**C 프로그래밍2 중간시험 2020-05-04**

분반\_\_ 학과: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 학번: \_\_\_\_\_\_\_\_\_\_\_\_­­­\_\_\_\_ 이름: \_\_\_\_\_\_\_\_\_\_\_\_­­­\_\_\_\_

1. UNIX/LINUX의 파일 편집기 vi의 사용방법을 명령어모드, 삽입모드, 마지막행모드의 관점에서 설명하시오. (10) (이 자리에 답을 쓰시오)

명령어 모드(Command mode): vi를 시작하면 기본적으로 명령어 모드로 시작됩니다. 이 모드에서는 파일 편집과 관련된 다양한 명령어를 사용할 수 있습니다. 예를 들어, 다음과 같은 명령어를 사용할 수 있습니다.

i: 삽입 모드로 전환

x: 현재 커서 위치에 있는 문자 삭제

dd: 현재 줄 삭제

:wq: 파일 저장 후 종료

삽입 모드(Insert mode): 삽입 모드에서는 편집할 텍스트를 입력할 수 있습니다. 명령어 모드에서 i를 눌러 삽입 모드로 전환할 수 있습니다. 이 모드에서는 일반적인 편집기와 비슷하게 작동합니다. 삽입 모드에서는 다음과 같은 명령어를 사용할 수 있습니다.

ESC: 명령어 모드로 전환

CTRL + c: 명령어 모드로 전환

CTRL + d: 현재 줄 삭제

마지막 행 모드(Last line mode): 이 모드에서는 파일 저장, 검색, 대체 등과 같은 vi의 고급 기능을 사용할 수 있습니다. 명령어 모드에서 :을 입력하면 마지막 행 모드로 전환됩니다. 명령어 모드에서 다음과 같은 명령어를 사용할 수 있습니다.

:w: 파일 저장

:q: vi 종료

:wq: 파일 저장 후 vi 종료

:set number: 행 번호 표시

이상으로 vi 편집기의 명령어 모드, 삽입 모드, 마지막 행 모드에 대한 간단한 설명을 드렸습니다. 이러한 모드들은 vi 편집기에서 파일 편집을 훨씬 효율적이고 효과적으로 할 수 있도록 도와줍니다.

2. storage class와 관련된 C의 keyword인 auto, static, extern, register 가 각각어떤 역할을 하는지 설명하시오. (10) (이 자리에 답을 쓰시오)

auto: 이 storage class가 지정된 변수는 함수 내에서 선언된 경우 지역 변수(local variable)가 됩니다. 그리고 선언된 블록에서만 유효하며, 블록이 종료되면 소멸합니다. 이 storage class는 생략 가능합니다.

static: 이 storage class가 지정된 변수는 선언된 파일 내에서 전역 변수(global variable)가 됩니다. 그리고 초기화되지 않은 경우에는 0으로 초기화됩니다. 함수 내에서 선언된 경우에는 지역 static 변수가 됩니다. 이 변수는 블록이 종료되어도 소멸하지 않으며, 프로그램이 종료될 때까지 존재합니다.

extern: 이 storage class가 지정된 변수는 선언된 파일 외부에서 선언된 전역 변수를 사용할 때 사용됩니다. 이 변수는 다른 파일에서 선언되었지만, 현재 파일에서 사용됩니다.

register: 이 storage class가 지정된 변수는 CPU의 register에 저장됩니다. 그리고 변수의 주소를 사용하지 않으므로, 접근 속도가 매우 빨라집니다. 이 변수는 지역 변수(local variable)로만 선언될 수 있습니다. 그리고 이 storage class는 추천되지만, 컴파일러에서 register로 선언된 변수를 실제로 register에 저장하지 않을 수도 있습니다.

3. 다음 UNIX 명령어가 무엇을 하는 명령어인지 정확하게 쓰시오. (10) (이 자리에 답을 쓰시오)

가. ls :현재 디렉토리에 있는 파일과 하위 디렉토리 목록을 출력합니다.

나. mkdir:새 디렉토리를 생성합니다.

