2021년 1학기

**프로그래밍과 문제해결**

Assignment #3

담당 교수 : 윤은영

학번 : 20210084

학과 : 무은재학부

이름 : 김지민

POVIS ID : kjm1672

================명예서약(Honor code)===============

“나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.”

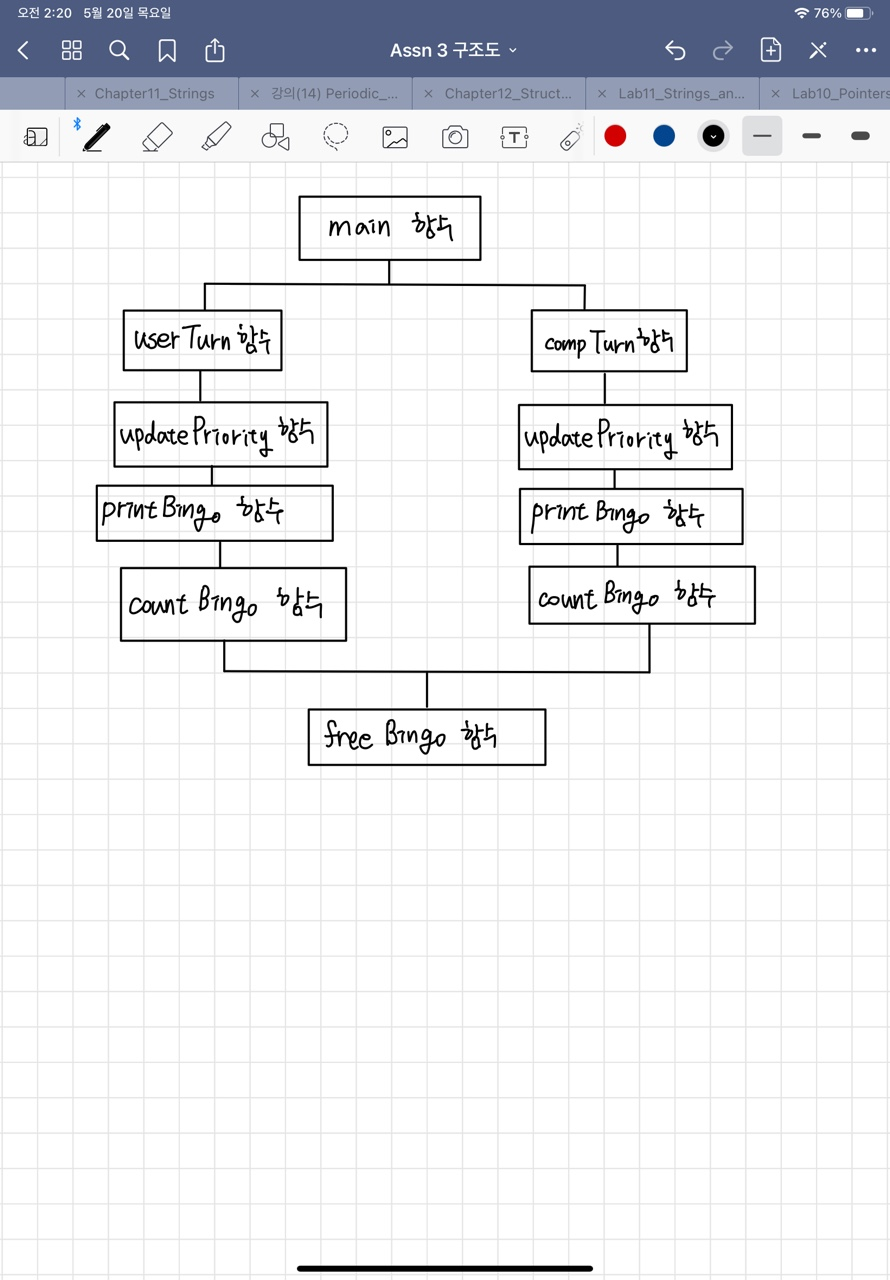
==============================================

1. 문제의 개요

본 프로그램을 간략히 소개하면 다음과 같다.

