둘째날

C 맛보기 계속 냠~냠~

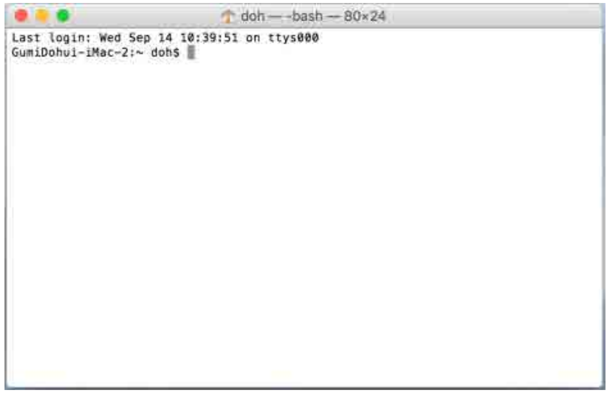
CSE2018 시스템프로그래밍기초
2016년 2학기

한양대학교 ERICA
컴퓨터공학과 => 소프트웨어학부
도경구

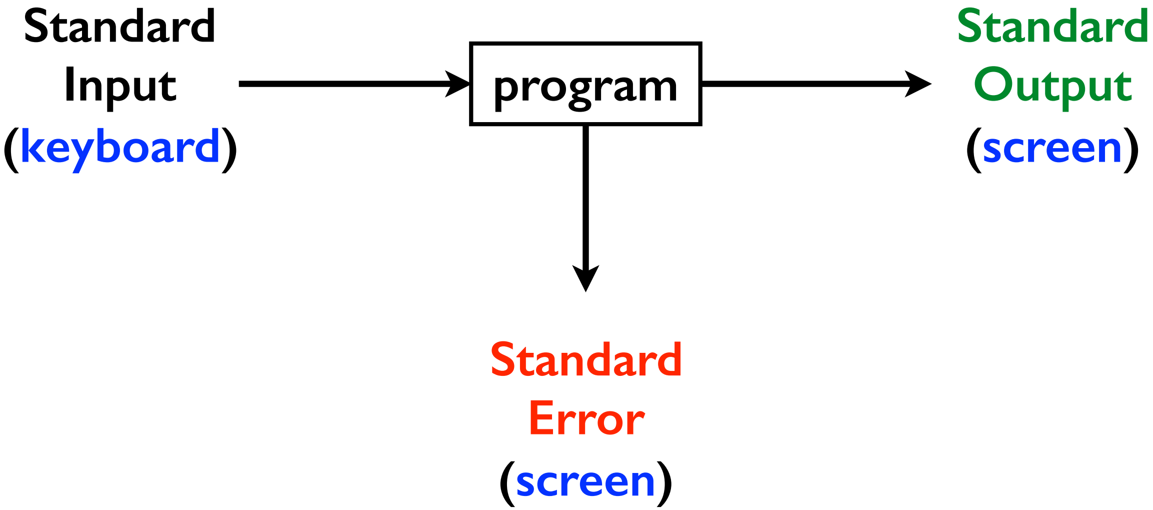
표준 입출력
Standard Input and Output

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

Standard Library



```
Last login: Wed Sep 14 10:39:51 on ttys000
GumiDohui-iMac-2:~ doh$
```

