C □□

□□□□ □□□ □□□□□ , □□□ □□□□ □□□□□ .

ps. □□ □□ □□ □□□ □□□ □□ □□□□□ □□□ □□ □□□ ;

- 1 □ □□□□
  - 1-1. hello world □□
  - 1-2. □□□ □□ □□
  - 1-3. For □
  - 1-4. □□ □□
  - 1.5.1 □□ □□□
  - 1.5.2 □□□ □□ □□
  - 1.5.3 □□ □□ □□
  - 1.5.3 □□ □□ 3□□
  - 1.5.4 □□ □□ □□
  - 1.5.5 □□ □□ 2□□
  - 1.6 □□ (Array)
  - 1.6.1. □□ □□ 2□
  - 1.7 □□
  - 1.9 □□□□
  - 1.10 □□□□
- 2□ □ , □□□ , □□
  - 2.1 □□□
  - 2.2 □□□ □□ □□

1 □ □□□□

1-1. hello world □□

> □□ □ □□

01. C □□□□□ □□ □□ □□□ □□□ □□□□ .

02. □□□ □□□ □□□□ □ □□□□□ □□□ □ main□□ □□ .

03. □□ □□□□□□ □□□ □□□ printf□□ stdio(□□ □□□ □□□□□)□ □□□□ .

03-1 #include <stdio.h>□ □□ □□□□□□ □□□□ .

04. □□□ □□□ □□□□ □□□ □□□□□ □□□□ .

04-1. □□□□□ □□ □□ □□□ □□□ . ex. main(params)

04-2. □□ □□□□□□ □□□□□ □□□□ □□□ , □□□□□ □□□□ □□□ □ □□□□□ □□ .

05. □□□ {} □ □□□□ □□□ □□□ □□□□ □□□□□ □□□ □□

05-1. □□ □□□□□□ { □ } □ □□□ main □□□ □□□□ □□□□ □□ .

> **Hello world □□**

□□ □□□ □□□ □□□

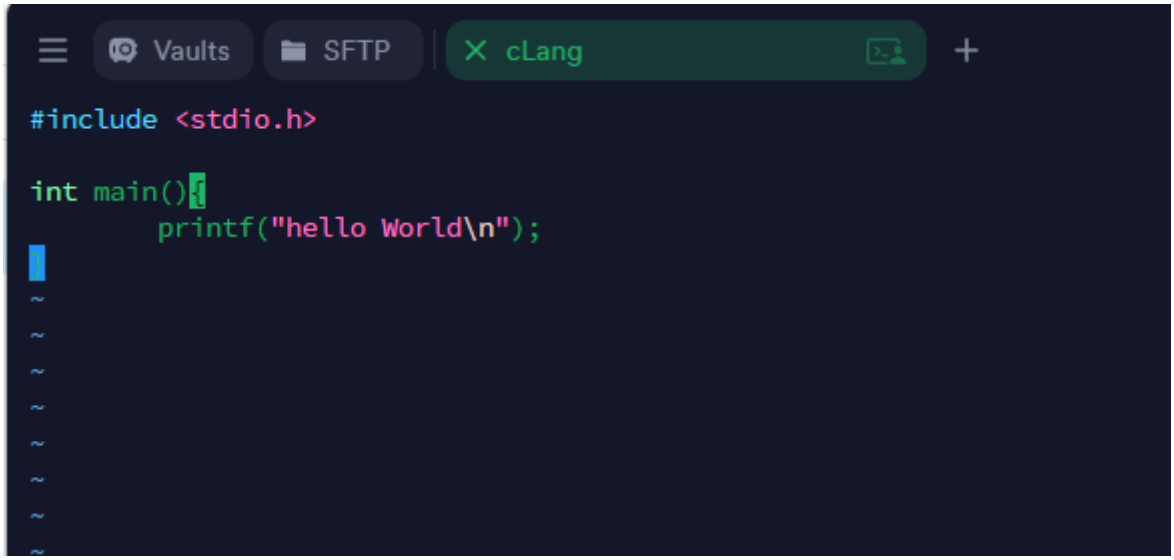
□□ □□□ **Ubuntu□ □□□□ .**

```
sudo apt-get update
```

□ □□□□ □□ □□□□□ □□□□

```
vi hello.c
```

□ □□□□ □□ c □□ □□□ □□□□ .

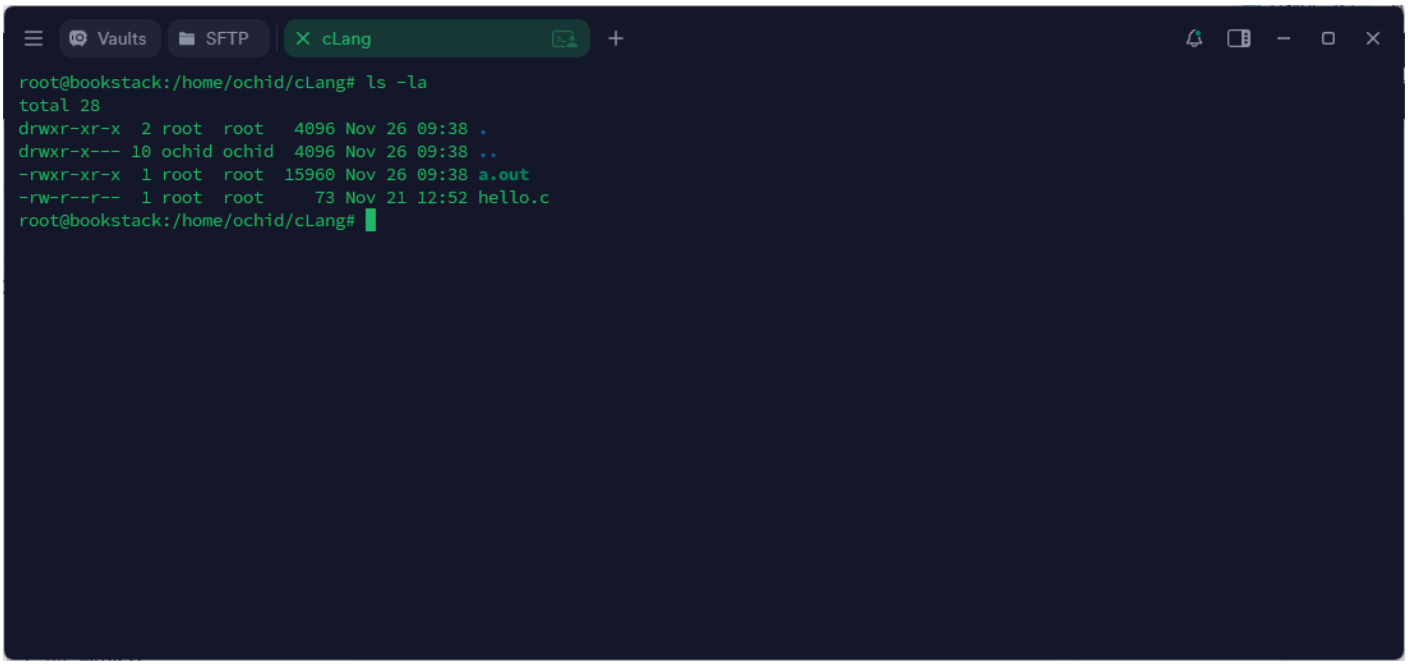



```
#include <stdio.h>

int main(){
  printf("hello World\n");
  return 1;
}
```

□ □□□□ □□ □

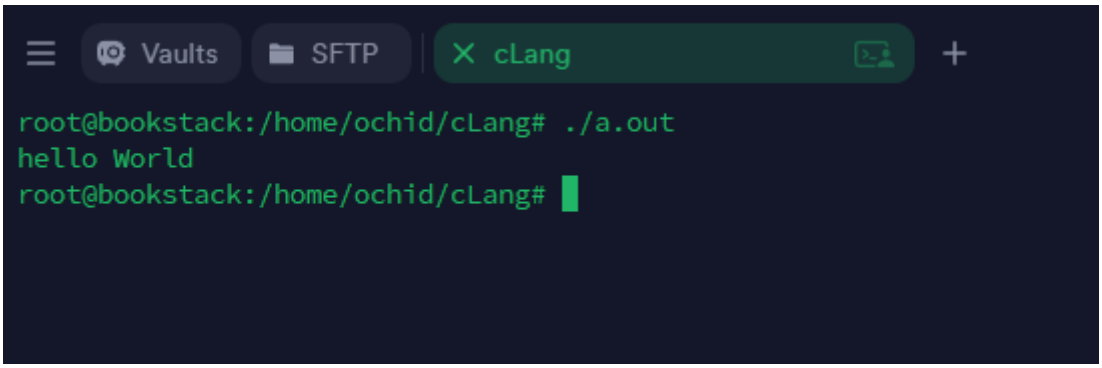
```
cc hello.c
```

□□□□ □□□□ □□□□ □□□□ .



