



nano와 vim

로봇SW 교육원

최상훈(shchoi82@gmail.com)

학습 목표

2

- nano 편집기
- vim 편집기

nano

로봇SW 교육원


최상훈(shchoi82@gmail.com)

실습 1 : nano 파일 편집기

4

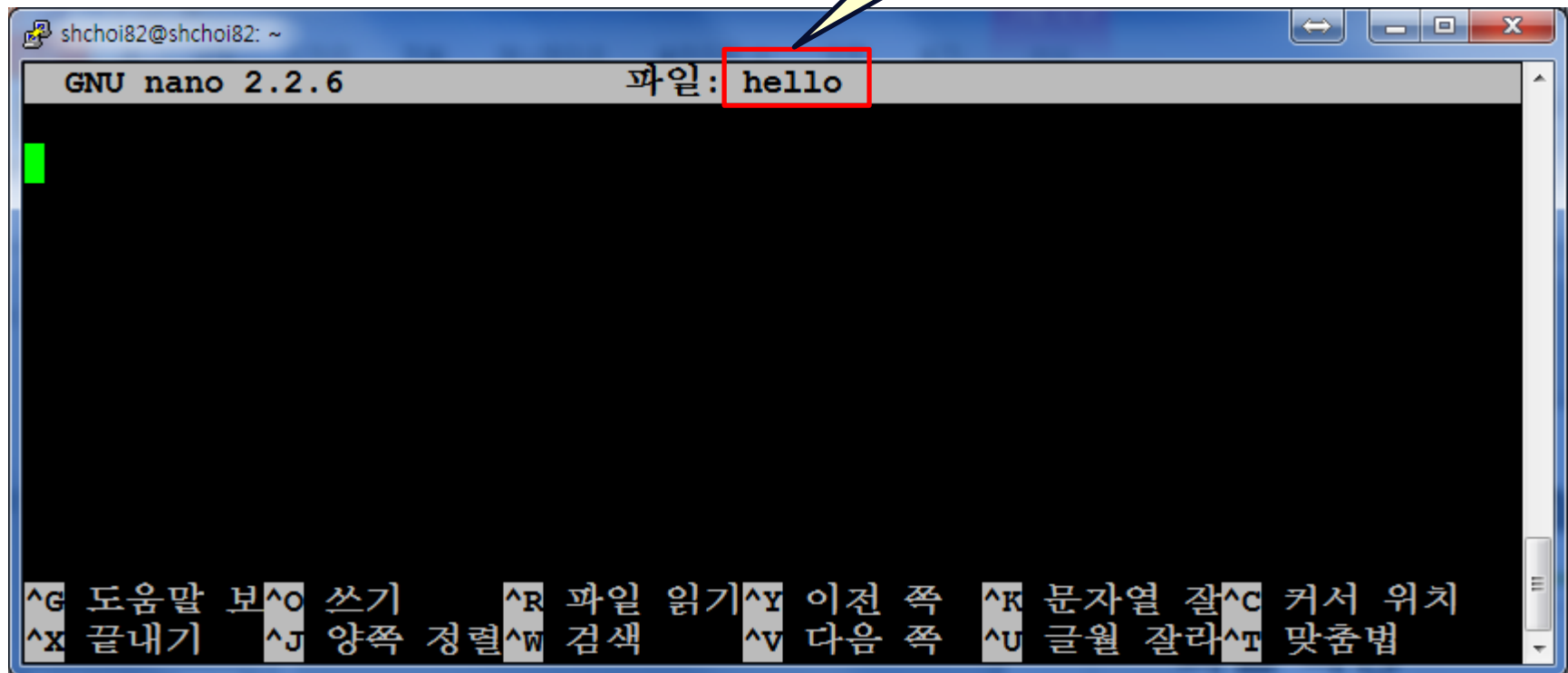
- 파일 만들기

- nano [파일명]

- ex) nano hello 

```
raspberrypi:~$ nano hello
```

파일명

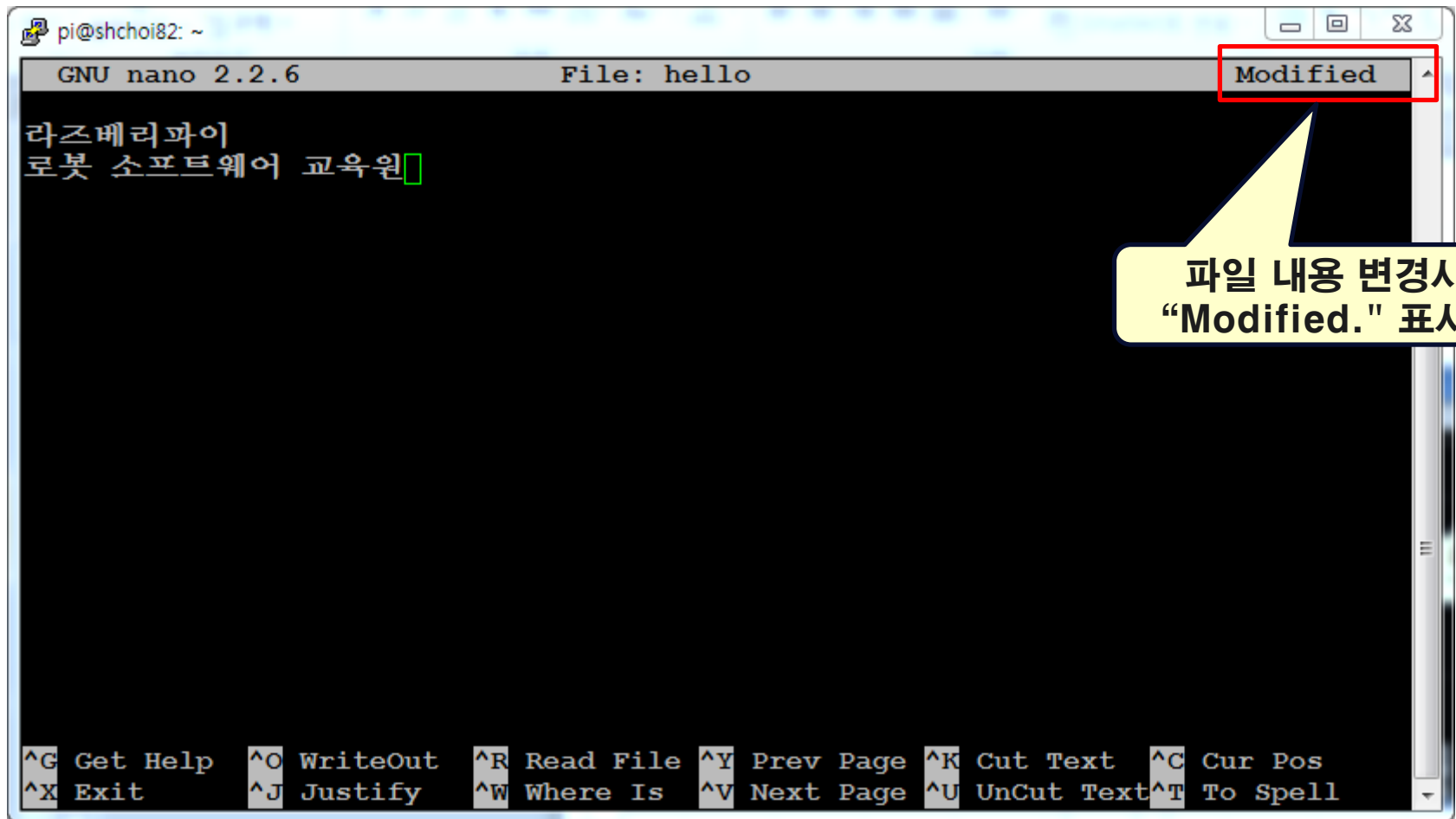


```
shchoi82@shchoi82: ~  
GNU nano 2.2.6      파일: hello  
  
^G 도움말 보기 ^O 쓰기 ^R 파일 읽기 ^Y 이전 쪽 ^K 문자열 잘라내기 ^C 커서 위치  
^X 끝내기 ^J 양쪽 정렬 ^W 검색 ^V 다음 쪽 ^U 글월 잘라내기 ^T 맞춤법
```