다. cd :현재 작업 중인 디렉토리를 변경합니다.

라. rm:파일이나 디렉토리를 삭제합니다.

마. pwd:현재 작업 중인 디렉토리의 경로를 출력합니다

4. 다음 두 선언은 어떻게 다른가 차이점을 설명하시오. (10) (이 자리에 답을 쓰시오)

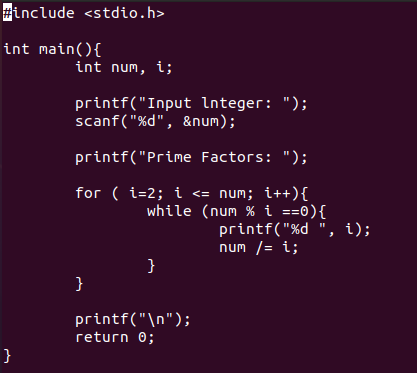
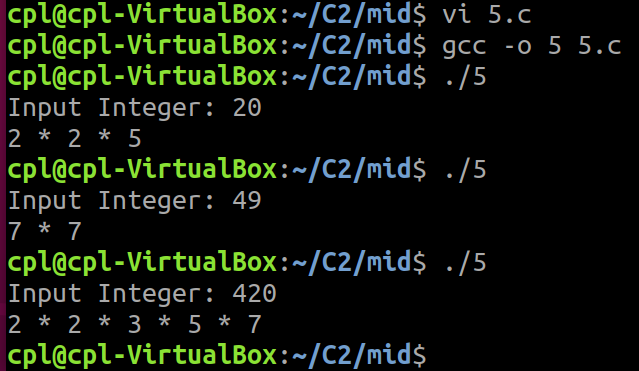


메모리 할당: amessage는 컴파일러가 메모리를 할당하고, 문자열의 크기에 따라 충분한 메모리가 할당됩니다. pmessage는 문자열 상수가 저장된 메모리 영역의 주소를 가리키는 포인터이며, 해당 문자열이 상수로 선언되었으므로 프로그램이 실행될 때 수정할 수 없습니다.

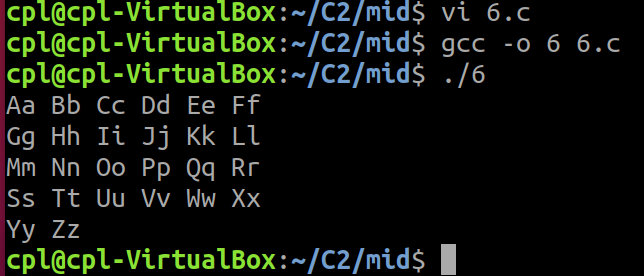
초기화: amessage는 문자열 리터럴로 초기화됩니다. 따라서 이후에 문자열을 수정할 수 있습니다. pmessage는 문자열 상수로 초기화됩니다. 따라서 이후에 문자열을 수정하려고 하면 프로그램이 예기치 않게 종료될 수 있습니다.

변경 가능성: amessage는 배열이므로 다른 문자열로 변경할 수 있습니다. pmessage는 문자열 상수를 가리키는 포인터이므로 포인터를 변경할 수는 있지만, 문자열 상수 자체를 변경할 수는 없습니다.

