Report

C프로그래밍(CSE2035-03) 프로젝트 1

미니 그림판



제출일 : 2017 . 11 . 07

과목명 : C프로그래밍(CSE2035-03)

담당교수 : 김세준

전공 : 컴퓨터공학 심화

학번 : 20171698

이름 : 이찬희

**1. 요구사항**

1.1 합성

- 포인터: 포인터는 메모리의 주소 값을 저장하기 위한 변수이다. 메모리에는 byte 단위로 주소가 지정되어 있으며 포인터 상수(pointer constants)는 이러한 메모리 주소(address)로 정의할 수 있다. C프로그램에서는 이 주소 값(address value)를 저장할 수 있을 뿐만 아니라 식별하여 사용할 수 있다. 주소 연산자(&)를 사용하면 해당 변수의 주소 값을 알 수 있다.[[1]](#footnote-1)

- 동적할당: 메모리의 낭비를 최소화하기 위해 프로그램의 실행 중에 입력되는 데이터에 맞게 기억공간이 할당되어야 한다. 따라서 메모리를 동적으로 할당하여 사용한다.[[2]](#footnote-2) 동적 메모리 할당 또는 메모리 동적 할당은 컴퓨터 프로그래밍에서 실행 시간 동안 사용할 메모리 공간을 할당하는 것을 말한다. 사용이 끝나면 운영체제가 쓸 수 있도록 반납하고 다음에 요구가 오면 재 할당을 받을 수 있다.[[3]](#footnote-3) 메모리의 동적 할당과 해제 기능은 라이브러리로 구현되어 있고, 이를 이용하기 위해서는 반드시 포인터를 이용한다.

1.2 분석

1.2.1 main

|  |  |
| --- | --- |
|  | int main (); |
| 기능요약 | 메인 함수에서는 testcase 값을 받고 run\_function을 받은 testcase의 값만큼 for문을 사용하여 호출한다. 이 때 testcase에 들어오는 값을 정수로 한정하기 위하여 scanf의 반환값이 1 이 나닌 경우 에러 메시지를 출력하고 프로그램을 중단한다. |

1.2.2 run\_function

|  |  |
| --- | --- |
|  | int run\_function (); |
| 기능요약 | run\_function 함수는 본 프로그램의 큰 틀 역할을 하는 함수로, 이미지를 읽어오고 구동 할 쿼리의 개수를 입력받은 후 run\_query를 저장한 개수만큼 호출한다 |
| 설명 | 우선 이미지를 저장할 char\*\* 형 변수 image를 선언한다. 그리고, 이미지의 높이와 너비를 저장할 int 형 변수 height와 width, 실행할 query의 개수를 저장할 int 형 변수 testcase\_query를 저장한다.  read\_image를 height, width, image를 이용하여 호출하여 이미지를 입력 받는다. 이 때 read\_image가 1을 반환 할 경우 에러가 발생한 것이므로 더 이상 진행하지 않도록 1을 반환하여 함수 실행을 중지한다.  testcase\_query에 실행할 query의 개수를 입력받고, 입력받는 값을 정수로 한정하기 위하여scanf가 1을 반환하지 않을 경우 getchar를 ‘\n’를 만날 때 까지 실행시켜 buffer를 비운 후 1을 반환하여 함수 실행을 중지한다. |

1.2.3 read\_image

|  |  |
| --- | --- |
|  | int read\_image (int \*height, int \*width, char\*\*\* image\_adress); |
| 기능요약 | read\_image에서는 image를 선언하고 이를 make\_board를 호출하여 동적으로 할당한다. 이후, 문자를 입력받고 get\_character를 호출하여 image를 채워 이를 \*image\_adress에 저장한다. |
| 파라미터 | height는 이미지의 높이 값을 저장하는 변수의 주소, width는 이미지의 너비를 저장하는 변수의 주소이고 image\_adress는 이미지를 저장하는 배열의 주소이다. |
| 설명 | 우선 이미지를 저장할 char\*\* 형 변수 image 와 get\_character에서 반환 받은 에러타입을 저장할 int 형 변수 error\_type 을 선언한다.  그리고 check\_size를 호출하여 입력 받기로 한 이미지가 너무 크지 않은 지 검사한다. 이때, check\_size의 반환값이 1이라면 이미지가 너무 크다는 에러 메세지를 출력하고 1를 반환하며 함수 실행을 종료한다.  이미지의 사이즈가 너무 크지 않다는 것을 확인하면 make\_board를 호출하여 image를 동적으로 할당 한다. 이후 get\_character를 호출하여 이미지를 입력받고, 반환값을 error\_type에 저장한다. Error\_type이 1이거나 2라면, 상황에 맞는 에러 메세지를 출력하고 1을 반환하여 함수 실행을 종료한다.  에러 없이 입력을 받았다면, 이를 전달받은 \*image\_adress에 저장한다. |