□□ □□□ □□□□ □□□□□ **a.out** □□□ □□□ □□□□□ **.**

*□□□□ □□ □□□ □□□ □□□□□□ □□ □□□□□□ □□□□ □□□ **.**




```
./a.out
```

□□□□ □□ □□ □□□□□ **a.out** □□□ □□□□ □□□□□ **hello world**□ □□□□ □□□ □ □□ **.**

> □□ □ □□

01. printf("hello, world\n"); □□ □□□ **prinf** □□ □□□ □□□□ □□□□□ **"hello, world\n"** □ □□□□□□ **.**

02. prinf □□□ □□□□□ □□□□ □□□□ **.**

03. \n □ □□□□ □□□ **. ***□□□□□ □□ □□□ **.**

03-1. □□ □□□ \t □ tab□ , \b □ □□□□□ □□□ □□ . (\b□ □□□ □□□ □□□ □□□□ .)

1-2. □□□ □□ □□

□□ □ □□

01. □□ □□□□ □□□ □□□ □□ □□ □□□ □□ □□□ □□ □□□□ □□□□□ □□□□□ .

02. □□□ □□□ □□ , □□□□ □□ , □□ □□ □□□ □□□ □□□□ □□□□□ □□□ □□ □□ . □□□ □□□□ .

03. □□□ □□□ □□□ □□□□ □□ , □□□ □□□ □□□□ □□□□ □□ □□□ □□□□□ □□□□□ □□□□ .

04. □□ □□□□□ □□ □□ □□ □ □□□□ □□□ □ □□ .

05. /**/ □ □□ □□□ □□□□ □□□ □□□□ □ □□ .

05-1. /* □□□ □□ □□□□□ □□□ □□□□ .*/

06. C□□ □□ □□□ □□□□ □□ □□□ □□□□□ .

07. □□ □□ □ □□□□□ □□□□□□ . (□□□)

08. printf() □ □□□ □□□ □□ □□□□□□ □□ □□□□ □□□□□ □□ .

123 □□□ □□ □□ □□□ num □□□ printf()□□ □□□□□

□□□□□ print(num)□□ □□□□ □□□ , printf("%d",num) □□ □□□□ □□ □□ □□□□□□□ .

□□□□ printf□ □□□□ □□ □□ □□□ □ □□□□ □□□ □□□ □ □ □□ □□ □□□□ □□ □□ □□□□□□□ □□ .

*□□□ □□□□□ □ □□□

□□

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

```
int main()
{
        int fahr, celsius;
        int lower, upper, step;

        lower = 0;
        upper = 300;
        step = 20;

        fahr = lower;
        while (fahr <=upper) {
        celsius = 5 * (fahr-32) / 9;
        printf("%d\t%d\n", fahr, celsius);
        fahr = fahr + step;
        }
        return 1;
}
```

□□□□ □ □ □□□ □□□ □□□□□ □□ .

□ □□□□ **C** □□□ □□ □□□ □□ □ □□□ □□□□□ □□□□

```
int fahr, celsius;
        int lower, upper, step;

        lower = 0;
        upper = 300;
        step = 20;
```

□□ □□ □□□□ □□□ □ □□□ □□ □□□□ .

□□ □□□□

int lower; -> □□ lower □□□ □□□ □□ (int)□ □□□□□ ! (□□□ □□□□ int□ □□□□□ .)

lower = 300; -> □ □□□ lower□□ int□ □□□□ □□□ 300□ □□□ 300□ □□ □□ 300 □□□□ .

□□ □□□ □ □□ □ □□ .

while (fahr <=upper){

...

}

□ fahr □□ upper □□ □□□ □□ □□ while()□□□ □□□ □□ □□□□ .

□ □□□ □□□□□□ fahr □□□ step □□□ □□□□ □□ □□□□ □□□□□ upper □□ □□□ □□□□ □□□□ □□□□ □ □ □□ .

□□□ □□□□□

```
root@bookstack:/home/ochid/cLang# ./a.out
0       -17
20      -6
40      4
60      15
80      26
100     37
120     48
140     60
160     71
180     82
200     93
220     104
240     115
260     126
280     137
300     148
root@bookstack:/home/ochid/cLang#
```

□□□□ fahr □□ step□ □□ 20 □ □□□□ □□□□ □□ fahr□ celsius□ □□□□ □□□ □□ □□ .

```
printf("%3d\t%3d\n", fahr, celsius);
```

□ □□ □□ %d□ %3d□ □□□□

```
root@bookstack:/home/ochid/cLang# ./a.out
  0     -17
 20      -6
 40       4
 60      15
 80      26
100      37
120      48
140      60
160      71
180      82
200      93
220     104
240     115
260     126
280     137
300     148
```

□□□□□ □□ □□ □□□ □□□□ □□□□ □□ □□□ □ □□ . □□ □□ □□□□ □□□ **3** □ □□□ □

```
#include <stdio.h>

int main()
{
        float fahr, celsius;
        int lower, upper, step;

        lower = 0;
        upper = 300;
        step = 20;

        fahr = lower;
        while (fahr <=upper) {
        celsius = 5.0 * (fahr-32.0) / 9.0;
        printf("%3.0f\t%6.1f\n", fahr, celsius);
        fahr = fahr + step;
        }
        return 1;
}
```

□□□ □□□□ **int**□ □□ **float**□□ □□□□□ □□ □□□□ □□□□ **%d** □□ **%f** □ □□□□ **.**

%6.1f □ □□ 6□□ , □□□ 1□□□ □□□□□ □□□□ □□ □□□□□ □□□□ .

*□□ □□□ %3.0f □ □□□□□ , □□□ □□□ □□□□ □□ □□□□

5.0, 32.0 □ □□ □□□ □□ □□□ □□□ □□□ □□□ □□ □□□□□ □□□□□ .

```
root@bookstack:/home/ochid/cLang# ./a.out
  0      -17.8
 20       -6.7
 40        4.4
 60       15.6
 80       26.7
100       37.8
120       48.9
140       60.0
160       71.1
180       82.2
200       93.3
220      104.4
240      115.6
260      126.7
280      137.8
300      148.9
```

> □□ □ □□

01. int□ □□□□ □□□ □□□ □□□ □□□□□ □□□□ . □□□ □□ .

char □□□ , 1□□□

* char A □ A□ □□□ □□ A □□ 1□□□□ □□□ □□□ □ □□ .

** □□□□ ABCD □□ 1□□□ □□□ □□□ □□□ □□ char A[] = "ABCD" □□ □□□ □□ □□□ □□□ □□□□□ .

short □□□ □□□

long □□□ □□□

float □□□□□

double □□□ □□□□□

□□ □□ .