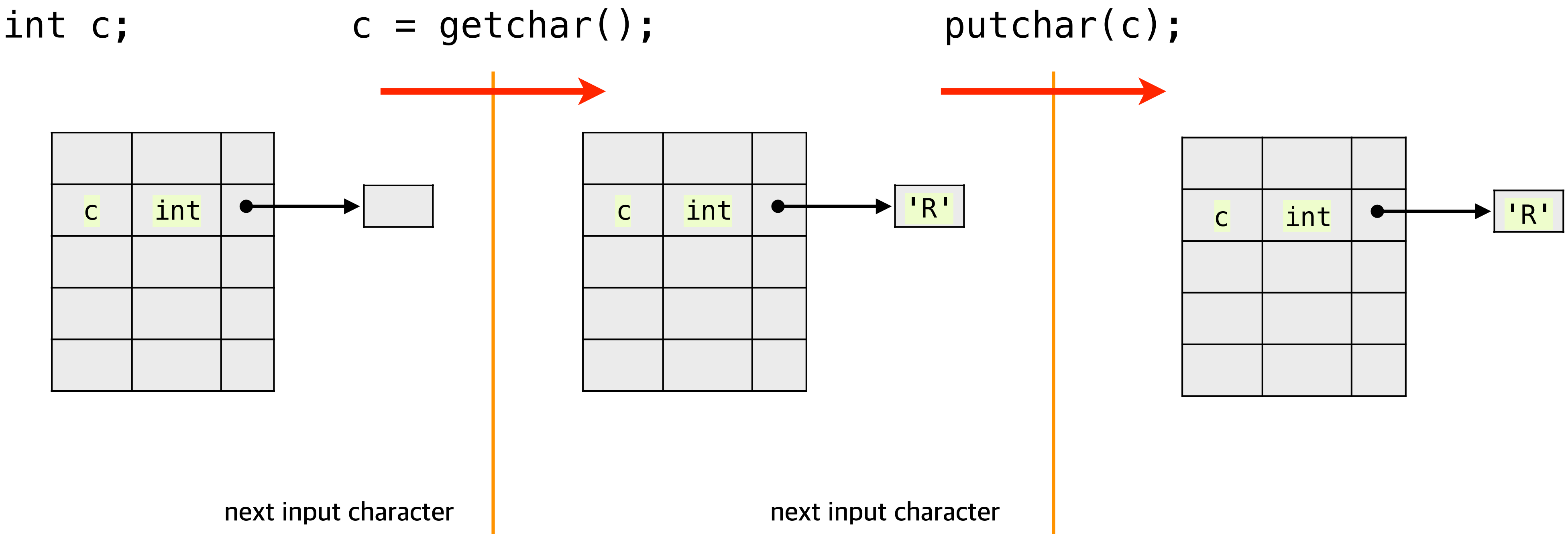

Standard Input (keyboard)

program

Standard Output (screen)

Standard Error (screen)

문자 입출력
Character Input and Output



```
int c;            c = getchar();            putchar(c);
```

next input character

Standard Input
text stream

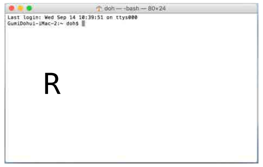
```
ERICA
Software
Hanyang
```

next input character

Standard Input
text stream

```
ERICA
Software
Hanyang
```

Standard Output



```
Last login: Wed Sep 14 10:39:51 on ttys000
GumiDohui-iMac-2:~ doh$

R
```

한 번에 문자 하나씩 입력에서 출력으로 카피하기
File Copying

알고리즘

```
read a character
while (character is not end-of-file indicator)
    output the character just read
    read a character
```

control-D

filecopy.c

```
#include <stdio.h>

/* copy input to output; 1st version */
int main() {
    int c;

    c = getchar();
    while (c != EOF) {
        putchar(c);
        c = getchar();
    }
}
```

한 번에 문자 하나씩 입력에서 출력으로 카피하기
File Copying

알고리즘

```
read a character
while (character is not end-of-file indicator)
    output the character just read
    read a character
```

filecopy.c

```
#include <stdio.h>

/* copy input to output; 1st version */
int main() {
    int c;

    c = getchar();
    while (c != EOF) {
        putchar(c);
        c = getchar();
    }
}
```

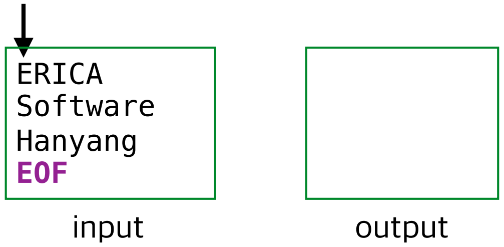
filecopy2.c

```
#include <stdio.h>

/* copy input to output; 2nd version */
int main() {
    int c;

    while ((c = getchar()) != EOF)
        putchar(c);
}
```

