF 값을 반환

표준 입출력 (4/6)

● 비형식화 된 입출력(Unformatted I/O) 함수 (1/3)

```
#include <stdio.h>
int getchar(void);    // int getc(stdin)
int putchar(int ch);  // int putc(ch,  stdout)
```

반환(성공): 표준 입출력에서 읽거나 가록한 문자를 반환
(실패): EOF 값을 반환

```
#include <stdio.h>
int getc(FILE* stream);
int fgetc(FILE* stream);

int putc(int ch, FILE* stream);
int fputc(int ch, FILE* stream);

int ungetc(int ch, FILE* stream);
```

반환(성공): 스트림으로부터 읽거나 기록한 문자를 반환
(실패): EOF 값을 반환

표준 입출력 (5/6)

● 비형식화 된 입출력 함수 (2/3)

```
#include <stdio.h>
char* gets (char *str);                          // removed in C11

char* gets_s(char* str,  rsize_t n);             // since C11
char* fgets(char* restrict str, int count, FILE* restrict stream);          // since C99
char* fgets(char* str, int count, FILE* stream);                            // until C99
```

반환(성공): str의 주소를 반환
(실패): NULL을 반환

```
#include <stdio.h>
int puts(const char* str);
int fputs(const char* restrict str, FILE* restrict stream);        // since C99
int fputs(const char* str, FILE* stream);                          // until C99
```

반환(성공): 음수가 아닌 정수를 반환
(실패): EOF를 반환

표준 입출력 (6/6)

- **파일에 대한 작업(Operations on files) 함수**

```
#include <stdio.h>

// 지정된 파일(filename)을 삭제 한다.
int remove(const char* filename);

// 기존 파일  이름(old_filename)을 새로운 파일 이름(new_filename)으로 변경한다.
int rename(const char* old_filename, const char* new_filename);
```

호출 성공: 0 을 반환
에러 발생: 0 이 아닌 값을 반환

표준 라이브러리

- 표준 입출력: <stdio.h>

백문이불여일타(百聞而不如一打)

- **표준 라이브러리: <stdlib.h>**
- 문자와 문자열 처리
- 날짜 및 시간 관련 함수: <time.h>
- 수학 관련 함수: <math.h>

표준 라이브러리 (1/8)

- **데이터 유형**(Data Types)

```
#include <stdlib.h>

typedef unsigned int        size_t;
typedef unsigned short      wchar_t;
```

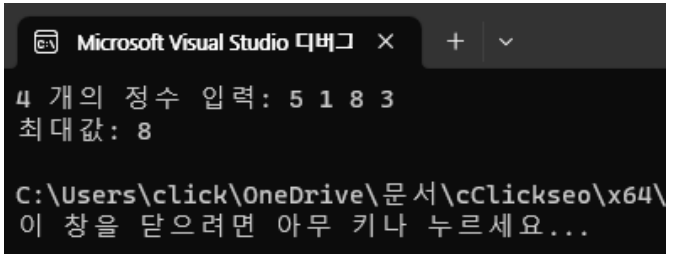
- **매크로**(Macro): 변수 및 상수 그리고 매크로 함수

```
#include <stdlib.h>

#define  NULL       ((void*) 0)

// exit 함수를 위해 정의된  값
#define  EXIT_SUCCESS       0
#define  EXIT_FAILURE       1

// rand 함수 사용 시 임의의 난수 발생 최대값
#define  RAND_MAX  0x7fff

#define  __max(a,  b)  (((a) > (b)) ? (a) : (b))      // 매크로 함수
#define  __min(a,  b)  (((a) < (b)) ? (a) : (b))      // 매크로 함수
```

표준 라이브러리 (2/8)

예제 6-1: 최댓값 구하기

```c
#include <stdio.h>
#include <stdlib.h>   // __max, __min
int main(void)
{
        int     a, b, c, d, max;

        printf("4 개의 정수 입력: ");
        scanf_s("%d %d %d %d", &a, &b, &c, &d);
        // scanf("%d %d %d %d", &a, &b, &c, &d);

        /*
                if (a > b)      max = a;
                else            max = b;
                if (c > max)    max = c;
                if (d > max)    max = d;
        */

        max = __max(a, b);
        max = __max(c, max);
        max = __max(d, max);

        printf("최대값: %d \n", max);
        return 0;
}
```