*□□□ □□ □□□ □□□□□ □□□□ □□□□ □□ □□□ □□□□ ;

02. □□ □□ □□□ □□□ □□ **.**

celsius = 5 * (fahr-32) / 9;

□ □□□

celsius = (5/9) * (fahr-32);

□ □□□ □□ □□□ **C** □□□□□ □□□□□ □□ □□□ □□□ □□□ □□□□ **. (** □□□□ □□ **)**

□□□□ **5/9** □ **0** □□□ **fahr**□ □□ □□ □□□□ **celsius**□ □□ **0**□ □□□ □□□ **.**

□□ □□□□□ □□ □ □□□ □ □□□□ □□□

1-3. For □

□□ □ □□

01. □□ □□□□ □□□□ □□□□□ □□□□□ **.** □□ □□□□□□□□ □□□ □□ **while**□ □□□□□ **for** □□□□□ □□□□□ **.**

□□

```
#include <stdio.h>



int main()
{
        int fahr;

        for (fahr = 0; fahr <=300; fahr = fahr + 20)
                printf("%3d %6.1f\n", fahr, (5.0/9.0) * (fahr-32) );
}
```

□□ □□□□ □□□ □□□□□□□□□□ **while** □□□ **for**□□□ □□□□□ **.**

fahr □□□ □□□□□ **printf** □□□ □□□ □□ □□□□ □□□ □□□ □□□□ □□□ □□□□ **.**

for□ □□ □□□ **;** □ □□□□ □□□ □□□

fahr □ □□ **0**□□ □□□□ **,**

fahr □ **300** □□ □□□ □□□□ **for**□□ □□□□ **.**

□□□□□ fahr□ 20□ □□□□ for□□ □□□□ □□□ □□□□□ □□□□ .

> □□ □ □□

□□□ □□□□ C□ □□□□ □□□ □□□□□ .

□□□ 0□□ 10000000(□□) □□ print □□ □□□ □□ □□□□□ .

```
9999994
9999995
9999996
9999997
9999998
9999999
10000000

[Done] exited with code=0 in 9.479 seconds
```

C□□□ 9.4□

```
9999992
9999993
9999994
9999995
9999996
9999997
9999998
9999999
10000000

[Done] exited with code=0 in 42.517 seconds
```

□□□□ 42.5 □

□□□□ □□ □□□ □□□□ ... □□□ □□□□□ □□□ □□□□□ □□□□□□ □□□ □□□□□ .

□□ □□ □□□ □ □□□□□

1-4. □□ □□

> □□ □ □□

□□ □□□□□□ □□□ **300,200** □□ □□□ □□□□□ .

□□□ □□□ □□□□ □□ □ □□ □ , □□ □□□ □□□ □□□ □□□ □□□ □□□

□□□□ □□ □□ □□□ □□□ □□ .

> □□

```
#include <stdio.h>

#define LOWER 0
#define UPPER 300
#define STEP 20
#define TEXT "hello world"

int main()
  {
    printf("%d\n", LOWER);
    printf("%d\n", UPPER);
    printf("%d\n", STEP);
    printf("%s\n", TEXT);
    return 0;
  }
```

```
문제    출력    디버그 콘솔    터미널    포트
● Active code page: 65001
0
300
20
hello world
* 터미널이 작업에서 다시 사용됩니다. 닫으려면 아무 키나 누르세요.
```

□□□□□ □□□ □□□□ □□□□ □□

*□ □□□□□ □□□□ □□□□□ □□□□□ □□□□ **VSCode□ □□□□ .**

https://velog.io/@jamkris/Visual-Studio-Code%EC%97%90%EC%84%9C-C%EC%96%B8%EC%96%B4-%EC%BB%B4%ED%8C%8C%EC%9D%BC-%ED%95%98%EA%B8%B0

vscode C □□□ □□□ □ □□□ □□□□ .

> □□ □ □□

□□ □□□ □□□□□ ;□ □□□□□ □□□ □□□ □□ ;□ □□ □□□ □□□□ .

1.5.1 □□ □□□

> □□ □ □□

□□□□□□ □□ □ □□□ □ □□□□ □□ □□ □□□ □□□□ □□□□ .

□□□□ □□□□□ **input()**□ □□ □□□ □□ □□□ □ □□□ □□□ □□ .

□□ □□□□□ □□ □□ □□ □□ **getchar()** □

□□ □□□ □□□□ □□ **putchar()**□ □□□□ .

□□ □□□□□□ **EOF** □□ □□□ □□□□ . **End of File** □ □□□□□ , □□□ □□ □□□□ .

□□□□□ □□□ □□□ □□ .

1. □□□ □□ □□□ .

2. while

3. □□□ □□□□ .

4. □□ □□□ □□□□□ .

> □□

```
#include <stdio.h>



int main(){
  int c;

  c = getchar();
```

```
while (c != EOF) {
  putchar(c);
  c = getchar();
  }
return 0;
}
```

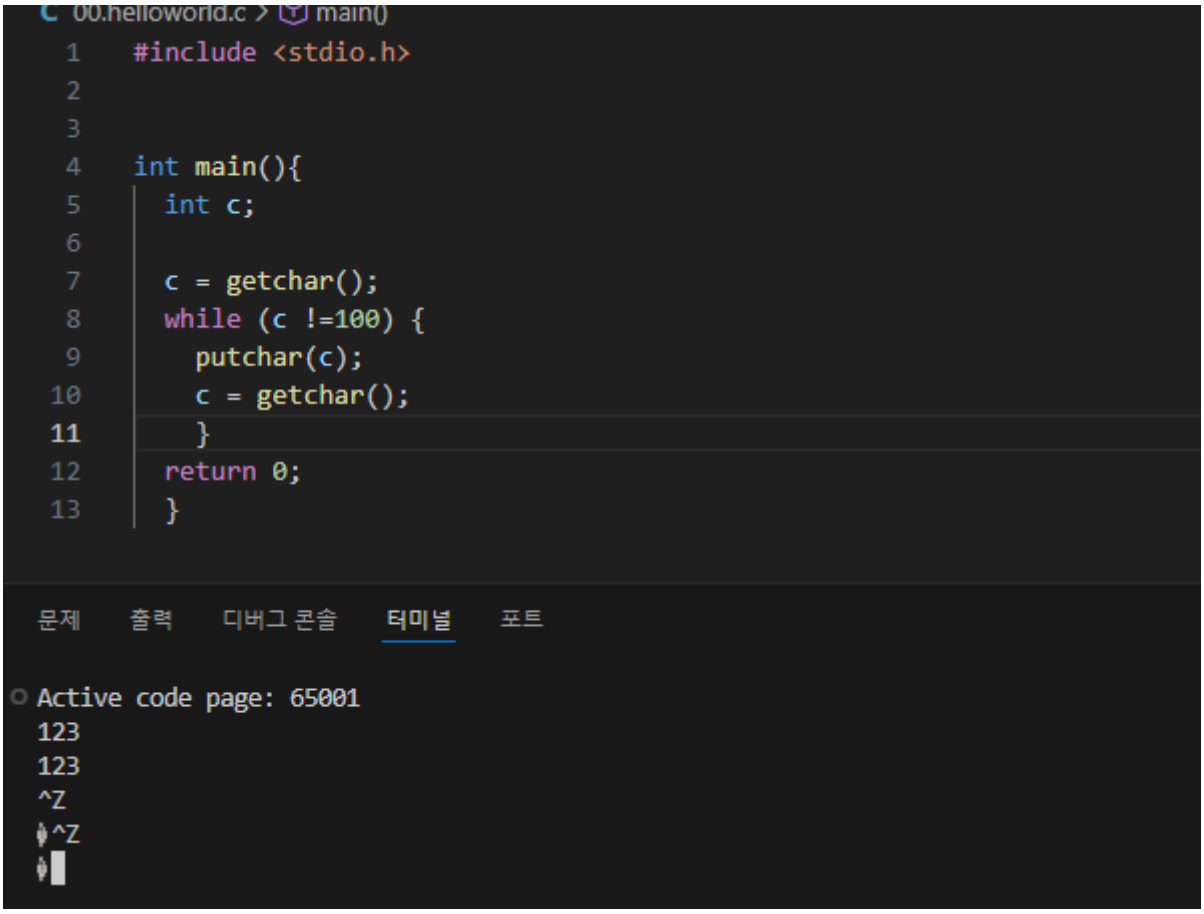
c □ □□ , while□□ □□□□ □ getchar□ □□ □□ □□ □□

□□ □□ □□ EOF □ □□□ □□ □□□□ .