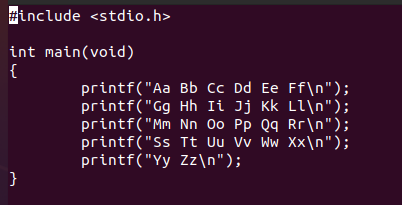
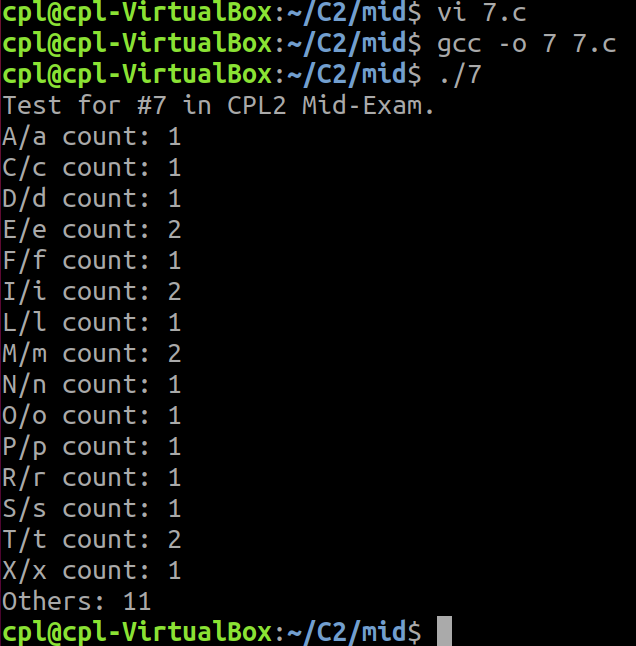
5. 아래와 같이 입력한 정수를 소인수 분해하여 결과를 내는 프로그램을 5.c로 만들고 이를 $gcc –o 5 5.c로 컴파일 하시오. $./5하여 수행하여 결과를 보이시오. (10) (모든 과정을 파악할 수 있도록 캡처하여 이곳에 붙이시오. 소스코드에는 comment를 해야 합니다. 5.c는 추후 압축파일에 추가합니다)



6. 다음과 같은 출력을 내는 프로그램을 6.c로 만들고 이를 $gcc –o 6 6.c로 컴파일 하시오. $./6하여 수행하여 결과를 보이시오. (10) (모든 과정을 파악할 수 있도록 캡처하여 이곳에 붙이시오. 소스코드에는 comment를 해야 합니다. 6.c는 추후 압축파일에 추가합니다)



7. 다음과 같이 문자열을 입력하고 ^D로 입력을 끝내면, 알파벳의 경우 대소문자를 합한 글자수를 출력하고, 그 외의 문자의 총합을 출력하는 프로그램을 7.c로 만들고 이를 $gcc –o 7 7.c로 컴파일 하시오. $./7하여 수행하여 결과를 보이시오. (10) (모든 과정을 파악할 수 있도록 캡처하여 이곳에 붙이시오. 소스코드에는 comment를 해야 합니다. 7.c는 추후 압축파일에 추가합니다)



8. 다음과 같은 이차원 배열이 있다.