표준 라이브러리 (3/8)

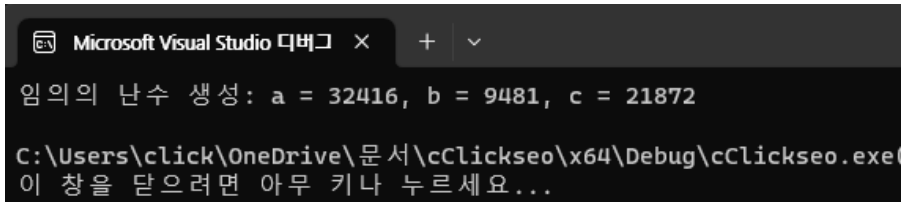
- 임의의 난수 생성 함수

```
#include <stdlib.h>

// 0 ~ RAND_MAX 사이에서 임의의 난수를 생성하여 반환
int  rand(void);

// seed 있는 랜덤발생 함수, 초기의 seed 는 1이다.
void  srand(unsigned int seed);
```

표준 라이브러리 (4/8)

예제 6-2: 임의의 난수 생성

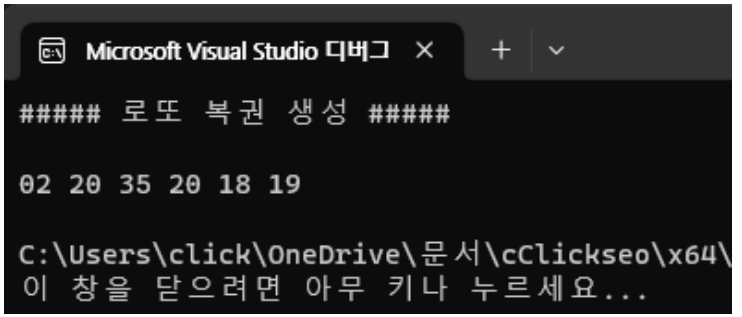
```
#include <stdio.h>
#include <stdlib.h>      // srand, rand
#include <time.h>        // time

int main(void)
{
        int a, b, c;

        srand((unsigned int) time(NULL));

        a = rand();
        b = rand();
        c = rand();

        printf("임의의 난수 생성: a = %d, b = %d, c = %d \n", a, b, c );
        return 0;
}
```



표준 라이브러리 (5/8)

예제 6-3: 임의의 난수 생성 -- 범위 설정

```c
#include <stdio.h>
#include <stdlib.h>      // srand, rand
#include <time.h>        // time

int main(void)
{
        int     temp;

        printf("##### 로또 복권 생성 ##### \n\n");
        srand((unsigned int)time(NULL));
        for (int i = 1; i <= 6; i++ ) {
                temp = rand() % 45 + 1;          // 최댓값과 최솟값
                printf("%02d ", temp );
        }
        printf("\n");
        return 0;
}
```



표준 라이브러리 (6/8)

● 프로세스 제어 관련 함수

```
#include <stdlib.h>

void  exit(int exit_code);        //프로그램을 정상적인 상태로 종료
void  abort();                    // 프로그램을 그 상태에서 정지(비정상적인 종료)

// 프로세스가 exit 함수를 호출하여 종료할 때 수행되는 함수들을 등록한다.
int  atexit(void(*func)(void));

// exit 함수와 같지만 clean-up-action을 수행하지 않고 프로그램 종료
void  _exit(int exit_code);
void  _Exit(int exit_code);       // C99
```