EOF □ □□ -1 □□□ 0 □□ □□□□□ ,

□□□□ □□□ □□□ -1 □ □□□□ □□□□ □□ □□□ .

EOF □ □□□□ control + z □ □□ □□□ enter □□ □□□□ □□ .



c □ EOF□□ □□□□ □□□ □□□ □□ control + z + Enter □□□ □□ □□□ □□ □□ □□□ □ □□ .

□□□ □□□□ □□□ □□□□ □□ □□ □□□ □□□□ □□ □□ □ □ □□ .

putchar() □ □□□□□ □□□ □□□□□ □□ □□□□□

> □□ □ □□

□□□□ □□ □□□□ **C** □□□ □□□□□ □□□□□ □□□□ □□□□ □□ **.**

□□□□□ □□□ **...**

1.5.2 □□□ □□ □□

> □□ □ □□

□□□ □□□□ □□ □ □□ □□□□□□ □□ □□□ □□□□ .

□□□□ □□ □□□ □□ □□□ □□□□□

□□□□ □□□ □□ □□ □□□ □□ □□□□□□ □□□□□ **ex. abcd -> 4**

> □□

```
#include <stdio.h>



int main(){
  int c;
  int count;

  c = getchar();
  count = 0;

  while(c != EOF){
    ++count;
    c = getchar();
  }
  printf("%d\n", count);
  return 0;
  }
```

□ □□□ □□ □□□□ □□□□ □□□ □□ □□□□ □□ □□□ .

```
#include <stdio.h>

int main(){
  long nc;
```

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

getchar□ □□ □□□ □□□□ □□□ □ □ □□□□□ .

long□ 16□□□ □ □□ □ □□ int □□ □ □□ □□ □□□ □ □□ □□□□ .

> □□ □ □□

1.5.3 □□ □□ □□

> □□ □ □□

□□ □□ □□□ □□ □□□ □□ □□□ **\n** □□□ □□□ □□ **.**

"□□ **\n**□□□ **.**" □ □□□□

□□

□□□

□ □□□ □□□ □□□ **.**

□□ □□□□□□ **if** □□ □□□□ □□ **.**

> □□

```
#include <stdio.h>

int main()
  {
  int c, nl;

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

□ □□□□ □□ **while** □ □□ □□□ **(c = getchar()) != EOF** □□ □□□□□□ □ □ □□ **.**

!= □ **=** □□ □□ □□□ □□ □□□ **c = getchar()** □ □□□ □□ □□ □□□□□ □□ □□ **.**

if□□ □ □□□□ □□□ □□ □□□□ □□□ □□ □□ if □□ □□□ □□□ □□□ □□□□ .

□□ □□ □□ □□□□ Enter□ □□□ □□ □□□ □□□ □□□□□ \n□ □□□ □□ □□□ □□□□ □□□ Enter □□□ □□□ □□ □□ .

```
문제    출력    디버그 콘솔    터미널    포트

Active code page: 65001
sd
f
sd
fs
df

^Z
6
```

□□ \n □ □□ □□□□ □□□□□□□ □□□ □ □□□ .

\n□ □□□□□□ □□ □□□□□ □□□□ □□□□□□ □□□□ □□ □□ □□□□ □□□□ .

□□ \n□ □□□□□ □□ □□

```
Active code page: 65001
asdfa
sdf
a
sdf
asdf
^Z
0
```

□□□ □□ □□□□□ □□□□ \n□ □ □ □□ .

> □□ □ □□

□□□□□□ □□□ □□□□ □□□ .

1.5.3 □□ □□ 3□□

□□ □ □□

예 제 1-8

빈칸, tab과 행의 갯수를 세는 프로그램을 작성하라.

예 제 1-9

한 파일을 읽어서 그 중 빈칸이 연달아 나오면 그것을 모두 한 칸으로 만들어 출력하는 프로그램을 작성하라.

예 제 1-10

한 파일을 읽어서 그 중에 tab이 있으면 그것을 \t, backspace가 있으면 그것을 \b로 그리고 \가 있으면 그것을 \\로 대치해서 출력하는 프로그램을 작성하라.

□□□ □□ □□□ □□□ .

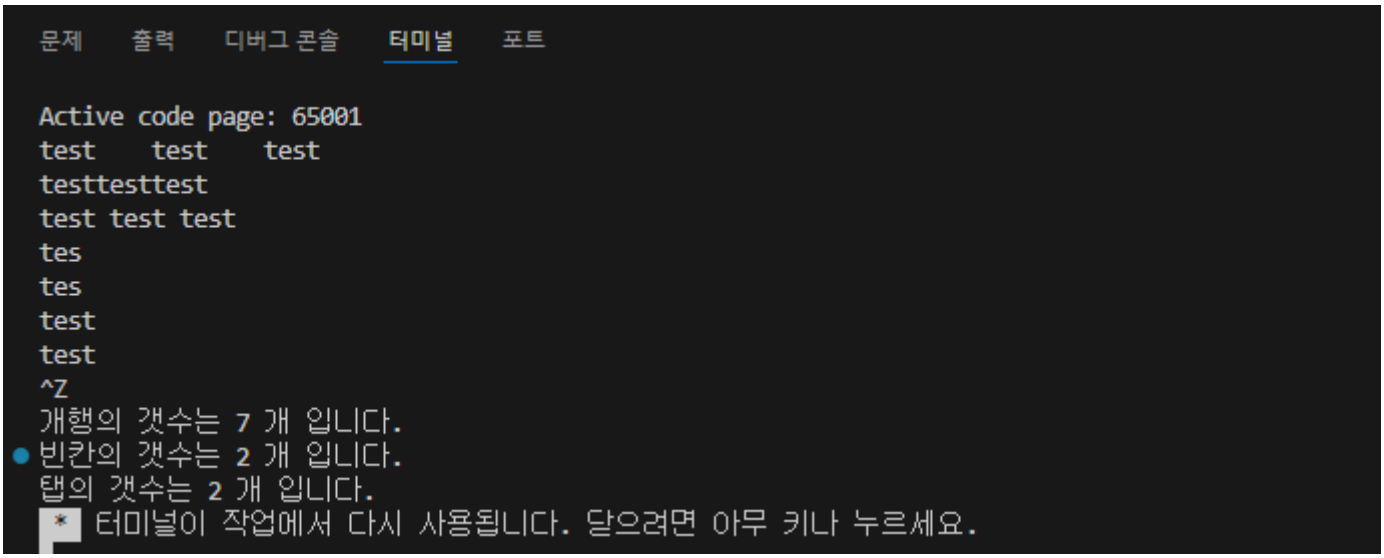
□□□ □□□ □□ □□□□□ **EOF** □□□ □□□ □□□□ □□□□ □□□□ □□□ □□□ □□ □□□ □□ □□ □□ .

□□

1. □□□ □□ - □□ , tab, □□ □□□ □□ □□□□

```
#include <stdio.h>

int main(){
    int c;
    int blankCount = 0;
    int tabCount = 0;
    int lnCount = 0;
```