* N \* N 크기의 빙고 판을 구현하여 플레이어와 컴퓨터가 대결한다.
* 처음 화면에서 사용자로부터 메뉴 선택을 입력 받는다. 1: 시작 / 2: 종료
* 빙고 보드는 파일로부터 단어를 읽어서 구성한다.
* 빙고 보드 크기와 승리 빙고 개수는 사용자로부터 입력 받는다.
* 사용자부터 시작하여 차례대로 빙고를 진행한다.
* 승리 빙고 개수를 먼저 채우는 쪽이 승리한다. 빙고가 완료되면 다시 처음 메뉴 화면으로 돌아간다.

이때 구조 차트 (structure chart)는 아래와 같이 표현될 수 있다.

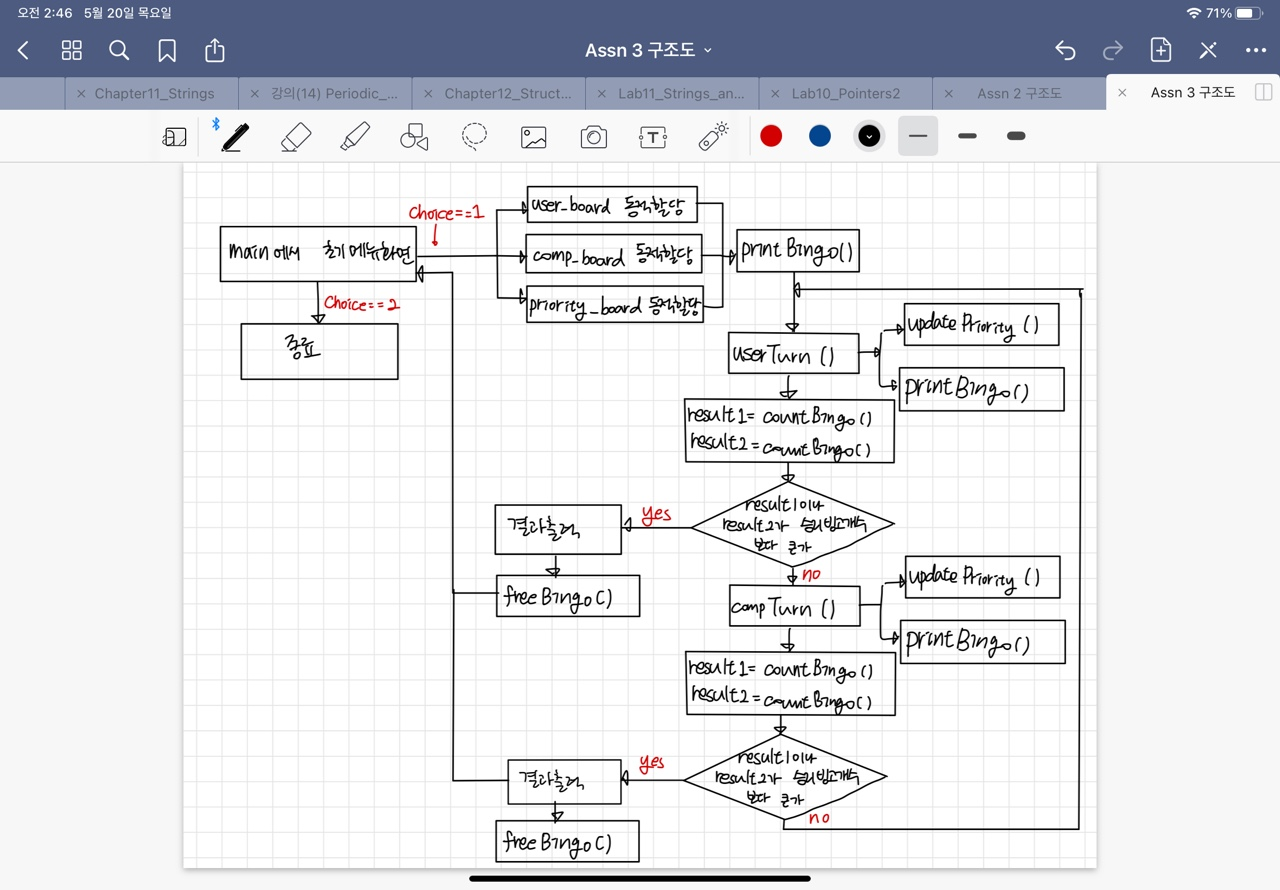


1. 알고리즘

본 프로그램 작성을 위한 알고리즘을 Presudo 코드 형태로 나타내면 다음과 같다.