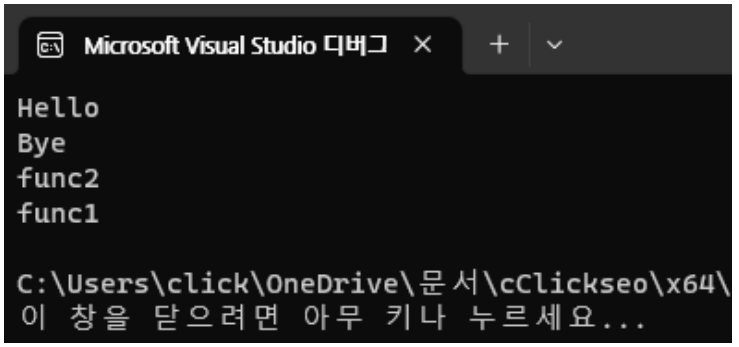
```
#include <stdlib.h>
int  system(const char* command);
```

호출 성공: 0 값을 반환
범위 오류: -1 값을 반환

표준 라이브러리 (7/8)

예제 6-4: 프로그램 흐름 제어 관련 함수

```
#include <stdio.h>
#include <stdlib.h>        // exit, atexit

void  func1(void);
void  func2(void);

int main(void)
{
        printf("Hello \n");
        atexit(func1);
        atexit(func2);
        printf("Bye \n");

        exit(0);           // _exit(0); 로 수정 시 func1과 func2는 실행되지 않고 종료한다.
}

void func1(void) {
        printf("func1 \n");
}

void func2(void) {
        printf("func2 \n");
}
```



표준 라이브러리 (8/9)

- **문자열을 숫자로 변환: atoi, atof**
  - 문자열을 숫자(정수형 또는 실수형)로 변환

```
#include <stdlib.h>

// 문자열을 정수형(int, long, long long)으로 변환하는 함수
int       atoi(const  char* str);
long      atol(const  char* str);
long long atoll(const  char* str);          // since C99

// 문자열을 실수형(double)으로 변환하는 함수
double    atof(const  char* str);
```

표준 라이브러리 (9/9)

- **숫자를 문자열로 변환하: itoa, ltoa**
  - 정수형 숫자를 2진수, 8진수, 10진수 또는 16진수의 문자열로 변환

```
#include <stdlib.h>

// 정수형(int) 숫자를 문자열 str 로 변환
char*      itoa(int  value,  char* buffer,  int  radix);                  // POSIX
errno_t    _itoa_s(int value, char* buffer, size_t size, int radix);      // ISO

// 정수형(long) 숫자를 문자열 str 로 변환
char*      ltoa(long  value,  char* buffer,  int  radix);                 // POSIX
errno_t    _ltoa_s(long value, char* buffer, size_t size, int radix);     // ISO
```

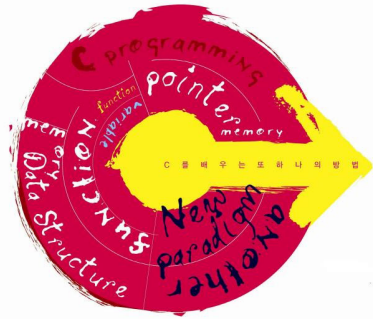
```
// itoa, ltoa 함수 사용 시 Visual Studio 2019부터 오류(error) 메시지
Error C4996:
The POSIX name for this item is deprecated.
instead, use the ISO C and C++ conformant name: _itoa.
See online help for details.
```

문자와 문자열 처리

- 표준 입출력: <stdio.h>

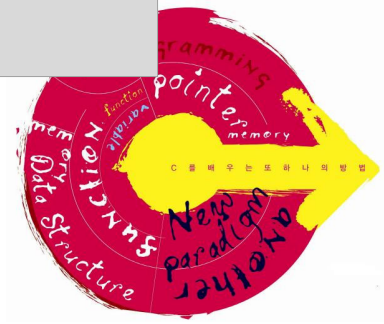
백문이불여일타(百聞而不如一打)

- 표준 라이브러리: <stdlib.h>
- **문자와 문자열 처리**
- 날짜 및 시간 관련 함수: <time.h>
- 수학 관련 함수: <math.h>



문자와 문자열 처리

문자 처리: ctype.h



문자 처리 (1/2)