실습 2 : nano 파일 편집기

5

- 파일 내용 작성하기



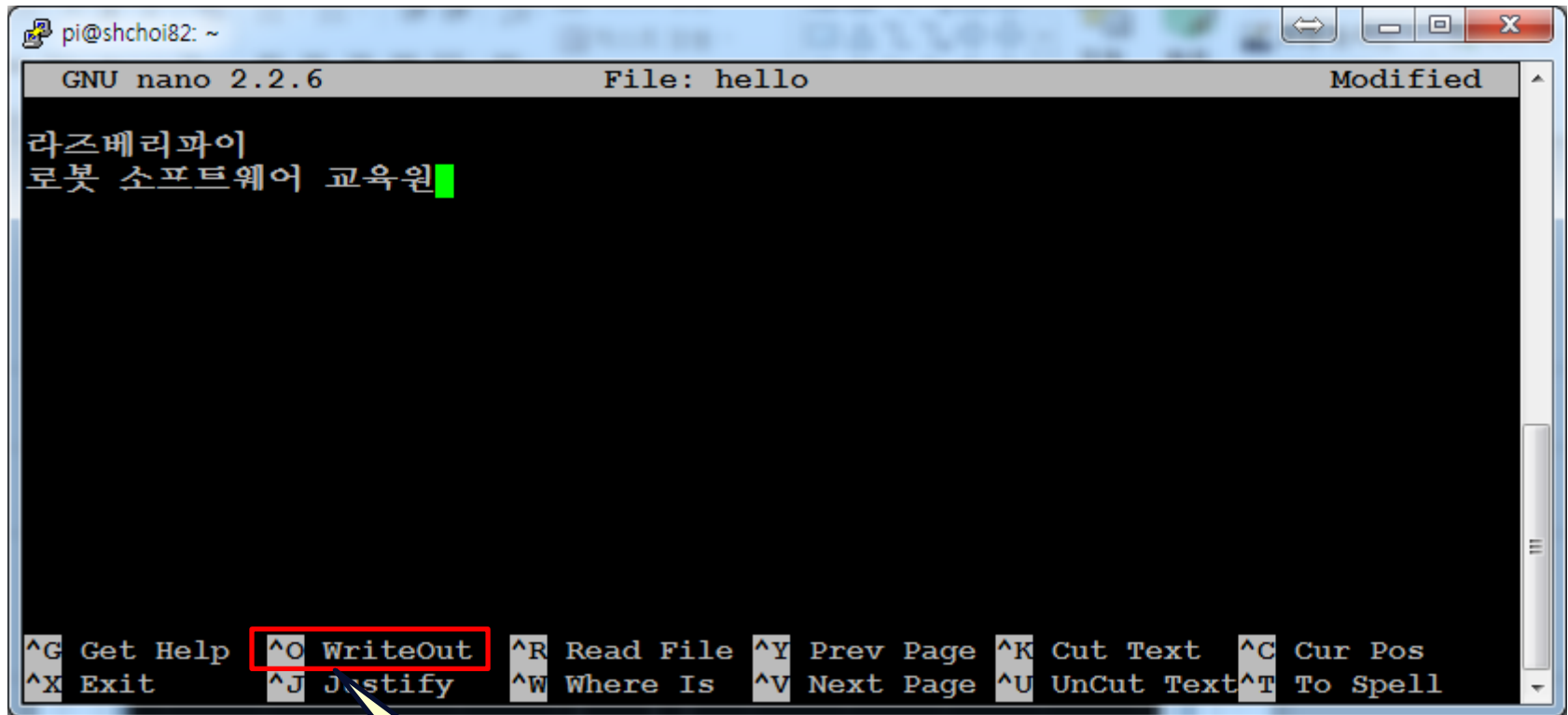
```
pi@shchoi82: ~  
GNU nano 2.2.6 File: hello Modified  
라즈베리파이  
로봇 소프트웨어 교육원  
  
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos  
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

실습 3 : nano 파일 편집기

6

- 저장하기

Ctrl + o



The screenshot shows the GNU nano 2.2.6 text editor interface. The title bar indicates the file is named 'hello'. The main editing area contains two lines of Korean text: '라즈베리파이' and '로봇 소프트웨어 교육원'. The bottom status bar displays various keyboard shortcuts. The shortcut '^O WriteOut' is highlighted with a red rectangular box. A yellow callout bubble with a black border points from this box to the text 'Ctrl + o' located below the editor window.

^G Get Help	^O WriteOut	^R Read File	^Y Prev Page	^K Cut Text	^C Cur Pos
^X Exit	^J Justify	^W Where Is	^V Next Page	^U UnCut Text	^T To Spell

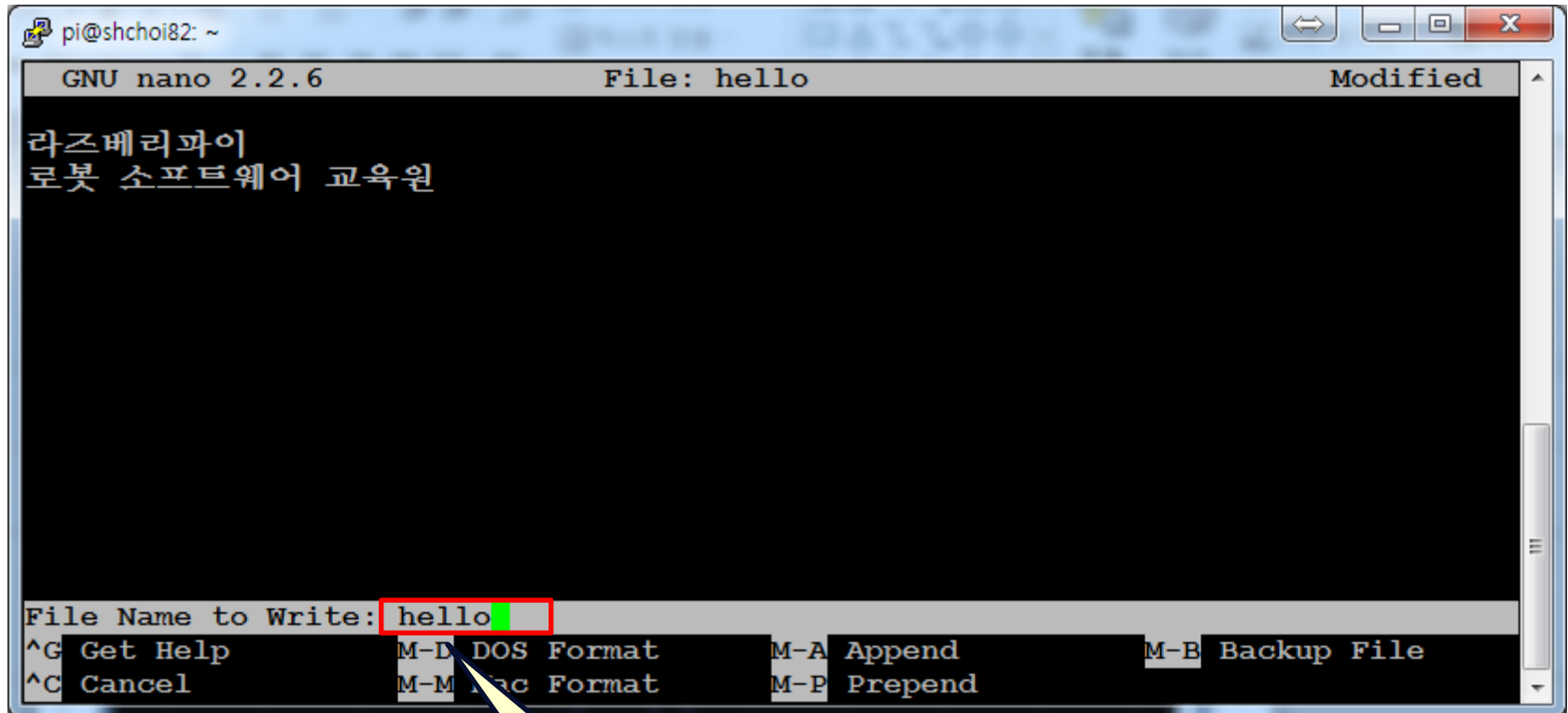
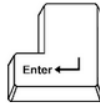
Ctrl + o

실습 4 : nano 파일 편집기

7

- 저장하기

파일명 확인 후 Enter



저장될 파일명

실습 5 : nano 파일 편집기

8

- 저장이 완료되면 오른쪽 위의 'Modified.' 표시가 사라짐
- 저장된 줄수를 표시해 줌