문자 개수 세기
Character Counting

charcount.c

```
#include <stdio.h>

/* count characters in input; 1st version */
int main() {
    long nc;

    nc = 0;
    while (getchar() != EOF)
        ++nc;
    printf("%ld\n", nc);
}
```



```
ERICA
Software
Hanyang
EOF
```

문자 개수 세기
Character Counting

charcount.c

```c
#include <stdio.h>

/* count characters in input; 1st version */
int main() {
    long nc;

    nc = 0;
    while (getchar() != EOF)
        ++nc;
    printf("%ld\n", nc);
}
```

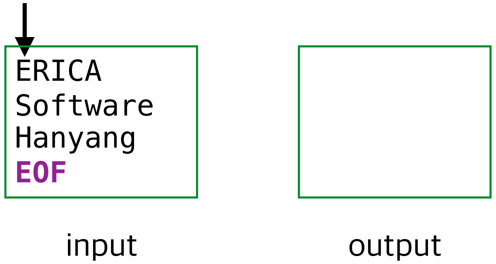
charcount2.c

```c
#include <stdio.h>

/* count characters in input; 2nd version */
int main() {
    double nc;

    for (nc = 0; getchar() != EOF; ++nc)
        ;
    printf("%.0f\n", nc);
}
```

줄수 세기
Line Counting

linecount.c

```
#include <stdio.h>

/* count lines in input */
int main() {
    int c, nl;

    nl = 0;
    while ((c = getchar()) != EOF)
        if (c == '\n')
            ++nl;
    printf("%d\n", nl);
}
```



```
ERICA
Software
Hanyang
EOF
```

input

output

단어 개수 세기
Word Counting

wordcount.c

```
#include <stdio.h>

#define IN  1    /* inside a word */
#define OUT 0    /* outside a word */

/* count lines, words, and characters in input */
int main() {
    int c, nl, nw, nc, state;

    state = OUT;
    nl = nw = nc = 0;
    while ((c = getchar()) != EOF) {
        ++nc;
        if (c == '\n')
            ++nl;
        if (c == ' ' || c == '\n' || c == '\t')
            state = OUT;
        else if (state == OUT) {
            state = IN;
            ++nw;
        }
    }
    printf("%d %d %d\n", nl, nw, nc);
}
```

input

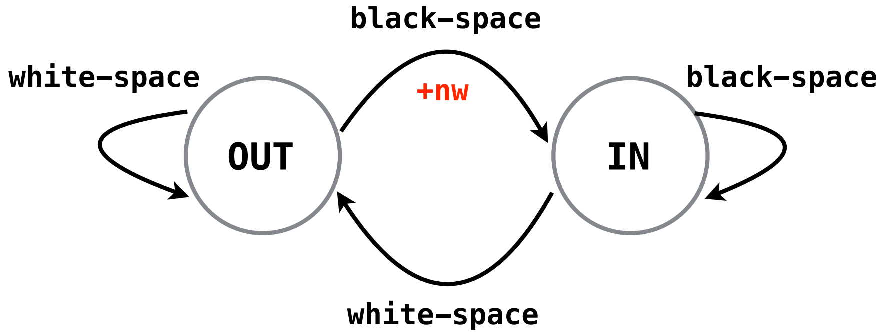
```
Boys,
be ambitious!
EOF
```

output

```
2 3 20
```

white-space = {' ' , '\n', '\t'}

black-space = all other chars



숫자(0-9)와 투명문자(white space), 기타문자 개수 세기

countdigits.c

Arrays
배열

```
#include <stdio.h>

/* count digits, white space, others */
int main() {
    int c, i, nwhite, nother;
    int ndigit[10];

    nwhite = nother = 0;
    for (i = 0; i < 10; ++i)
        ndigit[i] = 0;

    while ((c = getchar()) != EOF)
        if (c >= '0' && c <= '9')
            ++ndigit[c-'0'];
        else if (c == ' ' || c == '\n' || c == '\t')
            ++nwhite;
        else
            ++nother;

    printf("digits =");
    for (i = 0; i < 10; ++i)
        printf(" %d", ndigit[i]);
    printf("\nwhite space = %d\nother = %d\n",
           nwhite, nother);
}
```

