*linux 명령어

mkdir [디렉토리명] : 디렉토리명으로 디렉토리를 생성함

ls : 현재 디렉토리에 있는 것들을 출력

vi [파일명] : 파일명을 편집(메모장이라고 생각하면 쉽다 ?)

gcc [소스파일명] : 소스파일명을 컴파일해서 a.out 이라는 실행파일을 생성함
(소스 파일이란 반드시 *.c 확장자를 가지고 있어야한다)

gcc -o [실행파일명] [소스파일명] : 소스파일명을 컴파일해서 실행파일명의 실행파일을 생성함
(실행파일명은 소스파일명과 일치시키지 않아도 무방하지만
용도를 명확히 해줘야 이게 뭔지 파악을 하기에 좋다)

gcc -g -o [실행파일명] [소스파일명] : 위와 동일하게 컴파일을 하는데 디버깅 옵션을 추가적으로 주는 명령이다.
gcc -g [소스파일명] : 이와 같이 하여도 상관없지만 a.out 이라는 실행파일이 만들어짐

cp [복사하고자하는 파일] [원하는 이름] : 복사하고자하는 파일을 원하는 이름으로 복사함

pwd : 현재 디렉토리 위치를 확인하는 명령어

cd [디렉토리명] : change directory 의 약자로 디렉토리명으로 디렉토리 이동을하는 명령어

mv [이름을 바꾸고자 하는 파일] [원하는 이름] : 이름을 바꾸고자하는 파일을 원하는 이름으로 변경!
[이동시키고자 하는 파일] [디렉토리 이름] : 이동시키고자 하는 파일을 디렉토리 이름으로 이동시킴

rm -rf [삭제하고자하는 이름] : 삭제하고자 하는 이름을 디렉토리, 파일에 관계 없이 다 지움

gdb [실행파일 이름] : 실행파일을 디버깅 하고자 할 경우 사용한다.

* 컴파일 방법

소스 코드 작성후 컴파일 하고자 할 경우 gcc 를 사용한다.

gcc ~~~.c

실행 파일은 a.out 이라는 이름으로 생성된다.

./a.out

위와 같이 실행한다.

a.out 이라는 이름이 혼란을 유발할 수 있으므로 이름을 다르게 저장 할 수 있다.

gcc -o test ~~~.c

이렇게 하면 test 라는 이름으로 실행 파일이 생성된다.

./test

위와 같이 실행한다.

디버깅을 하고자 할 경우 -g 옵션을 추가한다.

gcc -g -o debug ~~~.c

역시 마찬가지로 실행은 ./debug 로 수행한다. 디버깅 옵션이 들어있으므로 gdb 를 사용할 수 있다.
디버거를 키고자 할 경우 아래와 같이 한다.

gdb debug

나가고자 할 경우엔 q 를 누르고 y 를 누른다.

w - 저장하기
:q - 나가기 gdb
:wq - 저장하고 나가기

yy : 1 줄 복사
y 숫자 y : 숫자만큼의 줄을 복사
p : 붙여넣기

dd : 1 줄 지우기
d 숫자 d : 숫자만큼의 줄을 지우기

:set nu - 줄 라인 보이기
:set ts=숫자 - 탭의 사이즈를 숫자만큼으로 지정함

v : 블록 지정(화살표 이동으로 블록지정 범위 선택이 가능함)
= : 정렬되지 않은 소스코드가 존재할 때 이를 깔끔하게 정렬해줌

추가적으로 c 언어 들여쓰기 방법을 알아보자.

터미널 상에서
vim~/.vimrc 를 치고 들어간다.

Set autoindent
set cindent
set ts=4
set shiftwidth=4

하고 저장하고 나오면 c 언어 들여쓰기가 가능하다.

c 언어 는 기본 문법 및 함수 사용법에 대하여 알아 보았다.

[반환형][함수이름][매개변수] 식으로 함수가 구성이 된다.
ex) int Add(int a, int b){
algorithm
return xxx;
}

메인함수에서 위 함수에 매개변수를 전달하여 해당 알고리즘을 간편하게 불러올 수 있다.