```
pi@shchoi82: ~  
GNU nano 2.2.6 File: hello  
라즈베리파이  
로봇 소프트웨어 교육원 █  
[ Wrote 2 lines ]  
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos  
^X Exit ^J Justify ^W Where Is ^N Next Page ^U UnCut Text ^T To Spell
```

“Modified.”
표시 사라짐

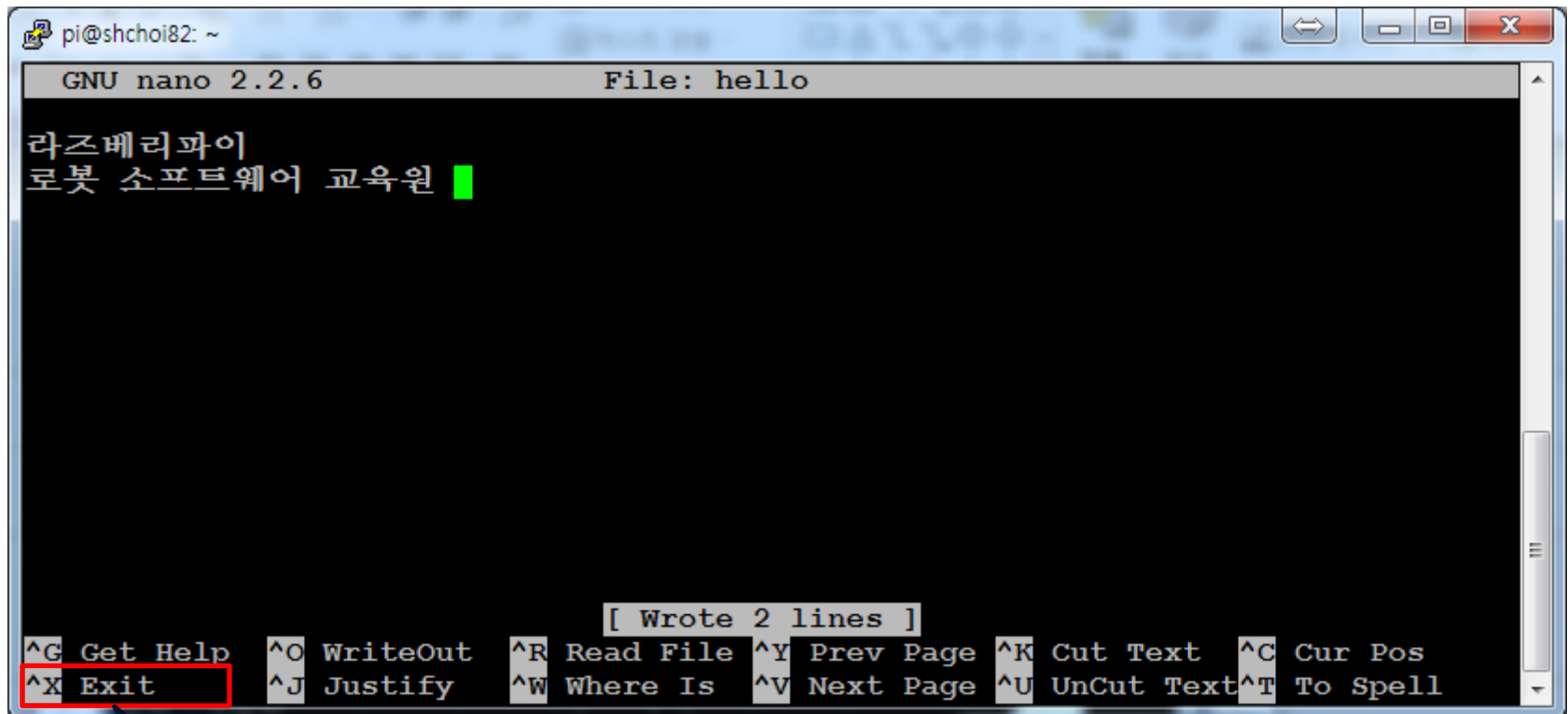
저장된 줄수를 표시해 줌

실습 6 : nano 파일 편집기

9

- 종료하기

Ctrl + x



The screenshot shows the GNU nano 2.2.6 text editor interface. The title bar indicates the file is named 'hello'. The editor contains two lines of Korean text: '라즈베리파이' and '로봇 소프트웨어 교육원'. A status bar at the bottom indicates '[Wrote 2 lines]'. Below the status