- **매크로(Macro): 변수 및 상수, 매크로 함수**

```
#include <ctype.h>

#define  _UPPER          0x1      // upper case letter
#define  _LOWER          0x2      // lower case letter
#define  _DIGIT          0x4      // digit[0-9]
#define  _SPACE          0x8      // tab, carriage return, newline,
                                  // vertical tab or form feed
#define  _PUNCT          0x10     // punctuation character
#define  _CONTROL        0x20     // control character
#define  _BLANK          0x40     // space char
#define  _HEX            0x80     // hexadecimal digit

#define  _LEADBYTE       0x8000   // multibyte leadbyte
#define  _ALPHA     (0x0100|_UPPER|_LOWER)   // alphabetic character
```

문자 처리 (2/2)

ASCII values (hex)		characters	iscntrl iswcntrl	isprint iswprint	isspace iswspace	isblank iswblank	isgraph iswgraph	ispunct iswpunct	isalnum iswalnum	isalpha iswalpha	isupper iswupper	islower iswlower	isdigit iswdigit	isxdigit iswxdigit
0 - 8	0x00-0x08	control codes (NUL, etc.)	≠0	0	0	0	0	0	0	0	0	0	0	0
9	0x09	tab (\t)	≠0	0	≠0	≠0	0	0	0	0	0	0	0	0
10 - 13	0x0A-0x0D	whitespaces (\n,\v,\f,\r)	≠0	0	≠0	0	0	0	0	0	0	0	0	0
14 - 31	0x0E-0x1F	control codes	≠0	0	0	0	0	0	0	0	0	0	0	0
32	0x20	space	0	≠0	≠0	≠0	0	0	0	0	0	0	0	0
33 - 47	0x21-0x2F	!"#$%&'()*+,-./	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
48 - 57	0x30-0x39	0123456789	0	≠0	0	0	≠0	0	≠0	0	0	0	≠0	≠0
58 - 64	0x3a-0x40	:;<=>?@	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
65 - 70	0x41-0x46	ABCDEF	0	≠0	0	0	≠0	0	≠0	≠0	≠0	0	0	≠0
71 - 90	0x47-0x5A	GHIJKLMNOPQRSTUVWXYZ	0	≠0	0	0	≠0	0	≠0	≠0	≠0	0	0	0
91 - 96	0x5B-0x60	[\]^_`	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
97 -102	0x61-0x66	abcdef	0	≠0	0	0	≠0	0	≠0	≠0	0	≠0	0	≠0
103-122	0x67-0x7A	ghijklmnopqrstuvwxyz	0	≠0	0	0	≠0	0	≠0	≠0	0	≠0	0	0
123-126	0x7B-0x7E	{|}~	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
127	0x7F	backspace character (DEL)	≠0	0	0	0	0	0	0	0	0	0	0	0

[출처 : http://en.wikipedia.org/wiki/C_character_classification]

문자 변환 (1/2)

- **문자 조작:** 영문 대소문자
  - **영문 대문자를 소문자로 또는 영문 소문자를 대문자로 변환**
    - 변환 대상 문자는 **unsigned char** 로 표현 될 수 없다.
    - **EOF** 와 같지 않은 경우의 동작은 정의되지 않는다.

```
#include <ctype.h>
```

대소문자 변환 성공: **변환 된 문자** ch 값(ASCII code) **반환**
대소문자 변환 실패: 기존 문자 ch 값(ASCII code) 반환

```
int  tolower(int ch);          // 주어진 문자 ch 를 소문자로 변환
int  toupper(int ch);          // 주어진 문자 ch 를 대문자로 변환
```

Microsoft Visual Studio 디버그 콘솔

```
tolower('A'): a
tolower('a'): a
tolower('0'): 0
tolower('*'): *

C:₩Users₩click₩OneDrive₩문서₩cClickseo₩x64₩
이 창을 닫으려면 아무 키나 누르세요...
```