```
    while ((c = getchar()) != EOF){
        if(c == '\n')
            ++lnCount;
        if(c == ' ')
            ++blankCount;
        if(c == '\t')
            ++tabCount;
    }
    printf("□□□ □□□ %d □ □□□.\n□□□ □□□ %d □ □□□.\n□□ □□□ %d □ □□□.\n", lnCount, blankCount,
tabCount);
    return 0;
}
```



□□□ □□□ □□□

*□□ □□□□□ □ □□□ □□□ □□ □□□ .

2. □□□ □□ - □ □□□ □□ □ □ □□□ □□□ □□□ □□□ □□ □□□□ □□□ □□□□ □□□□□ □□

□ **if** □□□ □□□ □□ □ □□ □□ □□□ □□ □□□ □□ □□□□ □□□

```
    int c;
    int firstBlank;

    firstBlank = 0;
    while ((c = getchar()) != EOF){

        if (firstBlank == 0){
            // printf("[!] firstBlank□ 0 □□□ putchar□ □□□□□.\n");
            putchar(c);
        }
        if (firstBlank == 1 && c != ' '){
            firstBlank = 0;
            // printf("[!] □□□□ □ □□, firstBlank□ 0(□)□ □□□□□.\n");
        }
        if (c == ' '){
            firstBlank = 1;
            // printf("[!] □□□□ □□, firstBlank□ 1(□)□ □□□□□.\n");
        }
    }



    return 0;
}
```

□□□□ □□□□ ; □□□ getchar□ □□□□□ □□□□□□□ ; □□ firstBlank □□ □□□ □□□□ □□□□ □□ □□□□□ □□□□ .

□□ □□ □□ □□□□□□ □ □□□□□ □□□□□□□ □□□□ .

3. □□□ □□ - tab□ \t □ backspace□ \b□ \ □ \\ □ □□□□ □□□□ □□□□

□ □□ □□ □□□□ □□

>

□□ □ □□

1.5.4 □□ □□ □□

> □□ □ □□

□□□□ □□ □□ □□ □□□□□□ .

□□□ □□□ □□ □□□ □□ , **tab,** □□ □□□ □□□ □ □□□ □□□□ □□□□ .

□□ □□□□□□ || □ □□ □□□ □□□□ □ , □□ , □□□ □□□ □□ .

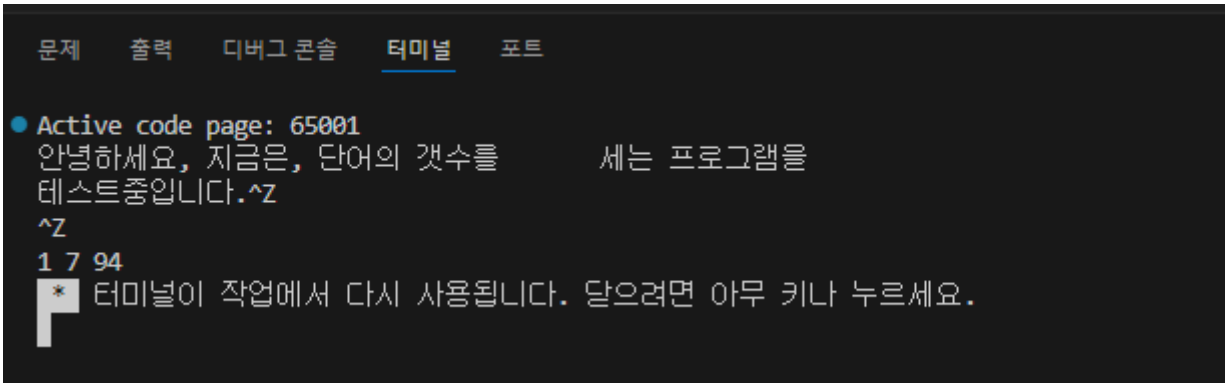
> □□

```
#include <stdio.h>

#define IN 1
#define OUT 0

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);
}
```

□□ nl, nw, nc □ □□ line, word, count □□ □□□□ .

state□□ □□□ □□□□ □□□ □□ □□□ □□□□ □□□ □□□ □□ state□ out □□ □□□ □□ □□ (nw)□ 1 □□□□ ,

□□ state□ IN□□ □□□ .



□□□□□ □□□ □□□□ EOF□ □□□□ □□ , □□ □ , □□□□□ □□□□□ .

if □□□ || □ or □□□□ □□□□ if(a || b || c) □□□ a,b,c □ □□□□ □□ □□ □□ if □□ □□ □□ □□□ □□□□ .

□□ □□□ && □ and □□□□□ □□□□ .

> □□ □ □□

1.5.5 □□ □□ 2□□

> □□ □ □□

□□ □□□□ □□□□ □□ □□ □□□ □□□□ □□ .

□□ □□□□□ □□ □□ □□□ □□□□ □□□ □ □□ □□□□□ □□□□ □□□□ .

□□□ .

> □□ □□

1. □□ □□□□□ □□□□ □□□ □□ □□□□□ □□□ □□ .

*□ □□□ □□ □□ .. □□□□ □□□ □□ □□□□□□ .. □□ □□ □□□ □□□□ □□□□□□□ .. □□□ □□ □□□□

2. □□□ □□ □ □□ □ □□□ □□□□ □□□□□ □□□ □□ .

```
#include <stdio.h>

#define IN 1
#define OUT 0

int main(){
    int c, state;

    state = OUT;
    while((c = getchar()) != EOF){
        if (c ==' ' || c =='\t'){
            state = OUT;
```

```
            c = '\n';
            putchar(c);
        }
        else{
        putchar(c);
        }
    }
}
```

```
Active code page: 65001
안녕하세요     지금    은     단어마다      개행으로 출력하도록 만든 프로그램      입니다.
안녕하세요
지금
은
단어마다
개행으로
출력하도록
만든
프로그램
입니다.
```

□ □□□ □□□ □ □□□□ □□ .

□□ □□ \t □ □□□□ □□ □□□ □□□□□ □□□□□ .

> □□ □ □□

1.6 □□ (Array)

> □□ □ □□

□□□ □□□ □ □□□□ □□□ □□□ □□□ □□□ □□□ □□□ ,□ □□ □□□□ □□□□ □□ □□□□□ □□□□□

□□□□ □□□ □□□□□□ .□□□ □□□ □□□□ □□□□ □□□ □□□□□ □ □□□ □□ .

□□□ □□ □□ ,□□□ □□□□ □ □□□ □□□ □□ □□ □□□□ □□□ □□□□□ □□□□□□ □□□ □□ .

□□ □□□□ □ C□ □□□ □□□ □□ □□□ □□□□□□ .

> □□

□□ □□□ □□□ □□ □□□ □□□ □□ □□□□ □□□□ □□□□□□ .

```
#include <stdio.h>



int main(){
    int c, i, nwhite, nother;
    int ndigit[10];
    nwhite = nother = 0;

    /*□□ □□ □□□□ 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(", white space = %d, other = %d\n", nwhite, nother);
}
```

```
int ndigit[10];
```

□ □□□ □□□□ □□□□ .

ndigit[0] □□ **ndigit[9]** □□ □□ □□□ □□□ □□□ □ □□ .

□□□ **0~9**□ □□□□ □□ **,** □□□ □□□ **0**□□ .