|  |
| --- |
| Presudo-algorithm |
| 1. 헤더파일 <stdio.h>, <stdlib.h>, <time.h>, <string.h>, <math.h>를 포함시킵니다. 2. 사용자로부터 메뉴를 선택 받습니다.   - 1: 빙고 게임 시작, 2: 프로그램 종료  - 1, 2 외 입력이 들어올 경우, 올바른 입력이 들어올 때까지 반복하여 입력 받습니다.  3. 사용자로부터 파일 이름을 입력 받고 해당 파일 안 단어들을 출력합니다. 이때 char pointer를 원소로 가지는 2차원 배열을 동적 할당하고 파일 단어를 모두 저장합니다.  - 파일이 없는 경우 처음 메뉴 화면으로 돌아갑니다.  4. 사용자로부터 빙고 보드의 크기를 입력 받습니다. 범위는 2부터, (N은 단어 개수)보다 작거나 같은 정수까지로 제한합니다.  - 입력 값이 범위를 벗어난 경우, 처음 메뉴 화면으로 돌아갑니다.  5. 사용자로부터 승리 빙고 개수를 입력 받습니다. 최소 1개에서 최대 2\*(빙고 판 크기)+2개까지 입력 받습니다.  - 범위를 벗어난 경우, 처음 메뉴 화면으로 돌아갑니다.  6. 보드 크기가 매번 달라지기 때문에 사용자 보드와 컴퓨터 보드는 각각 동적 할당 받아 생성합니다. 이때 한 칸당 단어 하나가 배정되어야 하므로 문자열을 저장할 char pointer를 원소로 가지는 2차원 배열을 동적 할당 합니다.  7. 사용자로부터 보드 생성 방식을 입력 받습니다. (1: Shuffle / 0: No)  -1(Shuffle): 빙고 보드 크기만큼 단어를 무작위로 추출하여 보드를 채웁니다. 이때 각 단어마다 필요한 길이만 동적 할당 받아 저장합니다. 아까 임의로 단어를 저장한 배열의 인덱스를 중복없이 무작위로 뽑아 각 보드 배열에 저장합니다.  -0(No): 사용자의 보드 배열은 첫 번째 단어부터 순서대로 저장합니다. 컴퓨터의 보드 배열에는 파일의 역순으로 저장합니다. 아까 임의로 단어를 저장한 배열을 복사하여 저장시킵니다.  - 각 보드 배열에 단어를 저장하였으면 임시 단어 저장 배열은 동적 해제 시킵니다.  8. 플레이어와 컴퓨터의 보드에 모두 단어가 출력되도록 합니다. 반복문을 이용하여 보드를 출력합니다.  9. 사용자부터 단어를 선택합니다. 존재하지 않는 단어를 입력한 경우, 다시 입력 받습니다. 이때, 반복문을 이용하여 사용자, 컴퓨터 보드에 남아있는 단어를 모두 검사합니다. 겹치는 원소가 존재한다면 “#########”로 바꾸고 출력합니다.  10. 컴퓨터는 우선 순위 보드 규칙에 따라 우선 순위가 높은 단어를 선택하도록 합니다. 우선 순위 보드 또한 매번 크기가 달라질 수 있기 때문에 동적 할당 받습니다. 매 턴마다 우선 순위 정도가 달라지기 때문에 updatePriorty 함수를 선언하여 업데이트하도록 합니다. 이때, 반복문을 이용하여 사용자, 컴퓨터 보드에 남아있는 단어를 모두 검사합니다. 겹치는 원소가 존재한다면 “#########”로 바꾸고 출력합니다.  11. countBingo함수를 매 턴마다 호출하여 각 보드의 빙고 개수를 셉니다. 이때, 둘 중 하나라도 승리 빙고 개수와 같거나 크게 되면 게임의 결과를 출력하고 다시 처음 메뉴 화면을 출력합니다. 이를 위하여 while문을 전체에 걸어줍니다. |

위 알고리즘을 flowchart 로 표현하면 아래와 같다.



1. 프로그램 구조 및 설명
2. 초기 화면 출력.