Microsoft Visual Studio 디버그 콘솔

```
toupper('A'): A
toupper('a'): A
toupper('0'): 0
toupper('*'): *

C:₩Users₩click₩OneDrive₩문서₩cClickseo₩x64₩
이 창을 닫으려면 아무 키나 누르세요...
```

문자 변환 (2/2)

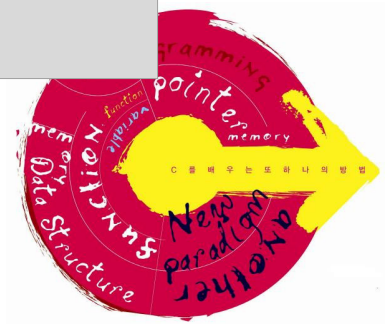
예제 6-5: 문자 분류 및 변환 -- islower, isupper, tolower, toupper

```
#include <stdio.h>
#include <ctype.h>      // islower, isupper, tolower, toupper

int main(void)
{
        char    str[] = "Hi~Clickseo";
        char*   pStr = str;

        fputs("원본 문자열: ", stdout);
        puts(str);

        printf("변환 문자열: ");
        for (; *pStr != '₩0'; pStr++) {
                if (islower(*pStr))         putchar(toupper(*pStr));
                else if (isupper(*pStr))    putchar(tolower(*pStr));
                else                        putchar(*pStr);
        }
        *pStr = '₩0';
        putchar('\n');

        return 0;
}
```

```
Microsoft Visual Studio 디버그
원본 문자열: Hi~Clickseo
변환 문자열: hI~cLICKSEO

C:\Users\click\OneDrive\문서\cClickseo\x64\
이 창을 닫으려면 아무 키나 누르세요...
```

문자와 문자열 처리

문자열 처리: string.h



문자열 처리 (1/2)

- **문자열 처리(String Handling): string.h**
  - C 표준 라이브러리: **<string.h>**
  - 널 문자를 제외한 문자 개수를 검증하는 과정

```
size_t  my_strlen(const char* pStr) {
        if  (pStr == NULL)  {        // 잘못된 문자열
                return 0;
        }

        int  count = 0;
        while (*pStr++)
                count++;
        return count;
}
```

```
#include <string.h>

size_t      strlen (const char* pStr);
```

문자열 처리 (2/2)

- **문자열 처리:** 데이터 유형과 매크로 상수
  - 데이터 유형

```
#include <string.h>

typedef unsigned int        size_t;
typedef unsigned short      wchar_t;
```

  - 매크로 상수

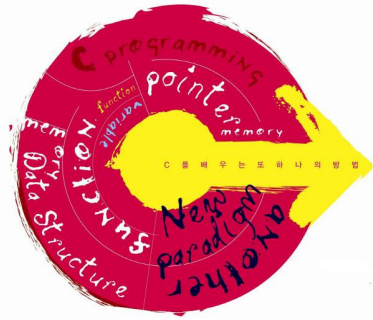
```
#include <string.h>

#define  NULL       ((void*) 0)
```

날짜 및 시간 관련 함수

- 표준 입출력: <stdio.h>

백문이불여일타(百聞而不如一打)

- 표준 라이브러리: <stdlib.h>
- 문자와 문자열 처리
- **날짜 및 시간 관련 함수:** <time.h>
- 수학 관련 함수: <math.h>



날짜 및 시간 관련 함수 (1/4)

- **데이터 유형(Data Types)**

```
#include <time.h>
struct tm {
        int     tm_sec;     // 초 [0, 60] (since C99)  // [0, 61] (until C99)
        int     tm_min;     // 분 [0, 59]
        int     tm_hour;    // 시 [0, 23]
        int     tm_mday;    // 일 [1 - 31]
        int     tm_mon;     // 월 [0, 11]          // 0 = January
        int     tm_year;    // 년       // 1900년 이후부터 경과한 년도
        int     tm_wday;    // 요일 [0, 6]         // 0 = Sunday
        int     tm_yday;    // 날짜 [0, 365]       // 1월 1일부터 경과된 날짜
        int     tm_isdst;   // 일광 절약 시간제(DST, Daylight Saving Time)
};                          // DST가 적용되면 양수이고, 그렇지 않으면 0이다.
                            // 사용 가능한 정보가 없으면 음수이다.

typedef  long       time_t;   // 시간을 표현하는 산술 유형
typedef  long       clock_t;  // 프로세스 실행 시간 유형
```