```
(c >= '0' && c <= '9')
```

□ □□□ □□□□ □□□ □□□□ □□□□ □□□□ □□□□ .

0□□ **9**□□ □□□ □□□□□ **++ndigit[c-'0];** □□□ □□□□ **. (?** □□□ □□□ □□□ **.)**

if (c >='0' && c <='9') ...

라는 부분은 읽어들인 문자가 숫자인지 아닌지를 검사하는 부분이다. 숫자인 경우, 그 숫자가 뭔지를 알려면

c - '0'

을 하면 된다(잘 생각해 보기 바란다). C에서 char와 int형이 섞여 있는 계산을 할 때 모

□ □□□□□□ □□ .

□□□ □□□□ □□

```
C□□ char □ int □□ □□□□□ □□□□□□ □□□□□□ int□□□ □□□□□□□□□□ □□□□ □□□□□□ □□.
□□□c□'1'□□□ □□ c□ □□□□□ 1□□□□□ □□ 1□ascii□□□□.
(□□ascii □□ □□ □□□□□□□□□□□□□□□□□□□□ □ □□□□□□ □□□□□ □□).
□□□c□□'0'□□□ □□□□□□□□□□□□□□□□.
□ □□(0~9□□□□□□)□□□□□□□□□□□□□□ □□.
```

□□□□ □□□□ **ascii** □□□□ □□□□□ **.**

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

□□□□□ **10□□ 16□□ □□** □□ □□□□□ **,** □□□ □□□□

□□□ □□□□□ "□□□□□□ □□ **3**□ □□□□□□ **.** □ □□□□□ **,** □□□ □□□□□□ □□ □□ □□□□□ □□□□ □□ **.**

(c >= '0' && c <= '9') □□□□ **c** □ **"3"** □ □□□□□□ **,** □□□□ □□□□

ndigit[3] □ □□□ **1**□ □□□□□ □□ **. (++ndigit[c-'0'];** □□ **)**

□□□ **ndigit[c]** □ □ □□ **c**□□ □□□□ **3**□ □□□□□ □□ □□□□ **ascii**□□ □□□ **51**□ □□□□ **. (**□□ □ □□ **)**

□□□ □□□ **51**□ **3**□□ □□□□□□□□ **51 - 48 = 3 , 48**□ **ascii**□□ □□□ **0**□ □□□ **.**

□□□ **ascii** □□□□ □□□□□ □□ □□□□ □□□□□□ □□□□

c - '0' = 3□ □□

3□ □□□ □□□

ndigit[3] □ □□ **1**□ □□□□ **.**

□ □□ □ □□□□□□ □□□ □□ □□□ **;**

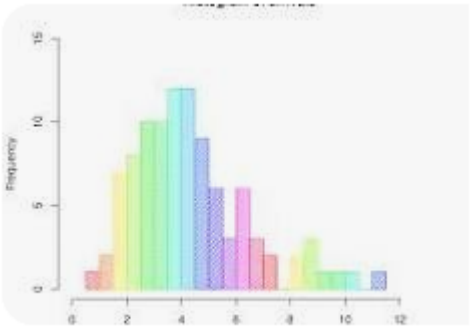
nwhite nother .

>

1.6.1. □□ □□ 2□

□□ □ □□

□□□□□ - □□ □□ □□ □□ □□□ □□ □□□□ □□□ □□□ .



□□□□□□ □□ □□□ □ □□□ □□□□ . □□ □□□□□ □□□□ □□□ □□ □□□□ □□□□ ; □ □□□

* □□□ □□□ □□□□□ □□□□ □□□□□□ □□ □□□ □□ .

□□ □□

1□ □□ □□□ □□□ □□□ □□ □□□□□□ □□□□ □□□□□ □□□□

```
#include <stdio.h>
#define IN 1
#define OUT 0
#define UPPER 10

int main(){
    int c, i;
    int ndigit[30];
    int wordLength = 0;
    int state;
```

```
    /*□□ □□ □□□□ 0□□ □□□□□□□; □□ □□□ □□□□□□□; */
    for (i = 0; i < UPPER; ++i){
        ndigit[i] = 0;
    }

    state = IN;
    while ((c = getchar()) != EOF){
        if (state = IN && c != '\n' && c != '\t' && c != ' '){
            ++wordLength;
        }
        else {
            ++ndigit[wordLength];
            wordLength = 0;
            state = OUT;
        }

    }
    printf("□□□ □□□ 1□ □□□ □□□ %d\n", ndigit[1]);

}
```

□□□ □□□□ □□ □□□□ □□ □□□□□□ □□□□ . 1□ □□□ □□□ □□□□ □□□ □□□□□ .

```
Active code page: 65001
a
a
a
a a a
^Z
단어의 길이가 1인 단어의 갯수는 6
* 터미널이 작업에서 다시 사용됩니다. 닫으려면 아무 키나 누르세요.
```

□□□□□ □□□□ □□ □□

```
#include <stdio.h>
#define IN 1
#define OUT 0
#define UPPER 10
```

```
int main(){
    int c, i;
    int ndigit[30];
    int wordLength = 0;
    int text;
    int state;
    int index;

    /*□□ □□ □□□□ 0□□ □□□□□□; □□ □□□ □□□□□□; */
    for (i = 0; i < UPPER; ++i){
        ndigit[i] = 0;
    }

    state = IN;
    while ((c = getchar()) != EOF){
        if (state = IN && c != '\n' && c != '\t' && c != ' '){
            ++wordLength;
        }
        else {
            ++ndigit[wordLength];
            wordLength = 0;
            state = OUT;
        }

    }

    for (index = 0; index < UPPER; ++index){
        printf("[%d]", index);
        for (text = 0; text < ndigit[index]; ++text){
            printf("=");
        }
        printf("\n");
    }

}
```

□ □□□□ □□ □□ □ □□□ □□□ □□□ = □□□ □□□□ □□□□ □□□ □□□□ □□□□ .

```
Active code page: 65001
a a a a a ab ab ab ab ab ab ab ab ab abc abc abc abc abc abc abc
abcd
abcd
abcd
abcd
abcdefg
^Z^X^Z
[0]
[1]=====
[2]=========
[3]=======
[4]====
[5]
[6]
[7]=
[8]
[9]
* 터미널이 작업에서 다시 사용됩니다. 닫으려면 아무 키나 누르세요.
```

2□ □□ - □□□ □□□□ □□□□□ □□□□ □□□□□□ □□□□ □□□□□ □□□□□

□□ **ndigit[]** □□□ □□□ □□ □□□□ □□ □□□ □□□ □□□□□□ □□□□□ □□ **?**

□□ □□ □□ □□ **ndigit** □□ □□ □□□ □□□□ □□□□ □□ □□ □□□□□ **.**

□□□□ □□ □□□ □□□ □□□□ □□ □□□□ □□□□

□□ **getchar** □ □ □□□ □□□□□ **..** □□□ □□□ □□□ **..**

□□□ □□ □□□ □□□□ □□□ □□□□□ **..** □□ □□□ □□□ □□ □□□ □□□ □□ □□□□□□□ **..**

***** □ □□□□□ □□□ □□□□□ □□□ □□□ □□□□□ **; a** □□□□□ **b** □□□□□ **..**

□□□ □□□□ □□□ □□□□□ □□ □ □□□□□□ □□□ **ascii**□ □□□□ □□□ □□□□□ **.**

65	0x41	A	97	0x61	a
66	0x42	B	98	0x62	b
67	0x43	C	99	0x63	c
68	0x44	D	100	0x64	d
69	0x45	E	101	0x65	e
70	0x46	F	102	0x66	f
71	0x47	G	103	0x67	g
72	0x48	H	104	0x68	h
73	0x49	I	105	0x69	i
74	0x4A	J	106	0x6A	j
75	0x4B	K	107	0x6B	k
76	0x4C	L	108	0x6C	l
77	0x4D	M	109	0x6D	m
78	0x4E	N	110	0x6E	n
79	0x4F	O	111	0x6F	o
80	0x50	P	112	0x70	p
81	0x51	Q	113	0x71	q
82	0x52	R	114	0x72	r
83	0x53	S	115	0x73	s
84	0x54	T	116	0x74	t
85	0x55	U	117	0x75	u
86	0x56	V	118	0x76	v
87	0x57	W	119	0x77	w
88	0x58	X	120	0x78	x
89	0x59	Y	121	0x79	y
90	0x5A	Z	122	0x7A	z

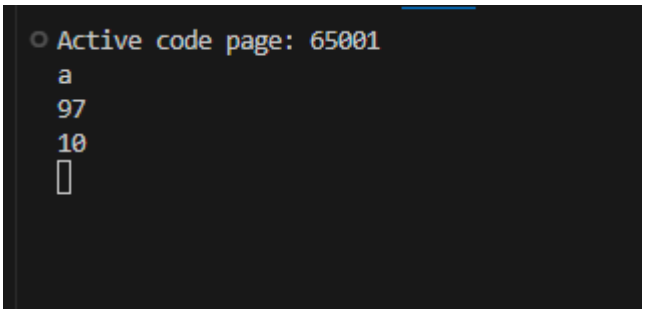
□□ □□□□ □□ □□ □□□□ □□□□ □□

□□□□ □□□ **26□ □□□ ndigit[26]□□ □□□□□ □□□□**