bar, a list of keyboard shortcuts is displayed. The 'Exit' option, represented by '^X', is highlighted with a red rectangular box.

Shortcut	Action
^G	Get Help
^O	WriteOut
^R	Read File
^Y	Prev Page
^K	Cut Text
^C	Cur Pos
^X	Exit
^J	Justify
^W	Where Is
^V	Next Page
^U	UnCut Text
^T	To Spell

Ctrl + x

실습 7 : nano 파일 편집기

10

- nano 파일 편집기를 빠져나옴

```
raspberrypi:~# nano hello  
raspberrypi:~#
```

실습 8 : nano 파일 편집기

11

- 파일 목록 확인 하기
 - ls 명령어로 확인

```
raspberrypi:~$ ls  
hello  
raspberrypi:~$
```

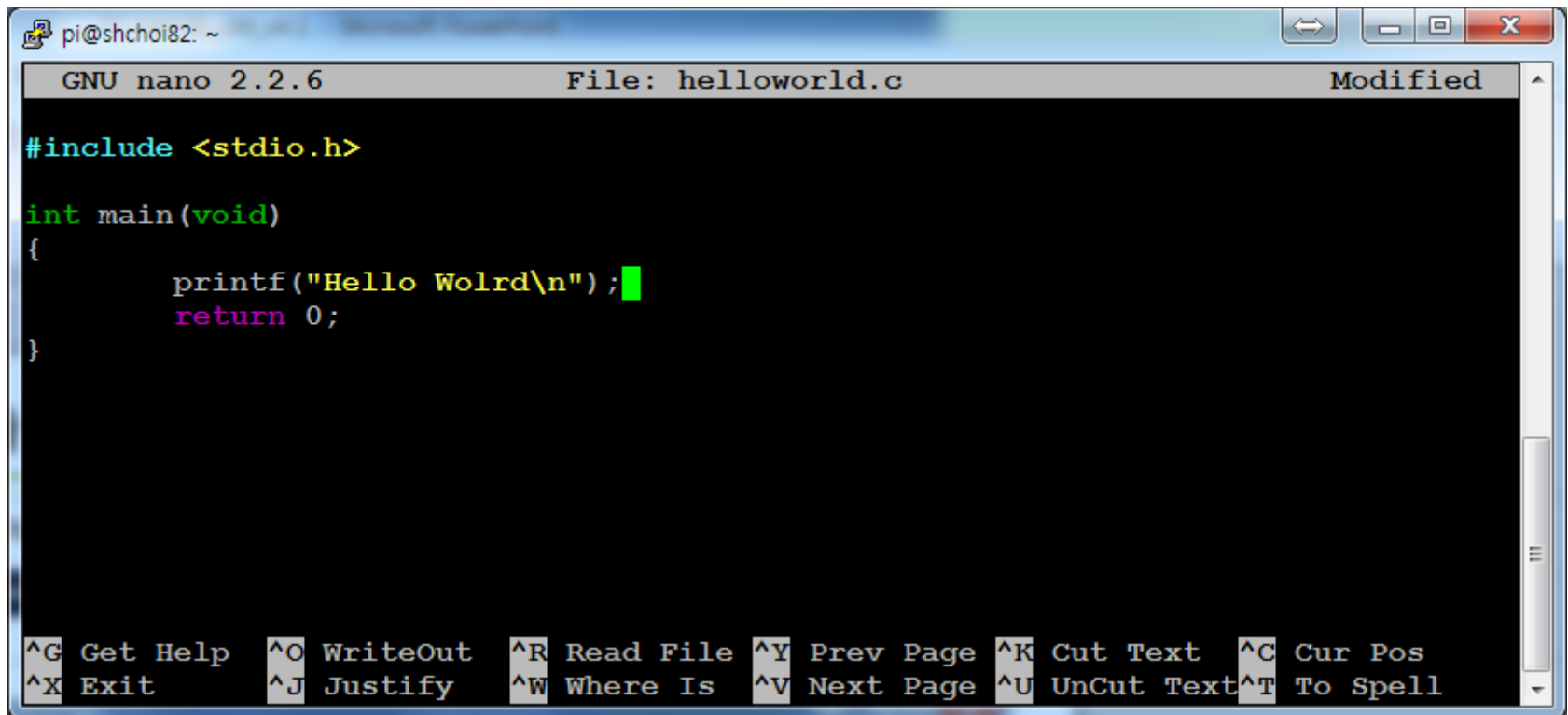
hello 파일이 생성됨

실습 9 : nano 파일 편집기

12

- nano 파일 편집기를 이용해 프로그램 소스 작성