1.2.4 check\_size

|  |  |
| --- | --- |
|  | int check\_size (int height, int width); |
| 기능요약 | 캔버스(이미지가 저장될 배열)의 크기를 height, width를 통해 확인하고 이미지가 너무 클 경우, 즉 height 나 width가 4096을 초과할 경우 에러를 반환한다. |
| 파라미터 | Height는 image의 높이를, width는 image의 너비를 전달받는다. |

1.2.5 make\_board

|  |  |
| --- | --- |
|  | char\*\* make\_board (char\*\* image, int height, int width); |
| 기능요약 | height과 width를 이용하여 image를 동적으로 할당한다. |
| 파라미터 | Image는 이미지를 저장할 이차배열을, height와 width는 그의 높이와 너비를 전달받는다. |

1.2.6 get\_character

|  |  |
| --- | --- |
|  | int get\_character (char\*\*\* image, int height, int width); |
| 기능요약 | 문자를 입력 받고 이를 image에 저장한다. |
| 파라미터 | Image는 이미지를 저장하는 이차배열의 주소값을, height와 width는 그 이차배열의 높이와 너비를 전달받는다. |
| 설명 | 우선, 입력 받을 문자들을 저장하기 위한 char형 변수 temp를 선언한다. 또, image값을 저장하는 char\*\* 형 변수 temp\_image를 선언한다.  이미지를 입력 받기 전에 get\_char()를 이용하여 버퍼를 비우고, temp에 문자를 하나씩 입력 받는다. 이 때 ‘\n’이 입력될 때의 카운트와 width의 값이 같아야 정상적인 입력으로 처리하고, 나머지는 상황에 맞는 에러를 반환하며 함수 실행을 종료한다.  정상적인 입력이 들어왔다고 판단될 경우 temp\_image에 이를 저장한다. |

1.2.7 run\_query

|  |  |
| --- | --- |
|  | int run\_query (int \*height, int \*width, char\*\*\* image); |
| 기능요약 | 어떤 query를 실행할 것인지 입력 받고, 그를 실행 할 함수를 호출한다. 입력값이 1 ~ 5가 아닌 경우 에러메세지를 출력하고 1을 반환한다. |
| 파라미터 | Image, height, width는 각각 이미지를 저장하는 이차배열과 그 높이, 너비를 저장하는 변수의 주소를 전달받는다. |

1.2.8 run\_query\_1

|  |  |
| --- | --- |
|  | int run\_query\_1 (int \*height, int\* width, char\*\*\* image); |
| 기능요약 | 옵션을 입력 받고, resize를 호출한다. 결과를 출력한다. |
| 파라미터 | image는 이미지를 저장하는 이차배열의 주소값을, height와 width는 그 이차배열의 높이와 너비를 전달받는다. |
| 설명 | 옵션을 입력받을 int 형 변수 option과 \*height 값을 저장할 int형 변수 temp를 선언한다.  Option에 입력 받은 옵션의 값이 실행 가능한 값이 아닐 경우 에러를 출력하고, 이미지를 출력한 후 1을 반환하여 함수 실행을 종료한다.  정상적인 값을 입력받은 경우 temp에 \*height값을 저장하고 resize를 호출하여 수정된 이미지를 \*image에 저장한다. 만약, resize를 호출 한 이후의 height와 temp가 서로 같을 경우 에러가 발생하여 수정되지 않은 것이므로 이미지를 출력하고 1을 반환하여 함수 실행을 종료한다.  만약 정상적으로 수정된 경우, 바뀐 \*height와 \*width값을 출력한 후 이미지를 출력한다. |