숫자(0-9)와 투명문자(white space), 기타문자 개수 세기

countdigits.c

```c
#include <stdio.h>

/* count digits, white space, others */
int main() {
    int c, i, nwhite, nother;
    int ndigit[10];

    nwhite = nother = 0;
    for (i = 0; i < 10; ++i)
        ndigit[i] = 0;

    while ((c = getchar()) != EOF)
        if (c >= '0' && c <= '9')
            ++ndigit[c-'0'];
        else if (c == ' ' || c == '\n' || c == '\t')
            ++nwhite;
        else
            ++nother;

    printf("digits =");
    for (i = 0; i < 10; ++i)
        printf(" %d", ndigit[i]);
    printf("\nwhite space = %d\nother = %d\n",
           nwhite, nother);
}
```

char
타입

숫자(0-9)와 투명문자(white space), 기타문자 개수 세기

countdigits.c

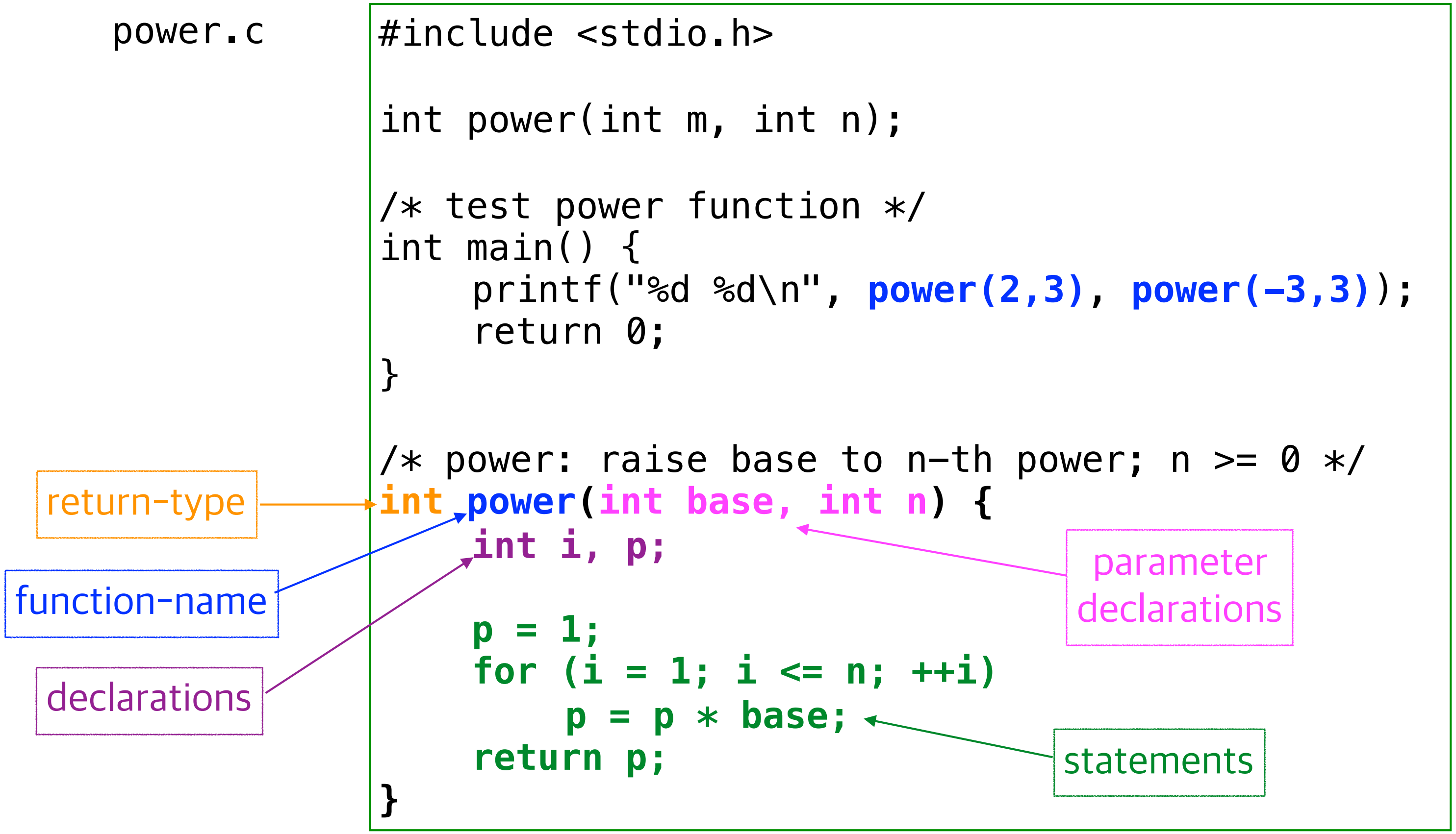
```
#include <stdio.h>

/* count digits, white space, others */
int main() {
    int c, i, nwhite, nother;
    int ndigit[10];

    nwhite = nother = 0;
    for (i = 0; i < 10; ++i)
        ndigit[i] = 0;

    while ((c = getchar()) != EOF)
        if (c >= '0' && c <= '9')
            ++ndigit[c-'0'];
        else if (c == ' ' || c == '\n' || c == '\t')
            ++nwhite;
        else
            ++nother;

    printf("digits =");
    for (i = 0; i < 10; ++i)
        printf(" %d", ndigit[i]);
    printf("\nwhite space = %d\nother = %d\n",
           nwhite, nother);
}
```