```
raspberrypi:~# nano helloworld.c  
raspberrypi:~# nano -c helloworld.c
```



The screenshot shows a terminal window with the GNU nano 2.2.6 text editor open. The editor is editing a file named 'helloworld.c'. The code inside the file is as follows:

```
#include <stdio.h>  
  
int main(void)  
{  
    printf("Hello Wolrd\n");  
    return 0;  
}
```

The bottom of the window displays the nano editor's command shortcuts:

^G Get Help	^O WriteOut	^R Read File	^Y Prev Page	^K Cut Text	^C Cur Pos
^X Exit	^J Justify	^W Where Is	^V Next Page	^U UnCut Text	^T To Spell

실습 10 : nano 파일 편집기

13

- 파일 목록 확인 하기
 - ls 명령어로 확인

```
raspberry@pi:~$ ls  
hello helloworld.c  
raspberry@pi:~$
```

helloworld.c 파일이 생성됨

실습 11 : nano 파일 편집기

14

- Nano 편집기 단축키 기능

단축키	기능
Ctrl + O	저장하기
Ctrl + X	종료
Ctrl + R	파일 읽어오기
Ctrl + K	잘라내기
Ctrl + U	붙여넣기
Ctrl + ^ + 방향키	블럭지정
Ctrl + W	문자열 찾기

vim

로봇SW 교육원

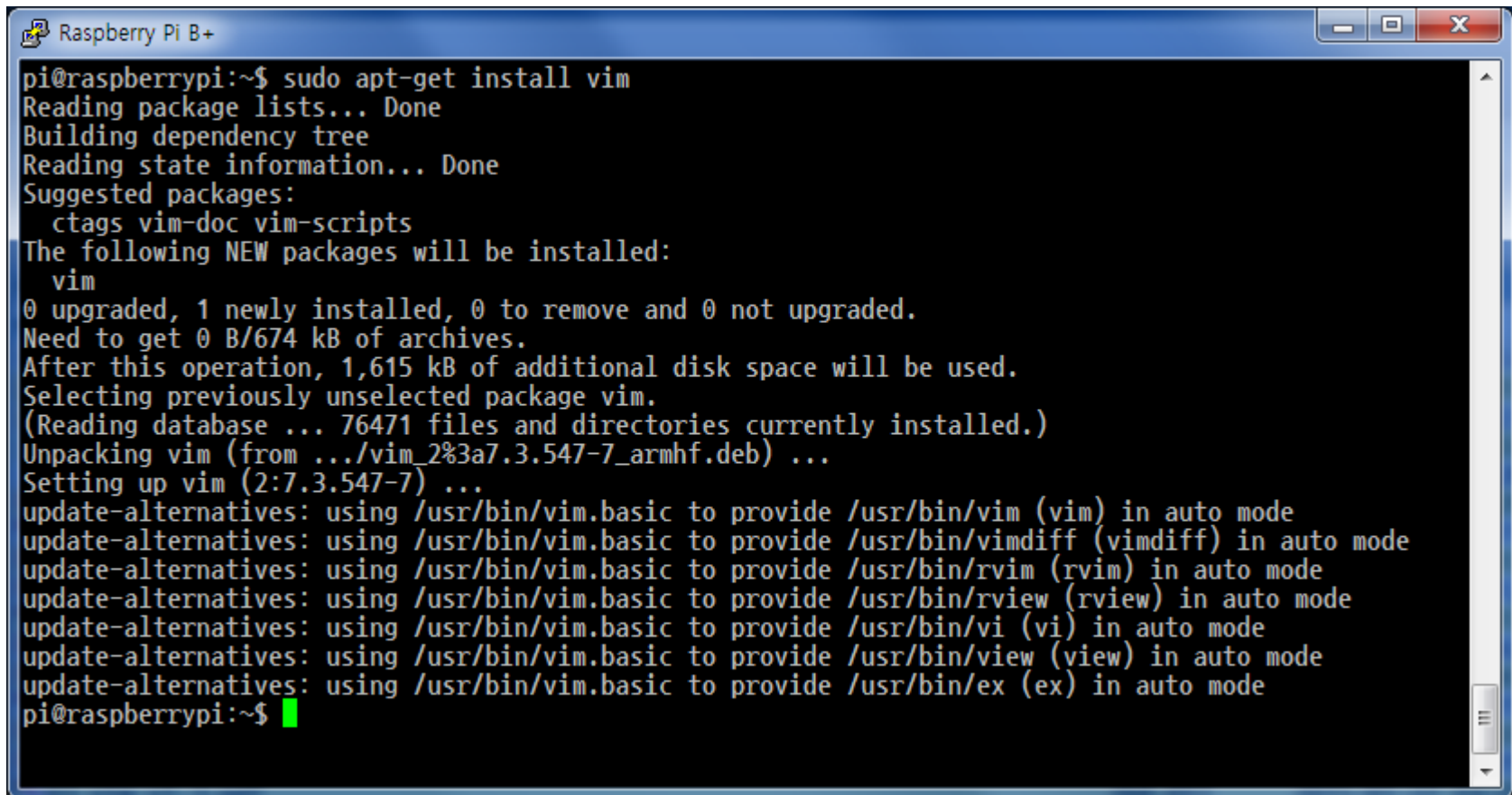
최상훈(shchoi82@gmail.com)

실습 2: vim 설치

16

- vim 설치