날짜 및 시간 관련 함수 (2/4)

- **데이터 유형(Data Types)**

```
#include <time.h>

typedef    long        time_t;    // 시간을 표현하는 산술 유형
typedef    long        clock_t;   // 프로세스 실행 시간 유형
```

- **매크로(Macro): 변수 및 상수**

```
#include <time.h>

#define CLOCKS_PER_SEC      1000
#define CLK_TCK             CLOCKS_PER_SEC
```

날짜 및 시간 관련 함수 (3/4)

- **시간 조작 함수**

```
#include <time.h>

// 프로그램 시작 후의 경과한 clock tick(1초에 18.2만큼 증가)를 반환
clock_t  clock();
```

호출 성공: process tick count 값을 반환
호출 실패: (clock_t) -1 값을 반환

```
#include <time.h>

// 1970년 1월 1일 자정부터 경과된 현재 시간을 초 단위로 계산하여 반환
time_t  time(time_t* time);
```

호출 성공: 현재 시간 값을 반환
호출 실패: (time_t) -1 값을 반환

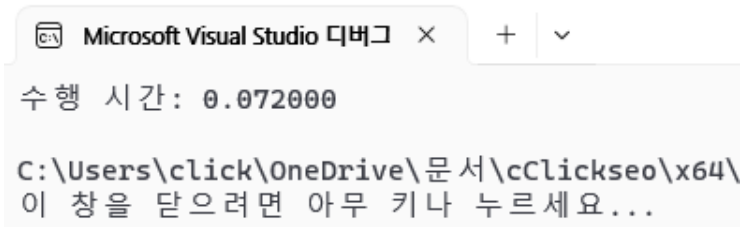
```
#include <time.h>

// time2 – time1 을 초로 계산하여 결과 값을 반환
double  difftime(time_t  time2, time_t  time1);
```

호출 성공: 두 시간의 차를 계산한 실수 값을 반환

날짜 및 시간 관련 함수 (4/4)

예제 6-5: 프로그램 수행 시간

```
#include <stdio.h>
#include <time.h>       // clock_t, CLK_TCK, clock 함수

int main(void)
{
        clock_t         start, end;
        double          time;

        start = clock();
        for(int i=0; i<30000000; i++);
        end = clock();

        time = (double)(end - start) / CLK_TCK;
        printf("수행 시간: %lf \n", time );

        return 0;
}
```

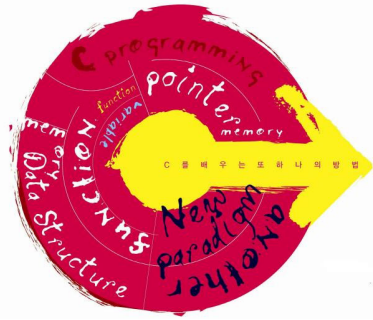


수학 관련 함수

- 표준 입출력: <stdio.h>

백문이불여일타(百聞而不如一打)

- 표준 라이브러리: <stdlib.h>
- 문자와 문자열 처리
- 날짜 및 시간 관련 함수: <time.h>
- **수학 관련 함수:** <math.h>



수학 관련 함수 (1/13)

- **수학 계산 함수: abs**

```
#include <math.h>

// 정수형의 절대값 을 계산하여 반환
int                 abs(int num);
long                labs(long num);
long long           llabs(long long num);          // C99
```

```
#include <math.h>

// 부동 소수점의 절대값을 계산하여 반환
float               fabsf(float arg);              // C99
double              fabs(double arg);
long double         fabsl(long double arg);        // C99
```

수학 관련 함수 (2/13)