1.2.9 run\_query\_2

|  |  |
| --- | --- |
|  | int run\_query\_2 (int \*height, int\* width, char\*\*\* image); |
| 기능요약 | 옵션을 입력 받는다. 이 때 옵션의 값이 실행 가능한 값이 아니면 에러를 출력하며 함수 실행을 종료하고, 그렇지 않으면 rotate함수를 호출하여 이미지를 수정한다. 결과를 출력한다. |
| 파라미터 | image는 이미지를 저장하는 이차배열의 주소값을, height와 width는 그 이차배열의 높이와 너비를 전달받는다. |

1.2.10 run\_query\_3

|  |  |
| --- | --- |
|  | int run\_query\_3 (int \*height, int\* width, char\*\*\* image); |
| 기능요약 | 옵션을 입력 받는다. 이 때 옵션의 값이 실행 가능한 값이 아니면 에러를 출력하며 함수 실행을 종료하고, 그렇지 않으면 flip함수를 호출하여 이미지를 수정한다. 결과를 출력한다. |
| 파라미터 | image는 이미지를 저장하는 이차배열의 주소값을, height와 width는 그 이차배열의 높이와 너비를 전달받는다. |

1.2.11 run\_query\_4

|  |  |
| --- | --- |
|  | int run\_query\_4 (int \*height, int\* width, char\*\*\* image); |
| 기능요약 | 좌표와 복사할 영역의 크기를 입력 받는다. 이때 좌표와 크기가 정해진 범위를 벗어나면 에러를 출력하며 함수 실행을 종료하고, 그렇지 않으면 copy와 paste를 호출한다. 결과를 출력한다. |
| 파라미터 | image는 이미지를 저장하는 이차배열의 주소값을, height와 width는 그 이차배열의 높이와 너비를 전달받는다. |

1.2.12 run\_query\_5

|  |  |
| --- | --- |
|  | int run\_query\_5 (int \*height, int\* width, char\*\*\* image); |
| 기능요약 | 좌표와 색상을 입력 받는다. 이 때 좌표가 이미지의 범위를 벗어나거나, 색이 원래 해당좌표의 색과 같다면 에러를 출력하며 함수실행을 종료하고, 그렇지 않으면 fill을 호출한다. 결과를 출력한다. |
| 파라미터 | image는 이미지를 저장하는 이차배열의 주소값을, height와 width는 그 이차배열의 높이와 너비를 전달받는다. |

1.2.13 print\_image

|  |  |
| --- | --- |
|  | void print\_image (int height, int width, char\*\* image); |
| 기능요약 | 넘겨받은 이미지를 출력한다. |
| 파라미터 | image는 출력할 이미지를, height와 width는 그 이미지의 높이와 너비를 저장한다. |

1.2.14 swap

|  |  |
| --- | --- |
|  | void swap (char\* ch1, char\* ch2); |
| 기능요약 | 넘겨받은 문자 두 개를 서로 바꿔준다. |

1.2.15 free\_canvas

|  |  |
| --- | --- |
|  | void free\_canvas (int height, char\*\* image); |
| 기능요약 | imgae를 free해준다 |
| 파라미터 | image는 free할 이차배열을, height는 그 이미지의 높이를 저장한다. |

1.2.16 realloc\_canvas

|  |  |
| --- | --- |
|  | char\*\* realloc\_canvas (int height, int width, char\*\* image); |
| 기능요약 | image의 크기를 realloc을 이용하여 바꿔준다. |
| 파라미터 | image는 realloc할 이차배열의 주소값을, height와 width는 그 이차배열의 바뀐 후 높이와 너비를 전달받는다. |
| 설명 | Width를 prev\_height에 저장하고, height를 prev\_width에 저장한다. 이후 min\_height에 prev\_height와 height중 더 작은 값을 저장한다. 만약 height가 prev\_height보다 작으면, image[height]부터 image[prev\_height - 1]까지 free해주고 image[0]부터 image[min\_height]까지 realloc한다. 그리고 min\_height부터 height까지 image[i]를 malloc한다.  Image를 반환한다. |