```
#include <stdio.h>



int main(){
    int c, i;
    int ndigit[26];
    int text;
    int word;

    for (i = 0; i < 26; ++i){
        ndigit[i] = 0;
    }

    while((c = getchar()) != EOF){
        printf("%d\n", c);
```

```
    }


}
```

□□□□ □□ a □ □□□□ Enter□ □□□□□□ .



a □ ascii□□ 97 □□ ascii□□ 10□ □□□□ . Enter □□ 10□□ □□□□□ □□ □□ .

ndigit □□□ 0□□ 25□□ □□□□□ c□ □□□□ □□□ 97□□ □□ □□□ □ □□□ □□□ □□□□ □□□

□□ □□□□ □□□□□ □□ □□ □□ □□□ □□□□□ □□□□ □□□ □□□

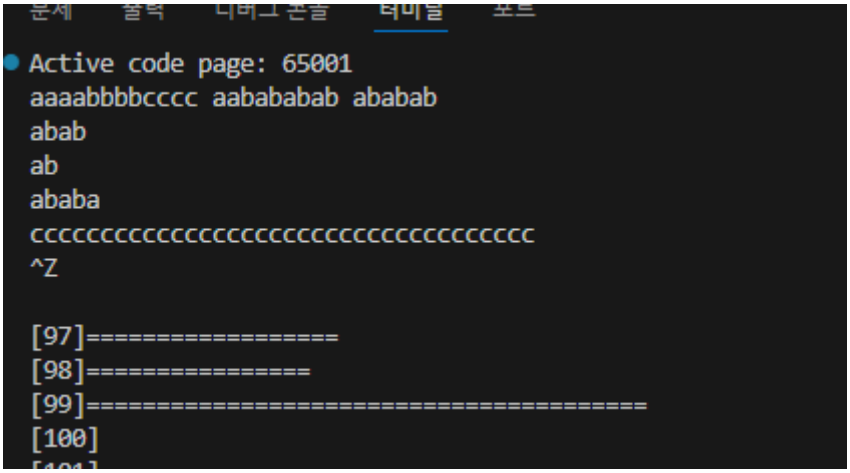
```
while((c = getchar()) != EOF){
        if(c >= 97 && c <= 122){
            c = c-'a';
            printf("%d\n", c);
        }
    }
```

if □□□ □□ , □□ □□□□ □□ ndigit □□□□ 1□ □□□□ □□□ .

```
#include <stdio.h>



int main(){
    int c, i, e;
    int ndigit[26];
    int index;
    int range;
    int alphabet;
    /*□□ □□ □□□□ 0□□ □□□□□□; □□ □□□ □□□□□□; */

    for (i = 0; i < 26; ++i){
        ndigit[i] = 0;
```

```
    }
    index = 0;
    while((c = getchar()) != EOF){
        if(c >= 97 && c <= 122){
            index = c-'a';
            ++ndigit[index];
            index = 0;
        }
    }


    for (range = 0; range < 26; ++range){
        alphabet = range + 97;
        printf("\n[%d]", alphabet);
        for (e = 0; e < ndigit[range]; e++){
            printf("=");
        }
    }
    printf("%d\n", ndigit[0]);



}
```

□□□□ □□□□



□□ □□□□ □□□□□ **... 97□ □□ □□□ □□□□□ □□□ □□□ □□□ ..?**

□□□ □□□ □□ □ □□□ □□□ □□□□ **.**

> □□ □ □□

1.7 □□

> □□ □ □□

□□□□□ □□□□ □□□ □□ . □□ □□ □□ □□□ □□ □□ □□□ □□ □□□ □□ □□ ?

□□□□□ □□□ □□ □□□ □□ □□□ □□□ .

□□ □□□□□ □□□□□ □ □□□ □□□□ □ □□□ . (□□□ 10 □ □□□□ □□□ □□)

□□ □□□ □□□ □□□ □□ □□□ □□□□ □□□ □□ □ □□□ □□□ □□□□□

□□□ □□□ □□ □□□□ □□□□ .

□□□□□ □□□ □□ □□□ □□□□ □□ □□□ □□□ .

□□□□ □□□□□ □□ □□□□□ , □□□□ , □□□□ □□□ □□ □

□□□□□ □□□□□ (□□□ □□ □□□ □□ □□□□ □□□□ .)

□□ □□□ □□□□ □□ □□ □ , □□□□□ □□□□ □□□□ □□□□□ □□□ □□ .

1. □□ □□

2. □□□ □□□ □□

3. □□□ □□□□ □□ □□ □□ □ □□

□ □□ □□□ 3□□ □□□ □□□ □□□□□ □□□ □□□□ □□□□□ □□□□□ □□□□ .

□□□□ □□□□□ □□□ □□□ □□□□□ □□□□ □ □ □□□ □□ , □□□□ □ □□□□
□□□□□ .

> □□

```
#include <stdio.h>


int power(int , int );


int main()
{
    int i;


    for (i = 0; i < 10; ++i)
        printf("%d %d %d\n", i, power(2,i), power(-3,i));
        return 0;
}



int power(int base, int n)
{
    int i, p;


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

main □□□ □□□□□ □□ □□□ □□ □□ , □□□ □□□ □ main□□□ □ □□□□ □□□□□ .

□□ power □□□ □□□□□ power□□ □□□□□ base□ , n □□□ □□□□ , "base" □ "n"□□□ □□□□ □□ □□□□ □□□□ .

□□□ □□□□□ □□□ □□□ □□□□□□□ □□ □□ .

□□□ □ □□□ □□□ , □□□□ □□□□ □□□□ □□□□ □□ □□□ □□□ □□□ □□□□□□□ □□ □□□ □□□□□□□ .

power(2,i) □ □□□ power□□ □□ □□ base, n □ □□ □□□ □□ □ □□□□ □□□□ return □□ □□ □□□ .

return □ □□ □□□ □□ void □□ □□□ □□□□ □□□□ □□ □□□□□□□ □□□□□□□

> □□ □ □□

1.9 □□□□

□□ □ □□

□□ □ C□□ □□ □□ □□□□ □□□ □□□□ (□□)

□□ □□ □□ □□□□□ □□ □□□□□

□□ □□ □□ ,□ □ □□ □ □□ □□□□ □□□□□ □□□□□

while(읽을 행이 있으면)
    if(지금읽는 행이 지금까지 가장 길었던 행보다 길면)
        그 행과 행의 길이를 기억
        가장 긴 행을 출력