- **수학 계산 함수:** fmod, fmax, fmin

```
#include <math.h>

// 실수 x 를 실수 y 로 나눈 나머지를 계산하여 반환
float                fmodf(float x,  float y);                    // C99
double               fmod(double x,  double y);
long double          fmodl( long  double x,  long  double y);     // C99

// 실수 x 와 실수 y 에서 큰 값을 반환
float                fmaxf(float x,  float y);                    // C99
double               fmax(double x,  double y);                   // C99
long double          fmaxl(long double x,  long double y);        // C99

// 실수 x 와 실수 y 에서 작은 값을 반환
float                fminf(float x,  float y);                          // C99
double               fmin(double x,  double y);                         // C99
long double          fminl(long double x,  long double y);              // C99
```

수학 관련 함수 (3/13)

- **지수와 로그 함수: log10**

```
#include <math.h>


// arg 의 자연 로그(기본 base는 10) 값을 반환
float                log10f(float arg);              // C99
double               log10(double arg);
long double          log10l(long double  arg);       // C99
```

수학 관련 함수 (4/13)

- **지수와 로그 함수:** exp, log

```
#include <math.h>

// 기본 base 가 자연 로그(e)인 arg 제곱 값을 반환
float          expf(float arg);          // C99
double         exp(double arg);
long double    expl(long double arg);    // C99
```

```
#include <math.h>

// arg 의 자연 로그(기본 base 는 e) 값을 반환
float          logf(float arg);          // C99
double         log(double arg);
long double    logl(long double arg);    // C99
```

수학 관련 함수 (5/13)

- **지수와 로그 함수:** pow, sqrt

```
#include <math.h>

// base 의 exp 승 값을 계산하여 반환
float          powf(float base, float exp);          // C99
double         pow(double base, double exp);
long double    powl(long double base, long double exp);   // C99
```

```
#include <math.h>

// arg 의 0 이 아닌 제곱근을 계산하여 반환
float          sqrtf(float arg);          // C99
double         sqrt(double arg);
long double    sqrtl(long double arg);    // C99
```

수학 관련 함수 (6/13)

- 가장 가까운 정수 또는 부동 소수점 연산 함수 (1/2)

```
#include <math.h>

// arg 를 초과하는 정수 중에서 가장 가까운 정수를 반환
float                ceilf(float  arg);              // C99
double               ceil(double  arg);
long double          ceill(long  double  arg);       // C99

// arg 보다 작은 정수 중에서 가장 가까운 정수를 반환
float                floorf(float  arg);             // C99
double               floor(double  arg);
long double          floorl(long  double  arg);      // C99

// arg 의 소수점 이하를 버리고 정수를 반환
float                truncf(float  arg);             // C99
double               trunc(double  arg);             // C99
long double          truncl(long  double  arg);      // C99
```

수학 관련 함수 (7/13)

- 가장 가까운 정수 또는 부동 소수점 연산 함수 (2/2)

```
#include <math.h>

// arg  에서 가까운 정수(즉, 반올림 값)를 반환
float                roundf(float  arg);              // C99
double               round(double  arg);              // C99
long double          roundl(long  double  arg);       // C99

long                 lroundf(float  arg);             // C99
long                 lround(double  arg);             // C99
long                 lroundl(long  double  arg);      // C99

long long            llroundf(float  arg);            // C99
long long            llround(double  arg);            // C99
long long            llroundl(long  double  arg);     // C99
```

수학 관련 함수 (8/13)

- **부동 소수점 조작 함수:** modf, ldexp

```
#include <math.h>

// 실수 x 를 정수 부분과 실수 부분으로 분리하여 한다.
// 정수부분은 iptr 에 저장되고, 실수부분은 반환
float             modff(float x, float* iptr);                  // C99
double            modf(double x, double* iptr);
long double       modfl(long double x,  long double* iptr);     // C99
```

```
#include <math.h>

// arg * 2^exp 를 계산하여 반환
// arg * pow(2, exp) 의 결과와 동일하다.
float             ldexpf(float arg, int exp);       // C99
double            ldexp(double arg, int exp);
long double       ldexpl(long double arg, int xp); // C99
```

수학 관련 함수 (9/13)