1.2.17 resize

|  |  |
| --- | --- |
|  | char\*\* resize (int arg1, int\* height, int\* width, char\*\* image); |
| 기능요약 | arg1에 저장된 옵션을 사용하여, 이미지의 크기를 바꾼다. |
| 파라미터 | image는 free할 이차배열을, height는 그 이미지의 높이를 저장한다. Arg1은 실행할 옵션을 전달받는다. |
| 설명 | 수정 후의 이미지를 저장해 주기 위해 char\*\* result\_image를 선언한다.  만약 arg1이 0 이면, \*height와 \*width의 두 배로 check\_size를 호출하여 만약 그 값이 1이면 에러를 출력하고 함수 실행을 종료한다. 그렇지 않으면 make\_board를 사용하여 높이, 너비가 현재 이미지의 두 배인 이차배열을 만들어준다. 이후에, for 문을 사용하여 result\_image에 이미지의 값을 넣어준다. 하나의 픽셀이 네 개로 resize되므로, image[i][j]의 값이 result\_image의 [2\*i][2\*j], [2\*I + 1][2\*j], [2\*i][2\*j + 1], [2\*I + 1][2\*j + 1]에 들어가면 된다. 그리고 free\_canvas를 호출하여image를 free해준 후 \*height 와 \*width의 값을 수정해주고 result\_image를 반환한다.  만약 arg1 이 1이면, \*height 또는 \*width 가 1 이 아닌지 검사하고, 맞다면 에러를 출력하고 함수 실행을 종료한다. 그렇지 않으면 make\_board를 사용하여 높이 , 너비가 현재 이미지의 반인 이차배열을 만들어주고, for문을 사용하여 result\_image에 값을 넣어준다. 네 개 픽셀의 평균값이 하나의 픽셀로 들어가므로, image의 [2\*i][2\*j], [2\*I + 1][2\*j], [2\*i][2\*j + 1],[2\*I + 1][2\*j + 1]의 평균이 result\_image[i][j]에 들어가면 된다. 그리고 free\_canvas를 호출하여 image를 free 해준 후 \*height와 \*width의 값을 수정해주고 result\_image를 반환한다. |

1.2.18 rotate

|  |  |
| --- | --- |
|  | char\*\* rotate (int\* height, int\* width, int angle, char\*\* image); |
| 기능요약 | angle에 저장된 옵션을 이용하여 이미지를 회전시킨다. |
| 파라미터 | image는 free할 이차배열을, height는 그 이미지의 높이를 저장한다. angle은 실행할 옵션을 전달받는다. |
| 설명 | 수정 후의 이미지를 저장해 주기 위해 char\*\* result\_image를 선언한다. 그리고 make\_board를 호출하여 result\_image를 동적으로 할당한다.  for문을 angle 번 돌리고, for문 내에서는 주어진 이미지를 90도 돌린다. 이 때 우선 result\_image의 크기를 realloc\_canvas를 사용하여 바꿔준 후 image의 값들을 result\_image에 넣어준다. 이후free\_canvas를 호출하여 image를 free 해주고 height 와 width의 값을 서로 바꿔준다. result\_image를 image에 대입해 준 후 만약 angle != count이면 result\_image를 다시 동적으로 할당 해 준다.  Image를 반환한다. |

1.2.19 flip

|  |  |
| --- | --- |
|  | void flip (int height, int width, int flag, char\*\*\* image); |
| 기능요약 | flag에 저장된 옵션을 이용하여 이미지를 반전시킨다. |
| 파라미터 | image는 free할 이차배열을, height는 그 이미지의 높이를 저장한다. Flag는 실행할 옵션을 전달받는다. |
| 설명 | 만약 flag가 0이면, swap 함수를 호출하여 이미지를 위아래로 뒤집는다.  만약 flag가 1이면, swap함수를 호출하여 이미지를 좌우로 뒤집는다. |