condition

statement

m(정수)의 n(양수) 승 구하기

power.c



```c
#include <stdio.h>

int power(int m, int n);

/* test power function */
int main() {
    printf("%d %d\n", power(2,3), power(-3,3));
    return 0;
}

/* power: raise base to n-th power; n >= 0 */
int power(int base, int n) {
    int i, p;

    p = 1;
    for (i = 1; i <= n; ++i)
        p = p * base;
    return p;
}
```

return-type

function-name

declarations

parameter declarations

statements

자가제작
함수
function definition

m(정수)의 n(양수) 승 구하기

power.c

```
#include <stdio.h>

int power(int m, int n);

/* test power function */
int main() {
    printf("%d %d\n", power(2,2+1), power(-3,3));
    return 0;
}

/* power: raise base to n-th power; n >= 0 */
int power(int base, int n) {
    int i, p;

    p = 1;
    for (i = 1; i <= n; ++i)
        p = p * base;
    return p;
}
```

arguments = actual parameters

parameters = formal parameters

함수호출
function
call

call-by-value

m(정수)의 n(양수) 승 구하기

power.c

```c
#include <stdio.h>

int power(int m, int n);

/* test power function */
int main() {
    printf("%d %d\n”, power(2,3), power(-3,3));
    return 0;
}

/* power: raise base to n-th power; n >= 0 */
int power(int base, int n) {
    int i, p;

    p = 1;
    for (i = 1; i <= n; ++i)
        p = p * base;
    return p;
}
```

return statement

m(정수)의 n(양수) 승 구하기

int power(int, int);