□□

```
#include <stdio.h>
#define MAXLINE 1000

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

int main()
{
    int len;
    int max;
    char line[MAXLINE];
    char longest[MAXLINE];
    max = 0;
    while ((len= getline(line,MAXLINE)) > 0 )
        if (len > max) {
            max = len;
            copy(longest, line);
        }
```

```
        if (max > 0)
            printf("%s", longest);
        return 0;
}

int getline(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;
}

void copy(char to [], char from[]){
    int i;

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

□□□ □□□□□ □□ □□□ □□ □□ .

(□□□ □□□□□ □□□ □□ . getline □□□ □□□ □□ □□□□□□ □□ □□□□□ □□□ □□□□ □□□□□)

□□□ □ □□□□□ □□□ □□□ □ □□□ □□□ .. □ □□ □□□□ □□□ □□ □□□ , □□ □□□□□ □□□ □□□ □□□□ □□□ □□□□ □□□□□ □□□□ □□□□□

MAXLINE = 1000□□ □□

getline □□ : □□□□ s, □□ lim □□□□□ □□□ MAXLINE -1 □□ □□ EOF □ □□□□ \n □ □□□□□ s□ i□□ □□□ c□ □□□ .

□□ \n□ □□ s[i] □ \n □ □□ i □ 1□ □□□ ? (□□ for□□□ □□ c != '\n' □□□ □□□ □□□ □□□ if □□□ □□ ?)

□ for □□ □□ □□ □□ □□ s[i] = c; □□ □□ ;

□□□□ □□□□ □□ □□ □□ □□ □□□□ **s** □ □□ **,** □□□ □□□□□ **null** □□ **,** □□□ □ □□□□ **(i)** □ □□

main □□ **: getline**□ □□□□ □□□ □□ □□□□ □□ □□ □□ □ □□□□ □□ **,** □□ □□□ □□ □□ □□ □□□ **(max)**□ □□□ □ **copy** □□ □□

copy □□ **:** □□ □□ □□□ **\0**□ □□□ □□ **to** □□□□□ □□□ □□□□□ **. (**□□□□ □□□ □□□ □□□□ **..)**

□□□□□□ □□ **i = t** □□□□□ □□ □□□□ □□□ □□□□ **C**□□□ □□□ □□□□□ □□□□ □□□ □□□ □□□ **..**

> □□ □ □□

□ □□□□ for□□ □□□□□□ , while□□ □□□□ □□□□□ □ □□□ □□□ □ □□□ .

□□□□ □□□□□ □□□ □□□□ C□ □□ □□□ ... □□□ □ □ □□□ □□□□ .

1.10 □□□□

> □□ □ □□

□□ □□□□ □□ □□□□□ □□ **main□ longest** □□ □□□ □□□ **.**

□ □□□□ **main** □□ □□□□ □□□ □□□ **.**

□□ □□□ □□□ □□ □□□□ □□□□ **, □□** □□□ □□ □□ **. □□** □□ **(local)** □□□□ □□ **.**

□□ □□□□ □□□□ □□□□ □□□ □□ □□ **(global)□□** □□ **.**

□□ □□□ □□□□□ **extern** □□□ □□□□ □□□□ **.**

□□ □□□ □□□ □□□□□ □□□□ **, □** □□□ □□□□ □ □□□□□ □□□□□ □□ **.**

□□□ □□ □□ □ □□ □□ □□□□□ □□□□ **longest□ max** □□□ **line□** □ □□□ □□□□ □□□□□ □□□□□ □□□ □□□ □□ **.**

> □□

```
#include <stdio.h>

#define MAXLINE 1000

int max;
char line[MAXLINE];
char longest[MAXLINE];

int my_getline(void);
void copy(void);

int main()
{
    int len;
    extern int max;
```

```
    extern char longest[];

    max = 0;

    while ( ( len = my_getline()) > 0)
        if (len > max) {
            max = len;
            copy();
        }

    if(max > 0)
        printf("%s", longest);
    return 0;
}

int my_getline(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;
}

void copy (void)
{
    int i;
    extern char line[], longest[];
    i = 0;
    while ( ( longest[i] = line[i]) != '\0')
        ++i;
}
```

□□□ □□ □□□□ □□ □□ . □□ □□□□□□ □□□□ □□□ □□ □□□□ □□□□□ □□
□

extern □□□□ □□ □□□ □□ □□□ □□ □□□□ □□ .

□□□ □□ □□□□ □□□□□ **void**□ □□□ □ □□ □□□ □ □□ .

□□□ **void**□ □□□ **copy**□ □□ □□ □□ □□□ □ ,

□□□□□ □□□ **void**□ " □ □□□ □□□□□ □□ □□ □□□ " □□ □

> □□ □ □□

□□□□□ □□□□ □□ □□ □□□ □□□□ □□□□ □□□ □□ □□ □□□ □□□ □□□□ ,

□□□□ □□□ □□□ □□□ □□□□ □□□□ □□□□□ □ □□□□ □□ □□ . □□□□
□□□

1□□ □□□ □□□ **,** □□ □□□□ □□□ □□□ □□□ □□□□ □□□ □□□ □□
□□□□□□ **.**

2□ □ , □□□ , □□

2.1 □□□

> □□ □ □□

□□□ , □□□□□ □□□ □□□ □□

□□□□ □□□ □□□ □□□□□ , □□□ □□□ □□□ □□ , "_" □ □□□ □□□ , □□□ □□□ □□□ □□

□□□□ □□□□ □□□ □□□□ .

C□□□□□ □□□□□ □□□□□ □□□□ , □□□□□ □□□□ □□□ .

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□ □□□ □□□□ □□□□ □□□ □□□ 31□□ □□□ □□□□ .

□□□□□ □□□□ □□ □□ □□□□ 31□□ □□ □□ □□□ □□□□ □□□ □□□ , □□□□□ □□ □ □□ □□□ □□□ □ □□ □□ □□ □□ . (□□□ □□□ □□ □□ □□□ □)

□□ □□□ □□□ ANSI □□□ 6□ □□□ □□□□ □□□□ □□□□ . □ abcdefg □ abcdefh □ □□ □□□□ □□□□ . (????)

(□□□ □□ □□□□□ □ 6□□ □□□□ □□ . □ abcdef □□ □□ □ □□□ g □ h □□□ □□□ □□□□ □)

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

if □ else, int float □□ □□□□□□ □□□□□ □□□□ , □□□□□ □□□□□ □□□ .

□□□□ □□ □□□ □□□ □□□ □□□ □□□ □□□ □ □□ .

□□□□ , □□ □□□□□ □□ □□□ □□ , □□ □□□□ □ □□□ □□ □□□ □□ .

> □□

>

□□ □ □□

2.2 □□□ □□ □□

□□ □ □□

C□□ □□□ □□ □ □□ □□□□ □□□ □□ □□ .

char	한 바이트의 문자
int	정수, 기종에 따라 크기가 다르다.
float	단정도(single-precision) 부동소숫점
double	배정도(double-presision) 부동소숫점

□ □□□□ □ □□ □□□□ □□ □ □□□ □□□□ short□ long□ □ □ □□ .

short int sh;

long int counter;

□□ □□□□ int□ □□□ □□ □□ .

□□□

□□□ 1□□□□ □□□□□ □ 1 □□□□□ 256□□□ □□ □ □ □□ .

□□□ □□□ □□□ □□□□ □□□□ □□□ □□□ □□□□□ □□ □□□ □□
□□□□□□□□ ,

□□□ -128, □□□ 128 □ □□□□ □□□□ 0 □ □□□ . □□□ □□□ □□ 2□ □□
□□□□□ 0□ □□□ □□□□□□ □□□□ .

□□ □□ 128 □ 0 □ □□□□□ □□ □□□ □□ □ □□ □□ 0~127□ □□ .

□□□ □□□ -128 □□ , □ □ ,□□□ □□□ □□ □□□ .

1□ □□ □□□ □□ -0 □ +0□ □□□□ □□□ -127□□ □□□ 127□□ □□□ □□□ .

□□

□□ □ □□