* main 함수 : 메뉴를 출력하고 1을 입력하면 시작, 2를 입력하면 프로그램을 종료한다. do while문을 이용하여 1, 2 이외의 숫자를 입력했을 경우 에러 메시지를 출력, 다시 입력 받는다. if문을 이용하여 2일 경우, 프로그램의 전체 흐름을 맡는 while문 밖을 탈출하도록 한다.

1. 빙고 보드 설정

* main 함수

1. 사용자로부터 파일명을 입력 받는다. if문과 continue를 이용하여 파일이 존재하지 않을 경우 전체 while문의 처음으로 돌아가도록 한다.
2. p라는, char pointer를 원소로 가지는 1차원 배열을 선언하고 파일 속 단어를 모두 저장한다. 그와 동시에 화면에 단어를 모두 출력한다. 이때, p[i]는 (단어 길이 + 1)만큼 메모리를 동적 할당 한다.
3. 빙고 보드의 크기를 사용자로부터 입력 받아 size라는 변수에 저장한다. (1)과 동일한 방식으로 에러 처리를 한다. (처음 화면으로 되돌아간다.) 승리 빙고 개수는 bingo\_num에 입력 받는다. (에러 처리는 동일하다.)
4. char pointer를 원소로 가지는 2차원 배열인 char\*\*\* user\_board, char\*\*\*comp\_board를 선언하여 문자열을 원소로 저장할 수 있도록 합니다. 이때, size 값에 맞게 동적 할당한다.
5. 사용자로부터 보드 생성 방식을 입력 받는다. (1은 섞기, 0은 순서대로)

* 1 입력 시: 임시로 파일 단어를 모두 저장한 p 배열의 인덱스는 (전체 단어 크기-1)까지 있다. 이 인덱스 중 무작위로 2개를 뽑아 해당 원소를 사용자와 컴퓨터 보드 배열에 각각 저장한다. 무작위로 뽑은 인덱스를 저장하기 위해 selected1(사용자 보드 배열에 저장하기 위해 뽑은 인덱스 저장), selected2(컴퓨터 보드 배열에 저장하기 위해 뽑은 인덱스 저장)라는 1차원 배열을 선언한다. 사용자와 컴퓨터 보드 배열에 단어를 저장할 때에는 calloc함수를 이용하여 (해당 단어 + 1)만큼만 동적 할당 받는다. 즉, 아래와 같이 코드를 작성할 수 있다.

k = 0;

for (i = 0; i < size; i++)

for (j = 0; j < size; j++)

{

do

{

agn = 0;

index1 = rand() % total\_word;

index2 = rand() % total\_word;

for (n = 0; n < k; n++)

if (selected1[n] == index1 || selected2[n] == index2)

agn = 1;

} while (agn);

selected1[k] = index1; selected2[k] = index2; k++;

user\_board[i][j] = (char\*)calloc((strlen(p[index1]) + 1), sizeof(char));

comp\_board[i][j] = (char\*)calloc((strlen(p[index2]) + 1), sizeof(char));

strcpy(user\_board[i][j], p[index1]);

strcpy(comp\_board[i][j], p[index2]);

}

* 0 입력 시: 사용자 보드 배열엔 첫 번째 단어부터 순서대로 대입하고, 컴퓨터 보드 배열엔 파일 단어 역순으로 대입한다. 이중 for문을 이용하여 p배열에 있는 단어를 순서대로 입력하도록 한다.
* 사용자, 컴퓨터 보드 배열에 단어를 모두 저장한 다음, p배열을 동적 할당 해제한다.

1. size, user\_board, comp\_board, output(출력 파일의 주소를 저장할 파일 포인터 변수)를 매개변수로 하는 printBingo 함수를 호출하여 빙고 판을 출력한다.