int a[][5] = {{ 1, 2, 3, 4, 5 },

{ 6, 7, 8, 9, 10 },

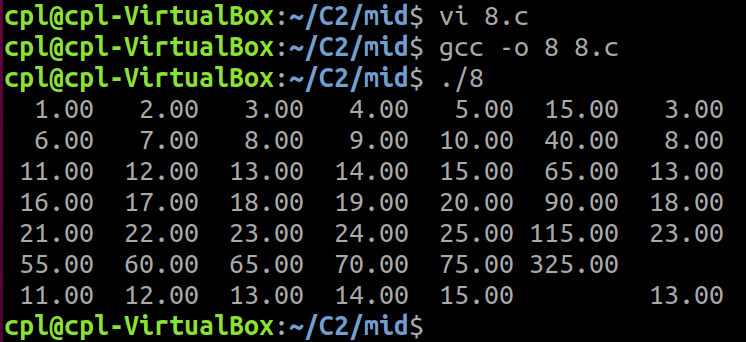
{ 11, 12, 13, 14, 15 },

{ 16, 17, 18, 19, 20 },

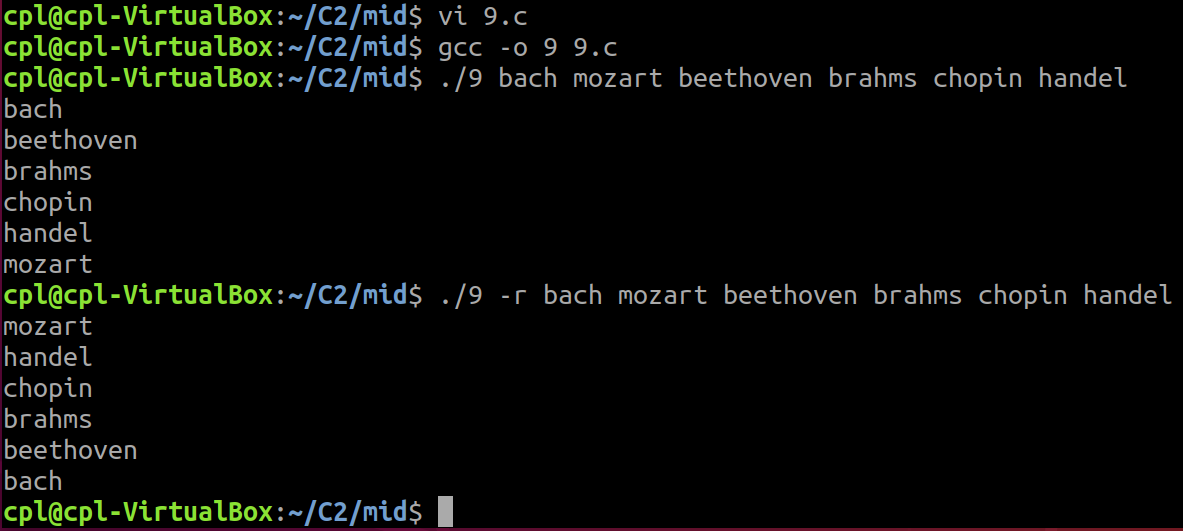
{ 21, 22, 23, 24, 25 }};

이 배열의 row의 합과 평균, column의 합과 평균, 전체 합과 평균을 아래 그림과 같이 출력해주는 프로그램을 8.c로 만들고 이를 $gcc –o 8 8.c로 컴파일 하시오. $./8하여 수행하여 결과를 보이시오. (10) (모든 과정을 파악할 수 있도록 캡처하여 이곳에 붙이시오. 소스코드에는 comment를 해야 합니다. 8.c는 추후 압축파일에 추가합니다)

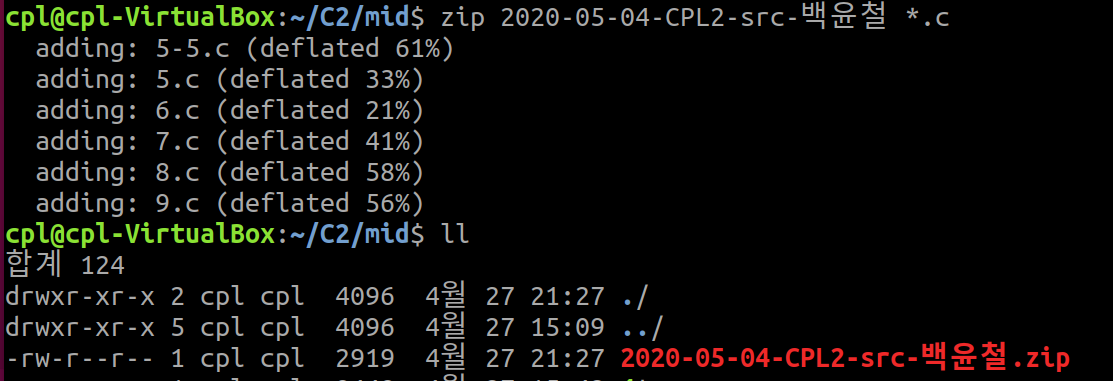
반복문 row와 하나씩



9. command-line argument로 공급되는 입력 문자열의 단어들을 사전순으로 정렬하여 출력하되 –r 옵션이 있으면 역순으로 출력하는 프로그램을 9.c로 만들고 이를 $gcc –o 9 9.c로 컴파일 하시오. 아래 그림과 같이 수행하여 결과를 보이시오. (10) (모든 과정을 파악할 수 있도록 캡처하여 이곳에 붙이시오. 소스코드에는 comment를 해야 합니다. 9.c는 추후 압축파일에 추가합니다)



(정리) 만들어진 소스파일을 아래와 같이 압축하고



(끝, 수고했습니다.)