power.c

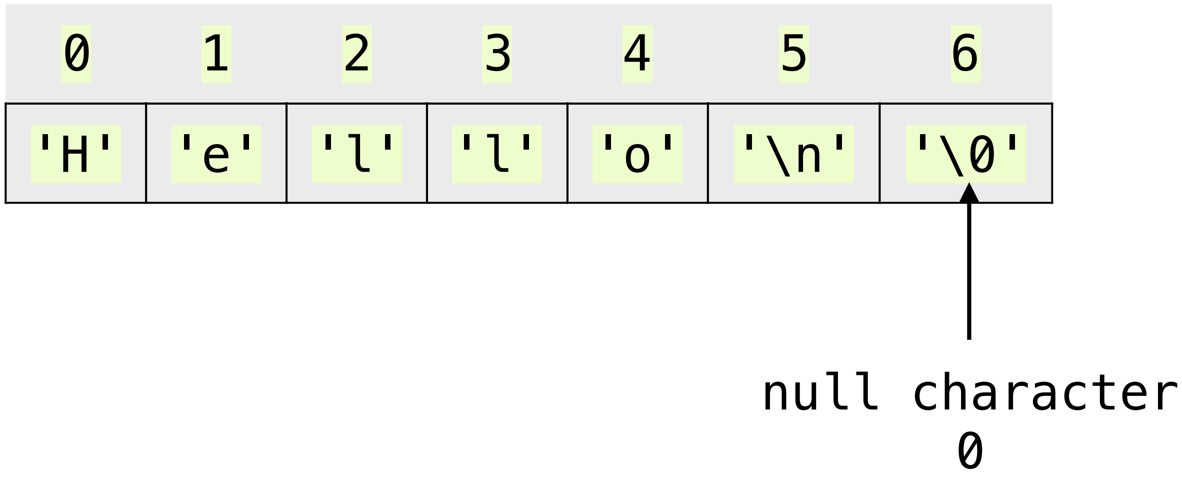
function prototype

```
#include <stdio.h>

int power(int m, int n);

/* test power function */
int main() {
    printf("%d %d\n", power(2,3), power(-3,3));
    return 0;
}

/* power: raise base to n-th power; n >= 0 */
int power(int base, int n) {
    int i, p;

    p = 1;
    for (i = 1; i <= n; ++i)
        p = p * base;
    return p;
}
```

normal termination

문자열
Character String

```
"Hello\n"
```



0	1	2	3	4	5	6
'H'	'e'	'l'	'l'	'o'	'\n'	'\0'

null character
0

여러 줄을 읽고 제일 긴 줄을 프린트하기

실행사례

input

```
ERICA
Software
Hanyang
```

output

```
Software
```

알고리즘

```
while (아직 읽을 줄이 있다.)
    if (그 줄이 지금까지 가장 긴 줄보다 길다.)
        그 줄을 기억한다.
        그 줄의 길이를 기억한다.
기억해 둔 가장 긴 줄을 프린트 한다.
```

```
#include <stdio.h>
#define MAXLINE 10  /* maximum input line size */

int readline(char line[], int maxline);
void copy(char to[], char from[]);

/* print longest input line */
int main() {
    int len;                /* current line length */
    int max;                /* maximum length seen so far */
    char line[MAXLINE];      /* current input line */
    char longest[MAXLINE];  /* longest line saved here */

    max = 0;
    while ((len = readline(line, MAXLINE)) > 0)
        if (len > max) {
            max = len;
            copy(longest, line);
        }
    if (max > 0)  /* there was a line */
        printf("%s", longest);
    return 0;
}

/* readline: read a line into s, return length */
int readline(char s[], int lim) {
    int c, i;

    for (i = 0; i < lim - 1 && (c = getchar()) != EOF && c != '\n'; ++i)
        s[i] = c;
    if (c == '\n') {
        s[i] = c;
        ++i;
    }
    s[i] = '\0';
    return i;
}

/* copy: copy 'from' into 'to'; assume to is big enough */
void copy(char to[], char from[]) {
    int i;

    i = 0;
    while ((to[i] = from[i]) != '\0')
        ++i;
}
```

automatic variables only

```
#include <stdio.h>
#define MAXLINE 10  /* maximum input line size */

int readline(char line[], int maxline);
void copy(char to[], char from[]);

/* print longest input line */
int main() {
    int len;                /* current line length */
    int max;                /* maximum length seen so far */
    char line[MAXLINE];      /* current input line */
    char longest[MAXLINE];  /* longest line saved here */

    max = 0;
    while ((len = readline(line, MAXLINE)) > 0)
        if (len > max) {
            max = len;
            copy(longest, line);
        }
    if (max > 0)  /* there was a line */
        printf("%s", longest);
    return 0;
}

/* readline: read a line into s, return length */
int readline(char s[], int lim) {
    int c, i;

    for (i = 0; i < lim - 1 && (c = getchar()) != EOF && c != '\n'; ++i)
        s[i] = c;
    if (c == '\n') {
        s[i] = c;
        ++i;
    }
    s[i] = '\0';
    return i;
}

/* copy: copy 'from' into 'to'; assume to is big enough */
void copy(char to[], char from[]) {
    int i;

    i = 0;
    while ((to[i] = from[i]) != '\0')
        ++i;
}
```