* void printBingo 함수

1. 이중 for문을 이용하여 사용자와 컴퓨터 모두 문자열이 보이도록 출력한다.
2. 왼쪽 정렬하기 위해, “%-9s”로 출력한다.
3. 파일 출력도 같이 한다.
4. 빙고 진행

* void userTurn

1. 사용자로부터 단어를 selection 변수에 입력 받는다.
2. 이중 for문과 if문을 이용하여 사용자 보드 배열 원소 중 selection과 일치하는 것이 있는지 검사한다. 이때, strcmp함수로 두 배열을 비교함으로써 검사했다.
3. strcpy함수를 이용하여 사용자, 컴퓨터 보드 배열 원소 중 일치하는 것을 찾고, 해당 원소 문자열을 “#########”로 바꾼다. 이때, realloc함수를 이용하여 해당 원소 메모리 크기를 10(문자열의 길이 + 1)으로 할당한다.
4. 컴퓨터 보드 배열 중 일치하는 것이 있을 경우, updatePriority 함수를 호출하여 우선순위 보드를 업데이트한다.
5. printBingo함수를 호출하여 빙고 판의 현황을 출력한다.

* void compTurn

1. 우선 순위 보드를 매개변수로 받아, 이중 for문과 if문을 이용하여 가장 우선순위가 높은 원소의 인덱스를 maxindex1, maxindex2에 저장한다.
2. 해당 인덱스를 가진 우선순위 보드 원소를 0으로 바꾼다.
3. 사용자 턴에서와 마찬가지로 realloc, strcmp, strcpy 함수를 이용하여 일치하는 원소의 값을 바꾼다.
4. updatePriority 함수를 호출하여 우선순위 보드를 업데이트한다.
5. printBingo함수를 호출하여 빙고 판의 현황을 출력한다.

* void updatePriority

1. 매 턴에서 컴퓨터 보드의 단어가 선택되면 해당 인덱스를 매개변수로 받는 updatePriority함수를 호출한다.
2. 해당 위치의 우선순위 보드의 원소를 0으로 설정한다.
3. 이중 for문과 if문을 이용하여 해당 원소와 연결된 줄에 있는 0의 개수를 센다. 가로, 세로, 2개의 대각선에 있는 0의 개수를 int num[4]로 선언된 1차원 배열에 저장한다.
4. 이중 for문과 if문을 이용하여 해당 원소와 연결된 줄에 있는 원소들 중 0이 아닌 원소를 대상으로 해당 줄의 0의 개수만큼 증가시킨다.

* int countBingo: 보드에 있는 빙고 개수를 반환한다.

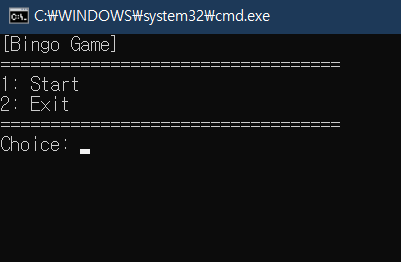
1. main함수에서 매 턴마다 호출하여 사용자 보드의 경우 result1, 컴퓨터 보드의 경우 result2에 함수의 반환 값을 저장한다.
2. 이중 for문, if문, strcmp함수를 이용하여 가로, 세로, 왼쪽 대각선, 오른쪽 대각선을 모두 검사한다. 빙고 한 줄이 발견될 때마다 0으로 초기화 했던 result변수 값이 하나씩 증가한다. 이 result값을 반환한다.

* main 함수 : printBingo함수를 호출하여 초기 빙고판을 출력한 후 while문 안에 userTurn함수, compTurn함수, countBingo함수를 넣는다. 이때, countBingo함수는 매 턴마다 호출하고, result1(사용자 보드 빙고 개수 저장)과 result2(컴퓨터 보드 빙고 개수 저장)에 반환 값을 각각 저장한다. if문을 사용하여 result1이나 result2가 bingo\_num보다 크거나 같으면 반복문 밖으로 나오고 게임 결과를 출력한다.

1. 빙고 종료

* void freeBingo: user\_board, comp\_board, priority\_board, size를 매개변수로 받아 동적 할당 해제를 시켜주는 함수이다. 이때, 해제는 할당 순서완 반대로 진행해야 한다.

1. 프로그램 실행방법 및 예제



< 초기 화면>

* 메뉴를 사용자로부터 입력 받는다.



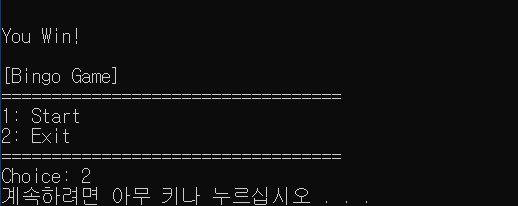
< 파일 단어 출력>

* 파일 속 모든 단어를 출력한다.
* 빙고 보드 크기를 입력 받는다.  
  <사용자 입력>
* 사용자로부터 빙고 판 크기, 승리 빙고 개수, 보드 생성 방식을 입력 받는다.