- **삼각 함수:** sin, cos, tan

```
#include <math.h>

// arg 의 sine 값을 계산하여 반환
float                sinf(float arg);              // C99
double               sin(double arg);
long double          sinl(long double arg);        // C99

// arg 의 cosine 값을 계산하여 반환
float                cosf(float arg);              // C99
double               cos(double arg);
long double          cosl(long double arg);        // C99

// arg 의 tangent 값을 계산하여 반환
float                tanf(float arg);              // C99
double               tan(double arg);
long double          tanl(long double arg);        // C99
```

수학 관련 함수 (10/13)

- **삼각 함수:** asin, acos, atan

```
#include <math.h>

// arg 의 arc  sine 값을 계산하여 반환
float                asinf(float arg);              // C99
double               asin(double arg);
long double          asinl(long double arg);        // C99

// arg 의 arc  cosine 값을 계산하여 반환
float                acosf(float arg);              // C99
double               acos(double arg);
long double          acosl(long double arg);        // C99

// arg 의 arc  tangent 값을 계산하여 반환
float                atanf(float arg);              // C99
double               atan(double arg);
long double          atanl(long double arg);        // C99
```

수학 관련 함수 (11/13)

- **삼각 함수:** atan2

```
#include <math.h>

// y/x 의 arc tangent 값을 계산하여 반환
float          atan2f(float y,  float x);          // C99
double         atan2(double y,  double x);         // C99
long double    atan2l(long double y, long double x); // C99
```

수학 관련 함수 (12/13)

- **쌍곡선 함수:** sinh, cosh, tanh

```
#include <math.h>

// arg 의 쌍곡선(hyperbolic)  sine 값을 계산하여 반환
float                sinhf(float arg);                // C99
double               sinh(double arg);
long double          sinhl(long double arg);          // C99

// arg 의 쌍곡선(hyperbolic)  cosine 값을 계산하여 반환
float                coshf(float arg);                // C99
double               cosh(double arg);
long double          coshl(long double arg);          // C99

// arg 의 쌍곡선(hyperbolic)  tangent 값을 계산하여 반환
float                tanhf(float arg);                // C99
double               tanh(double arg);
long double          tanhl(long double arg);          // C99
```

수학 관련 함수 (13/13)

- **쌍곡선 함수:** asinh, acosh, atanh

```
#include <math.h>

// arg 의 쌍곡선(hyperbolic) arc sine 값을 계산하여 반환
float                asinhf(float arg);              // C99
double               asinh(double arg);
long double          asinhl(long double arg);        // C99

// arg 의 쌍곡선(hyperbolic) arc cosine 값을 계산하여 반환
float                acoshf(float arg);              // C99
double               acosh(double arg);
long double          acoshl(long double arg);        // C99

// arg 의 쌍곡선(hyperbolic) arc tangent 값을 계산하여 반환
float                atanhf(float arg);              // C99
double               atanh(double arg);
long  double         atanhl(long double arg);        // C99
```

참고문헌

[1] 서현우, "혼자 공부하는 C 언어 : 1:1 과외 하듯 배우는 프로그래밍 자습서", 한빛미디어, 2023.

[2] Paul Deitel, Harvey Deitel, "C How to Program", Global Edition, 8/E, Pearson, 2016.

[3] Kamran Amini, 박지윤 번역, "전문가를 위한 C : 동시성, OOP부터 최신 C, 고급 기능까지!", 한빛미디어, 2022.

[4] 서두옥, "(열혈강의) 또 하나의 C : 프로그래밍은 셀프입니다", 프리렉, 2012.

[5] Behrouz A. Forouzan, Richard F. Gilberg, 김진 외 7인 공역, "구조적 프로그래밍 기법을 위한 C", 도서출판 인터비젼, 2004.

[6] Brian W. Kernighan, Dennis M. Ritchie, 김석환 외 2인 공역, "The C Programming Language", 2/E, 대영사, 2004.

[7] "C reference", cppreference.com, 2023 of viewing the site, https://en.cppreference.com/w/c.