with eternal variables

```
#include <stdio.h>
#define MAXLINE 10  /* maximum input line size */

int max;                 /* maximum length seen so far */
char line[MAXLINE];      /* current input line */
char longest[MAXLINE];   /* longest line saved here */

int readline(void);
void copy(void);

/* print longest input line; specialized version */
int main() {
    int len;
    extern int max;
    extern char longest[];

    max = 0;
    while ((len = readline()) > 0)
        if (len > max) {
            max = len;
            copy();
        }
    if (max > 0)  /* there was a line */
        printf("%s", longest);
    return 0;
}

/* readline: specialized version */
int readline(void) {
    int c, i;
    extern char line[];

    for (i = 0; i < MAXLINE - 1 && (c = getchar()) != EOF && c != '\n'; ++i)
        line[i] = c;
    if (c == '\n') {
        line[i] = c;
        ++i;
    }
    line[i] = '\0';
    return i;
}

/* copy: specialized version */
void copy(void) {
    int i;
    extern char line[], longest[];

    i = 0;
    while ((longest[i] = line[i]) != '\0')
        ++i;
}
```

automatic variables only

```
#include <stdio.h>
#define MAXLINE 10  /* maximum input line size */

int readline(char line[], int maxline);
void copy(char to[], char from[]);

/* print longest input line */
int main() {
    int len;                /* current line length */
    int max;                /* maximum length seen so far */
    char line[MAXLINE];      /* current input line */
    char longest[MAXLINE];  /* longest line saved here */

    max = 0;
    while ((len = readline(line, MAXLINE)) > 0)
        if (len > max) {
            max = len;
            copy(longest, line);
        }
    if (max > 0)  /* there was a line */
        printf("%s", longest);
    return 0;
}

/* readline: read a line into s, return length */
int readline(char s[], int lim) {
    int c, i;

    for (i = 0; i < lim - 1 && (c = getchar()) != EOF && c != '\n'; ++i)
        s[i] = c;
    if (c == '\n') {
        s[i] = c;
        ++i;
    }
    s[i] = '\0';
    return i;
}

/* copy: copy 'from' into 'to'; assume to is big enough */
void copy(char to[], char from[]) {
    int i;

    i = 0;
    while ((to[i] = from[i]) != '\0')
        ++i;
}
```

with eternal variables

```
#include <stdio.h>
#define MAXLINE 10  /* maximum input line size */

int max;                 /* maximum length seen so far */
char line[MAXLINE];      /* current input line */
char longest[MAXLINE];   /* longest line saved here */

int readline(void);
void copy(void);

/* print longest input line; specialized version */
int main() {
    int len;
    ~~extern int max;~~
    ~~extern char longest[];~~

    max = 0;
    while ((len = readline()) > 0)
        if (len > max) {
            max = len;
            copy();
        }
    if (max > 0)  /* there was a line */
        printf("%s", longest);
    return 0;
}

/* readline: specialized version */
int readline(void) {
    int c, i;
    ~~extern char line[];~~

    for (i = 0; i < MAXLINE - 1 && (c = getchar()) != EOF && c != '\n'; ++i)
        line[i] = c;
    if (c == '\n') {
        line[i] = c;
        ++i;
    }
    line[i] = '\0';
    return i;
}

/* copy: specialized version */
void copy(void) {
    int i;
    ~~extern char line[], longest[];~~

    i = 0;
    while ((longest[i] = line[i]) != '\0')
        ++i;
}
```