```
$ sudo apt-get install vim
```



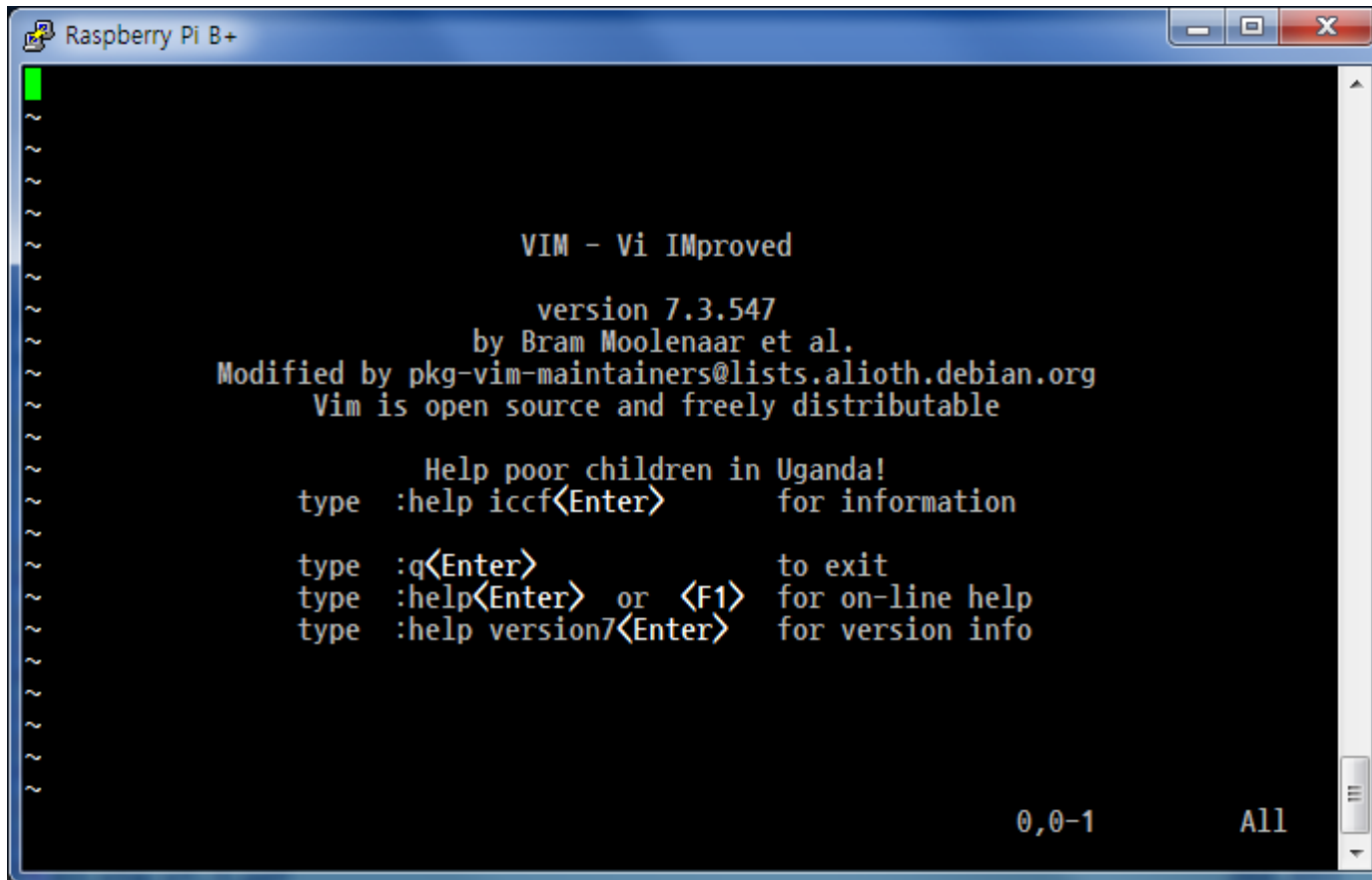
```
Raspberry Pi B+
pi@raspberrypi:~$ sudo apt-get install vim
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  ctags vim-doc vim-scripts
The following NEW packages will be installed:
  vim
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B/674 kB of archives.
After this operation, 1,615 kB of additional disk space will be used.
Selecting previously unselected package vim.
(Reading database ... 76471 files and directories currently installed.)
Unpacking vim (from .../vim_2%3a7.3.547-7_armhf.deb) ...
Setting up vim (2:7.3.547-7) ...
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/vim (vim) in auto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/vimdiff (vimdiff) in auto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/rvim (rvim) in auto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/rview (rview) in auto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/vi (vi) in auto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/view (view) in auto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/ex (ex) in auto mode
pi@raspberrypi:~$
```


실습 2: vim 실행

17

- vim 실행

```
$ vim
```



```
VIM - Vi IMproved
        version 7.3.547
        by Bram Moolenaar et al.
Modified by pkg-vim-maintainers@lists.alioth.debian.org
Vim is open source and freely distributable