1.2.20 copy

|  |  |
| --- | --- |
|  | char\* copy (int x1, int y1, int c\_h, int c\_w, int height, int width, char\*\* image); |
| 기능요약 | 저장된 좌표와 크기를 이용하여 이미지에서 복사할 영역을 temp에 저장한다. |
| 파라미터 | image는 free할 이차배열을, height는 그 이미지의 높이를 저장한다. X1과 y1은 복사될 영역의 가장 왼쪽 위의 좌표를, c\_h, c\_W에서는 복사할 영역의 크기를 전달받는다. |
| 설명 | int 형 변수 x\_end와 y\_end를 선언하여 복사하는 영역의 가장 오른쪽 아래 좌표를 저장한다. 이후에 (x\_end-x1+2) \* (y\_end-y1+1)의 크기로 temp를 동적할당해주고, 복사할 영역의 문자들을 넣어준다. 이 때 한 행이 들어갈 때 마다 ‘\0’을 넣어 다음행과 구분해준다. 마지막에 ‘\0’를 한 번 더 넣어 마지막임을 표시한다.  Temp를 반환한다. |

1.2.21 paste

|  |  |
| --- | --- |
|  | void paste (int x2, int y2, int height, int width, char\*\* image, char\* temp); |
| 기능요약 | 저장된 좌표와 temp를 이용하여 복사된 영역을 이미지에 붙여넣는다. |
| 파라미터 | image는 free할 이차배열을, height는 그 이미지의 높이를 저장한다. X2와 y2는 붙여넣기 할 영역의 가장 왼쪽 위 좌표를, temp는 복사한 영역의 값들을 전달받는다. |
| 설명 | For문을 사용하여 붙여 넣을 영역에 temp의 값을 차례대로 넣어준다. 벽을 만나면 다음행으로 넘어가고, ‘\0’을 만나도 다음행으로 넘어간다.  Temp를 free해준다. |

1.2.22 fill

|  |  |
| --- | --- |
|  | char\*\* fill (int x, int y, char color, int height, int width, char\*\* image); |
| 기능요약 | 저장된 좌표와 색을 이용하여 인접한 pixel을 탐색하고 같은 색일 경우 저장된 색으로 채워준다. |
| 파라미터 | image는 free할 이차배열을, height는 그 이미지의 높이를 저장한다. X와 y는 탐색을 시작할 좌표를, color는 바뀔 색을 전달받는다. |
| 설명 | Queue라는 구조체 배열을 생성하여 이곳에 좌표를 저장한다. 초기좌표에서 상, 하, 좌, 우를 탐색하여 초기좌표에 저장된 문자와 같은 문자를 가진다면 expand\_queue를 호출하여 queue를 한 칸 늘리고 그 자리에 해당 좌표를 넣어준다. 그리고 탐색을 완료하면 탐색을 시작한 좌표의 색상을 저장된 색상으로 바꿔준다. 더 이상 탐색할 수 없을 때 탐색을 완료하며 바뀐 pixel의 수, 즉 queue의 사이즈를 출력해주고 queue를 free해 준 후 image를 반환한다. |

1.2.23 expand\_queue

|  |  |
| --- | --- |
|  | struct coordinate\* expand\_queue (int \*size, struct coordinate \*queue); |
| 기능요약 | queue를 한 칸 늘리고 size값을 수정한다. |
| 파라미터 | queue는 수정할 queue의 주소를, size는 그 queue의 길이를 의미한다. |