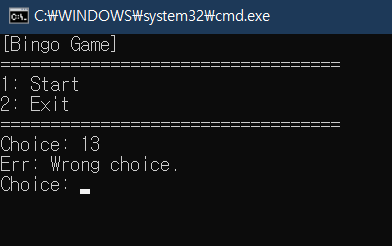
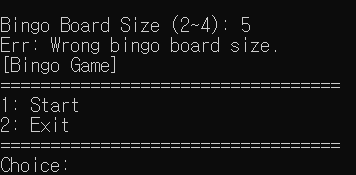


<빙고 출력>

* 사용자부터 단어를 입력 받고, 일치하는 것이 있으면 # 9개로 표시한다.

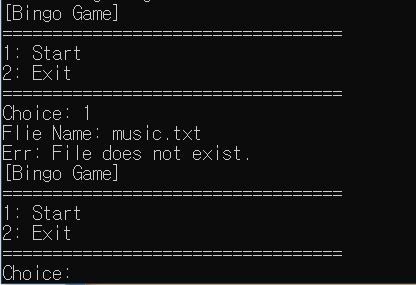
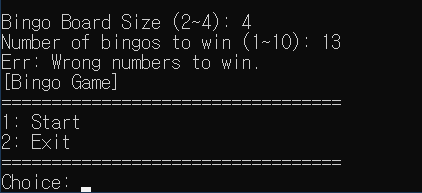
<게임 종료>

* 게임 결과를 출력한다.
* 다시 초기 메뉴 화면으로 돌아간다.
* 2를 선택하면 프로그램을 종료한다.