        Help poor children in Uganda!
type  :help iccf<Enter>      for information

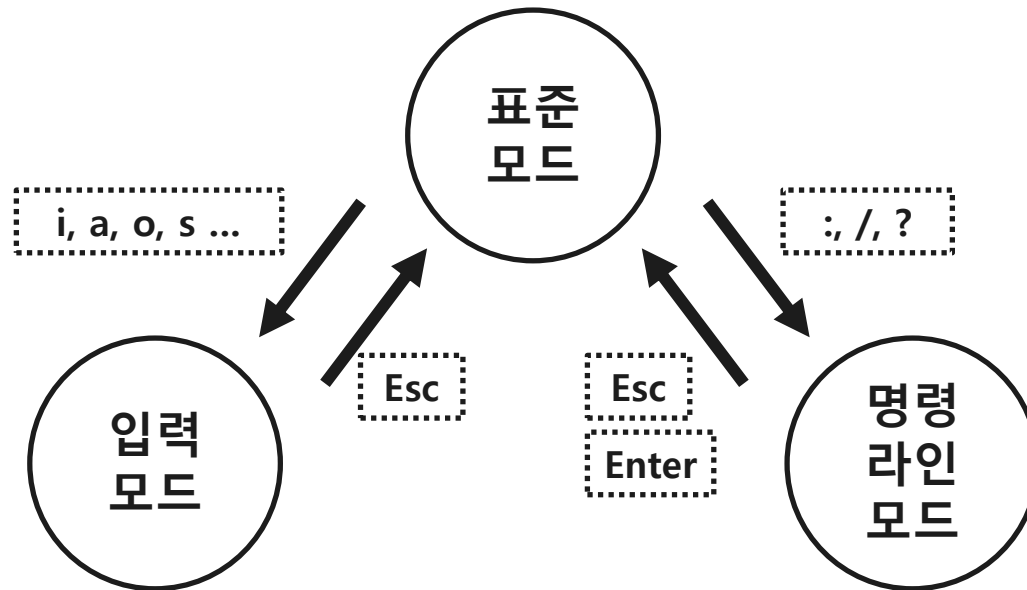
type  :q<Enter>              to exit
type  :help<Enter> or <F1>   for on-line help
type  :help version7<Enter> for version info

                                0,0-1      All
```

vim 편집기

18

- vim의 모드
 - 표준 모드
 - 입력 모드
 - 명령라인 모드
- vim 시작시 표준모드



vim 편집기

19

- 새 파일 작성
 - \$ vim 파일명

- 저장 / 종료

키	설 명
Esc : w Enter	저장
Esc : w 파일명 Enter	해당 파일로 저장
Esc : wq! Enter	저장 후 종료
Esc : q! Enter	저장하지 않고 종료

- 기존 파일 열기
 - \$ vim 파일명
 - vim 실행 후, 명령라인모드 ':' 전환 후 'e [filename]'

키	설 명
Esc : e 파일명 Enter	파일을 불러옴

vim 편집기

20

- 입력방법
 - 표준모드에서 입력모드 진입 후 입력

키	설 명
a	커서 위치 다음 칸부터 입력
i	커서의 위치에 입력
o	커서의 다음 행에 입력
s	커서 위치의 한 글자를 지우고 입력
A	커서 행의 맨 마지막부터 입력
I	커서 행의 맨 앞에서부터 입력
O	커서의 이전 행에 입력
cc	커서 위치의 한 행을 지우고 입력

vim 편집기

21

- 커서이동
 - 표준모드에서만 가능
 - 방향키로 이동 가능(또는 h, j, k, l)

키	설 명
w	다음 단어의 첫 글자로 이동
b	이전 단어의 첫 글자로 이동
gg	문서의 맨 첫 행으로 이동
G	문서의 맨 마지막 행으로 이동
: [n]	n번째 행으로 이동

vim 편집기

22

- 편집
 - 잘라내기

키	설 명
x	커서 위치의 한글자 잘라내기
dw	한 단어를 잘라내기
D	커서 위치부터 행의 끝까지 잘라내기
dd	커서가 있는 행 전체 잘라내기

- 복사 / 붙여넣기

키	설 명
yw	한단어를 복사
yy	한줄 복사
p	붙여넣기

vim 편집기

23

- 되돌리기 / 되살리기

키	설 명
u	되돌리기(명령취소)
Ctrl + r	되살리기

- 문자열 치환하기

키	설 명
Esc : %s/old/new/g Enter	문서 전체에서 old를 new로 교체
Esc : %s/old/new/gc Enter	문서 전체에서 old를 new로 확인하며 교체

실습 2 : vim 편집기 설정

24

- vim 설정파일
 - 설정파일 경로 : /home/pi/.vimrc

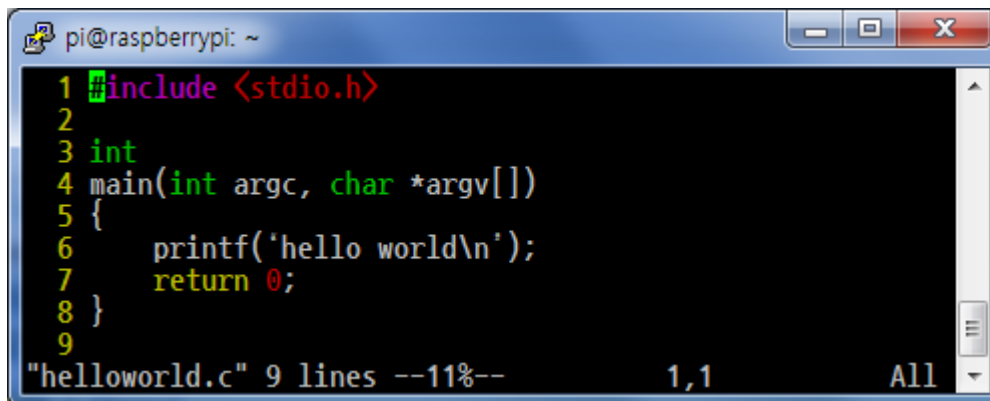
- 설정파일 작성

```
$ vim ~/.vimrc
```

```
set nu
syntax on
set tabstop=4
set shiftwidth=4
set softtabstop=4
```

- 설정파일 적용 확인

```
$ vim helloworld.c
```



```
pi@raspberrypi: ~
1 include <stdio.h>
2
3 int
4 main(int argc, char *argv[])
5 {
6     printf('hello world\n');
7     return 0;
8 }
9
"helloworld.c" 9 lines --11%-- 1,1 All
```


실습 2: vim 편집기

25

- 파일명 : ex1.c

```
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
Hello Wolrd
```

미션 1

26

- 다음 C 프로그램을 작성하시오.
 - 파일명 : helloworld.c

```
#include <stdio.h>

int
main(int argc, char *argv[])
{
    printf("hello world\n");
    return 0;
}
```

미션 2

27

- 다음 C 프로그램을 작성하시오.
 - 파일명 : stat.c

