Programming Practice Final Project - 2048

2019-15861 염준영

Contents

1 빌드 방법

2048main.c 와 lib-cross.c lib-cross.h, mainfeatures.c mainfeatures.h를 같은 디렉토리에 두고 2048main.c와 lib-cross.c, mainfeatures.c 를 함께 컴파일한다.(C99 또는 C11 표준)

```
ex) gcc 2048main.c lib-cross.c mainfeatures.c -std=c11
```

2 게임 플레이 방법

우선 컴파일된 실행 파일을 실행한다.

```
ex) ./a.out
```

실행하면 자동으로 5 8개의 칸에 숫자 2또는 4가 채워진 판이 생성된다. w a s d를 입력하면 각각 위 왼쪽 아래 오른쪽으로 숫자들을 움직일 수 있다.(Enter키를 누를 필요 없음)

2048을 만들면 You Win! 메시지가 뜨고 게임을 계속 진행할 수 있다.

더 이상 움직일 수 있는 숫자가 없으면 You Lose! 메시지가 뜨고 게임이 종료된다.

3 구현된 추가 기능들

3.1 2 또는 4를 랜덤 생성

빈 칸(0이 있는 칸)에 숫자 2또는 4를 랜덤 생성해 주는 함수 number()를 구현하였다. rand()함수를 이용하여 0 3범위의 x좌표와 y좌표를 각각 랜덤으로 생성하고, 생성한 좌표에 0이 있으면 그 자리에 숫자 2 또는 4를 2를 90%, 4를 10%확률로 생성한다.(rand()함수를 호출해서 나온 수를 10으로 나눈 나머지가 0이면 4생성, 그외에 2생성) 생성한 좌표의 수가 0이 아니면 다시 rand()함수를 호출하여 나온 수가 짝수면 x좌표를 1증가시키고 4로 나눈 나머지로 x좌표를 갱신, 홀수면 y좌표를 1 증가시킨 후 4로 나눈 나머지로 갱신한다. 이와 같은 작업을 갱신된 좌표의 수가 0일 때까지 반복한 후 수를 생성한다.

```
    srand(time(NULL));
    int x = rand() % SIZE; // #define SIZE 4
    srand(time(NULL));
    int y = rand() % SIZE;
    while (numbers[x][y] != 0)
    {
      srand(time(NULL));
        if (rand() % 2 == 0)
            x = (x + 1) % SIZE;
        else
      y = (y + 1) % SIZE;
    }
    srand(time(NULL));
  if (rand() % 10 == 0)
      numbers[x][y] = 4;
  else
    numbers[x][y] = 2;

```

3.2 숫자 자동 생성

시작할 때 숫자를 입력받지 않고 자동으로 2 또는 4로 이루어진 숫자 5-8개를 랜덤한 위치에 생성해 준다. 구현 방식은 rand()함수로 5-8범위의 난수를 생성 후 for 문을 이용하여 생성된 숫자만큼 number()함수를 실행하였다.

4 점수 계산 방법

4.1 점수 숫자 계산

2048게임에서는 2^n과 2^n이 있는 블록이 합쳐지면 점수가 2^{n+1}만큼 증가한다. 전역변수로 score변수를 선언한 후, move_left()함수에서 블록이 합쳐지는 부분에서 score가 그만큼 증가하게 하였다.

```
    /* Merge code */
    for (int i = 0; i < SIZE; ++i)
        for (int j = 0; j < SIZE - 1; ++j)
            if (arr[i][j] == arr[i][j + 1] && arr[i][j] > 0)
            {
                arr[i][j] *= 2;
                scoreadded += arr[i][j];
                arr[i][j + 1] = 0;
                zeros++;
                moveable++;
```

```
                if (arr[i][j] == 2048)
                    wincounter = true;
            }

```

4.2 승리 조건 판별

stdbool.h 헤더 파일을 불러온 뒤, bool wincounter변수를 전역으로 false로 선언하고, move_left()함수에서 1024와 1024 가 합쳐지면 wincounter를 true로 갱신한다. main에서 매 루프마다 wincounter변수를 체크한다. 또한 bool status 변수를 같이 false로 설정한 뒤 status가 false일 때 You Win! 메시지가 출력되게 하였다. 그리고 You Win! 메시지를 출력한 후 status변수는 true로 바뀌어 You Win! 메시지가 계속 출력되지 않게 하였다.

```
        if (wincounter == true && status == false)
        {
            printf("You Win!");
            status == true;
        }

```

4.3 패배 조건 판별

int zeros와 int losecounter변수를 전역으로 선언한다. zeros는 판 안에 있는 0의 개수이다. 게임을 시작하면 16에서 생성된 숫자의 개수를 빼서 zeros를 결정한다.

```
srand(time(NULL));
int count = (rand() % 4) + 5;
for (int i = 0; i < count; ++i)
  number();
  zeros = 16 - count;

```

매 회 명령을 실행한 후, zeros가 0이면 패배 조건을 판별한다. int temp[SIZE][SIZE] 배열을 만들고, 상 하 좌 우 로 한번씩 움직여 본 후 리턴값(= 증가된 숫자 총합)을 losecounter에 더한다. 0의 개수가 0개라는 것은 만약 움직인다면 반드시 합쳐져서 점수가 증가해야 한다는 뜻이므로, 상 하 좌 우 전부 움직였는데도 losecounter가 계속 0이면 상 하 좌 우 어디로도 움직일 수 없다는 뜻이다. 따라서 패배했다고 결론지을 수 있다. 또한 zeros 변수를 이용하여 매 회 이동후 number()함수를 실행할지 하지않을지 결정할 수 있다(zeros가 0이면 빈칸이 없으므로 새로운 숫자 생성 X)