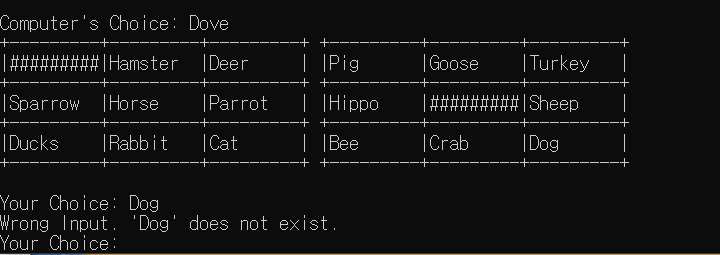


<에러 처리>

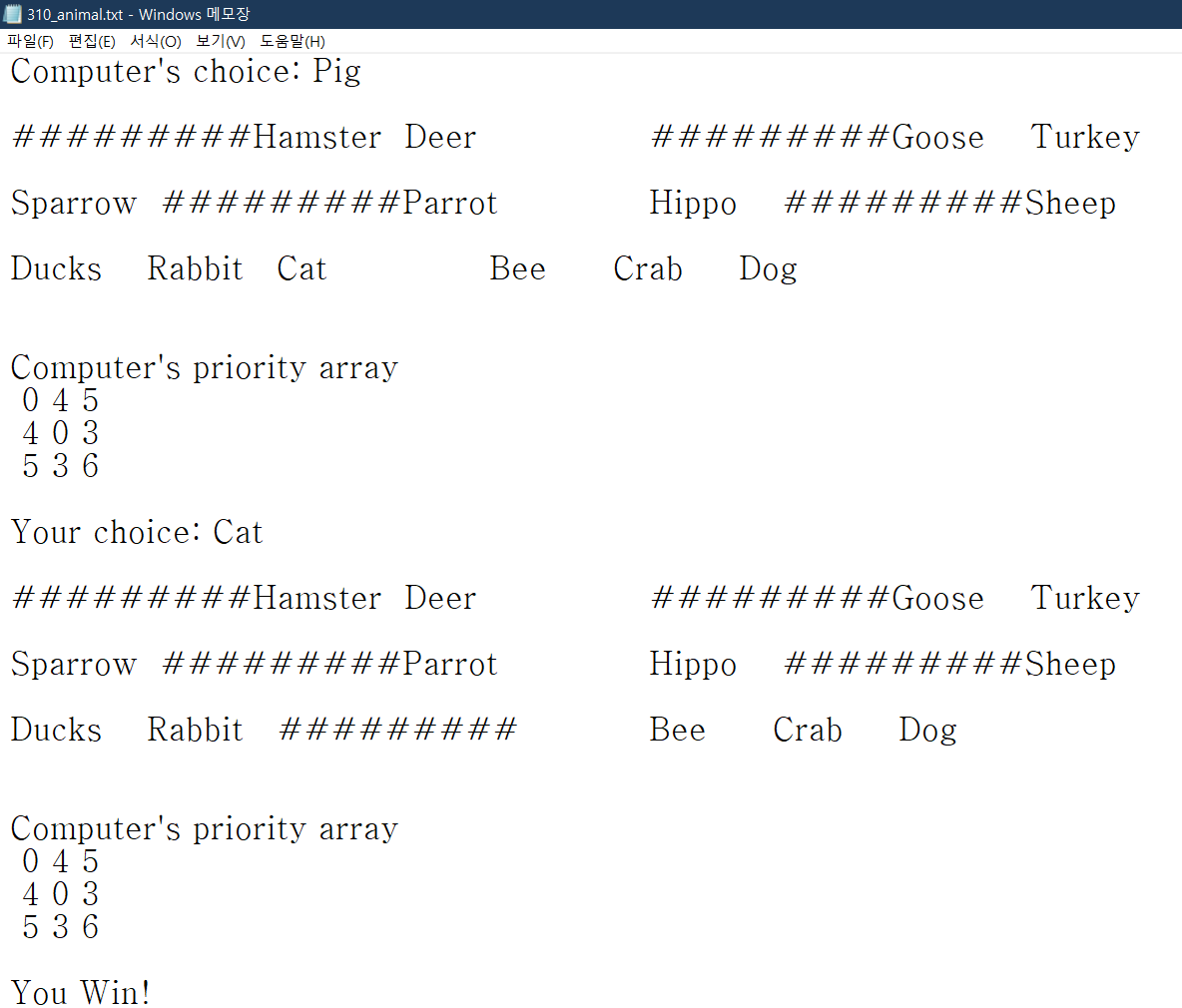
* 사용자가 1, 2 이외의 값을 입력 시 다시 입력 받는다.
* 범위를 넘어선 값을 입력 시 처음 메뉴 화면으로 돌아간다.



* 범위를 넘어선 승리 빙고 개수를 입력 시 다시 처음 메뉴 화면으로 돌아간다.
* 잘못된 파일명을 입력 시 처음 메뉴 화면으로 돌아간다.



* 사용자 보드에 존재하지 않는 단어를 입력 시 다시 입력 받는다.



<파일 출력>

* 우선 순위 보드, 빙고 판 현황, 선택 받은 단어, 게임 결과가 출력된 것을 확인할 수 있다.

1. 토론

* fgets함수를 이용해 파일 속 단어를 배열에 저장할 때 \n, \t같은 문자도 같이 입력되는 문제점이 발생하였었다.
* fsanf함수로 바꾸어 입력을 받으니 문자만 입력 받을 수 있었다.

1. 결론

본 과제에서는 빙고 판의 크기에 맞게 메모리의 크기를 조절하는 것이 관건이다. realloc, calloc, malloc 등의 함수를 활용하여 메모리를 동적으로 할당하는 법과, free함수를 통해 동적 할당 해제하는 방법을 익힐 수 있었다. 또한 문자열과 관련된 여러 함수를 다룸으로써 문자열에 대한 이해도, 응용 정도를 높일 수 있었다.

1. 개선방향

* 본 과제에서 요구하는 사항들은 문자열, 동적 할당과 같이, 이해를 제대로 하지 않으면 다루기 힘들 수 있다. 본인은 포인터에 대한 기초가 부족하여 어떤 주소를 갖는지 헷갈리는 경우가 다수 발생하였다. 또한 문자열을 저장하는 과정에서 적당한 메모리를 할당하는 것이 미숙하여 오류가 많이 발생하였다. 이번 과제를 통해 포인터와 문자열의 기초를 다지고 응용을 수월하게 할 수 있을 것이라 기대한다.