```
#include<stdio.h>
#include<sys/stat.h>

int main(int argc, char *argv[])
{
    struct stat buf;
    if(stat(argv[1], &buf) == -1){
        printf("stat error\n");
        return 1;
    }
    printf("file owner user   : %d\n", buf.st_uid);
    printf("file owner group  : %d\n", buf.st_gid);
    return 0;
}
```

미션 3

28

- 다음 C 프로그램을 작성하시오.
 - 파일명 : symbol.c

```
#include <stdio.h>

#define PI 3.14
#define NUM 100
#define BUFFER_SIZE 200

int main(int argc, char *argv[])
{
    printf("%lf \n", PI);
    printf("%d \n", NUM);
    printf("%d \n", BUFFER_SIZE);

    return 0;
}
```

- 위 소스파일의 백업용 파일을 만드시오.
 - 파일명 symbol_bak.c

미션 4

29

- 다음 C 프로그램을 작성하시오.
 - 파일명 : cla.c

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

int main(int argc, char *argv[])
{
    int i;

    for(i = 0 ; i < argc ; i++)
//    for(i = 0 ; argv[i] != NULL ; i++)
        printf("argv[%d]: %s\n", i, argv[i]);
    exit(0);
}
```

- 변수 i의 이름을 idx로 변경하시오.

미션 5

30

- 다음 프로그램 소스는 들여쓰기가 맞지 않다. 올바르게 수정하시오.
 - 파일명 : exam.c

```
#include<stdio.h>
int main(void)
{
int num;
printf("C : ");
scanf("%d", &num);
if (num>=95)
printf("A+\n");
else{
if (num>=90)
printf("A\n");
else{
if (num>=85)
printf("B+\n");
else{
if (num>=80)
printf("B\n");
else
printf("F\n");
}
}
}
return 0;
}
```

Esc

명령 모드

Vim 명령어 단축키

~ 대소문자 전환	! 외부 명령	@. 매크로 실행	# 이전 검색	\$ 행 끝으로 이동	% 짝 괄호 찾기	^ 행의 첫 글자	& :s 반복	* 다음 검색	(문장 시작) 행 앞으로	- 아래 행	+ 다음 행
,. 표식으로 이동	1	2	3	4	5	6	7	8	9	0 문장 끝	- 이전 행	= 자동 들여쓰기
Q 실행 모드	W 다음 단어 (의미상)	E 단어 (의미상) 끝으로	R 수정 모드	T. 행에서 후방 검색	Y 행 단위 복사	U 행 단위 실행 취소	I 행 시작에 삽입	O 행 위에 삽입	P 커서 이전에 붙여넣기	{ 문단 시작	}	문단 끝
q. 매크로 기록	w 다음 단어	e 단어 끝으로	r. 한 문자 교체	t. 행에서 전방 검색	y 복사 ¹ ³	u 실행 취소	i 커서 앞에 삽입	o 행 아래에 삽입	p 커서 이후에 붙여넣기 ¹	[]	
A 행 끝에 삽입	S 삭제 후 입력 모드	D 행 삭제	F. 행에서 후방 검색	G 파일 끝으로 이동	H 화면 상단	J 아래 행 합치기	K 도움말	L 화면 하단	: 명령행 모드	" 레지스터 ¹ 지정	열 이동	
a 커서 뒤에 삽입	s 삭제 후 입력 모드	d 삭제 ¹ ³	f. 행에서 전방 찾기	g. 확장 ⁶ 명령	h ←	j ↓	k ↑	l →	; /T/I/F 반복	' 표식으로 이동	\ 사용하지 않음	
Z. 종료 ⁴	X 백스 페이스	C 행 끝까지 바꾸기	V 비주얼 라인 모드	B 이전 단어 (의미상)	N 이전 (찾기)	M 화면 가운데	< 내어 쓰기 ³	> 들어 ³ 쓰기	? 찾기 (위로)			
z. 확장 ⁵ 명령	x 글자 삭제	c 바 ¹ ³ 꾸기	v 비주얼 모드	b 이전 단어	n 다음 (찾기)	m 표식 설정	, /T/I/F 반대 방향 검색	. 명령 반복	/ 찾기 (아래로)			

동작

커서를 이동하거나, 연산자가 동작할 범위를 지정합니다.

명령

바로 동작하는 명령입니다.
빨간색은 입력 모드로 변경됩니다.

오퍼레이션 팬딩 모드

커서 위치부터 목적지까지를 대상으로 명령을 실행합니다.

확장

추가적인 키 입력이 필요합니다.

q.

입력 후 글자를 입력해야 합니다.

단어

공백 문자나 특수 문자로 구분된 단어

test(123, 456, 789);

단어 (의미상)

공백 문자로 구분된 단어

test(123, 456, 789);

• 명령행 모드의 주요 명령어

:w 저장
:q 종료
:q! 저장없이 종료
:ef f 파일 열기
:%s/x/y/g 파일 전체에서 'x'를 'y'로 교체
:h name name 명령에 대한 도움말
:new 새 파일

• 일반 모드의 주요 명령어

CTRL-R 재실행
CTRL-F/B 페이지 위로/아래로
CTRL-E/Y 스크롤 위로/아래로
CTRL-V 비주얼 모드

• 참고

- 복사/붙여넣기/지우기 명령을 사용하기 전에 "a"를 입력하여 레지스터 a를 지정할 수 있습니다. (레지스터 이름은 a부터 z까지 사용 가능)
예를 들어 "ay\$는 커서 위치부터 행 끝까지의 내용을 레지스터 a에 저장합니다.
- 명령어 입력 전 숫자를 지정하면, 해당 숫자만큼 명령어가 반복됩니다.
- 연속으로 입력하면, 현재 행에 반영됩니다.
예를 들어 dd는 현재 행이 지워집니다.
- ZZ는 저장 후 종료, ZQ는 저장 없이 종료입니다.
- zt는 커서가 위치한 부분을 화면 상단으로 스크롤합니다.
zb는 바닥으로, zz는 가운데로 스크롤합니다.
- gg는 커서를 파일 처음으로 이동합니다.
gf는 커서 위치의 파일명을 인식하여 파일을 엽니다.