1.3 제작

1.3.1 main스크린샷이(가) 표시된 사진

높은 신뢰도로 생성된 설명

1.3.2 run\_function

스크린샷, 텍스트이(가) 표시된 사진

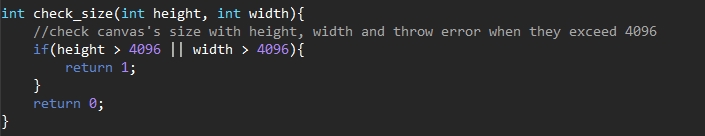
높은 신뢰도로 생성된 설명

1.3.3 read\_image

스크린샷, 텍스트이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.4 check\_size



1.3.5 make\_board

스크린샷이(가) 표시된 사진

높은 신뢰도로 생성된 설명

1.3.6 get\_character

스크린샷, 텍스트이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.7 run\_query

스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.8 run\_query\_1

스크린샷, 텍스트이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.9 run\_query\_2

스크린샷, 텍스트이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.10 run\_query\_3

스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.11 run\_query\_4

스크린샷, 텍스트이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.12 run\_query\_5

스크린샷, 텍스트이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.13 print\_image

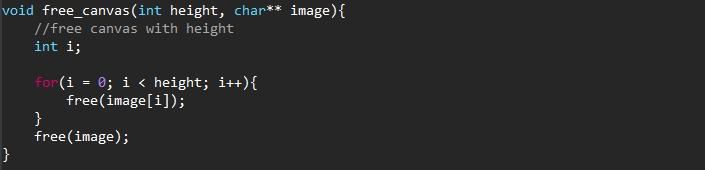
스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.14 swap



1.3.15 free\_canvas



1.3.16 realloc\_canvas

스크린샷, 텍스트이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.17 resize

텍스트, 스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

스크린샷, 텍스트이(가) 표시된 사진

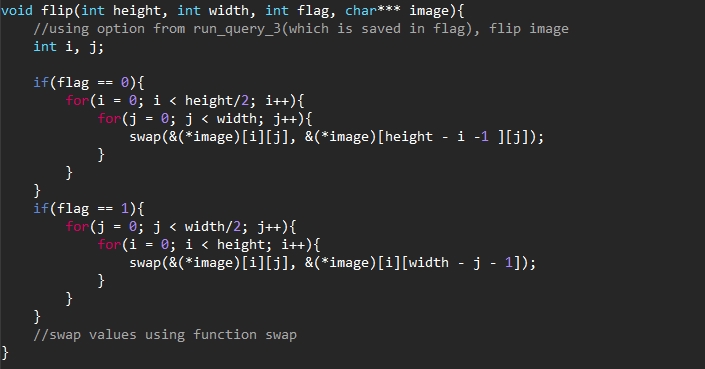
높은 신뢰도로 생성된 설명

1.3.18 rotate

텍스트이(가) 표시된 사진

높은 신뢰도로 생성된 설명

1.3.19 flip



1.3.20 copy

스크린샷, 텍스트이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.21 paste

스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.22 fill

텍스트이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1.3.22 expand\_queue

스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

2.4 시험

스크린샷이(가) 표시된 사진

높은 신뢰도로 생성된 설명

스크린샷이(가) 표시된 사진

높은 신뢰도로 생성된 설명

프로젝트 ppt의 설계화면과 같은 값을 입력하여 그 결과를 본 것이다. 두번째 testcase는 생략하였다.

2.5 평가

코드를 짤 때 최대한 중복되는 코드를 작성하지 않도록 중복되는 기능은 함수로 분리하였다. 이 덕분에 효율적인 코드를 작성할 수 있었던 것 같다. 프로그램에 아쉬운 점은 fill 함수에서 탐색을 할 때 이미 방문한 좌표를 제외하지 않아 size가 불필요하게 늘어나고, 수정한 pixel수가 의도한 대로 출력되지 않는 점이다.

2.6 보건 및 안정

전역변수를 사용하지 않았으면 동적으로 할당한 배열은 사용을 마치면 free해 주어 메모리를 관리하였다. Realloc 할 때도 free와 malloc을 적절히 섞어 사용하여 메모리를 관리 하였다.

2.7 생산성과 내구성

코드를 짤 때 최대한 중복되는 코드를 작성하지 않도록 중복되는 기능은 함수로 분리하여 효율적인 코드를 작성하고자 노력하였다.

2.8 산업표준

ansi표준은 준수하기 위해 컴파일 할 때 gcc –ansi명령어를 사용하였다.

**3. 기타**

3.1 환경구성

리눅스 서버에서 코드를 작성하고 실행해보았다.

3.2 참고사항

1. C programming 강의자료, [01] Pointer\_1 [↑](#footnote-ref-1)
2. C programming 강의자료, [06] Pointer Applications\_2-2 [↑](#footnote-ref-2)
3. 동적 메모리 할당 - 위키백과, 우리 모두의 백과사전 [↑](#footnote-ref-3)