

2021-1 C++ 프로그래밍 실습과제 05

학번	2020136129	이름	최수연
----	------------	----	-----

(1) 각 문제에 대한 분석과 및 해결 방법

1. 교재 231~232쪽의 영역 채색 프로그램을 구현하고 테스트하라.

- (1) 프로그램 5.14를 구현함
- (2) 232쪽과 같은 자신만의 그림(image[][])을 만들어 잘 동작하는지 확인할 것
- (3) 함수의 매개변수를 2차원 배열(22행과 같이)로 하는 경우의 문제점 또는 불편한 점과 개선할 수 있는 방법을 보고서에 자세히 적을 것. (힌트: 380~383쪽의 내용을 참고할 수 있음)

[문제분석 및 해결방법]

(3) 함수의 매개변수를 2차원 배열(22행과 같이)로 하는 경우의 문제점 또는 불편한 점과 개선할 수 있는 방법을 보고서에 자세히 적을 것. (힌트: 380~383쪽의 내용을 참고할 수 있음)

함수의 매개변수를 2차원 배열을 프로그램 5.14의 22행과 같이 하는 경우, 고정된 2차원 배열을 사용하는 영역이 커질수록 배열의 크기도 커져 메모리 공간의 낭비가 발생할 수 있다. 이때, 2차원 배열을 동적으로 할당하면 이 문제점을 해결할 수 있다. 동적 할당은 런타임 중 2차원 배열 구조를 직접 할당 및 해제할 수 있기 때문에 코드에서 미리 배열의 크기를 고정하지 않고 사용할 수 있어서 프로그램 5.14보다 메모리 공간 낭비를 훨씬 줄일 수 있다.

2. 교재 237~241쪽의 지뢰 찾기 게임을 구현하고 테스트하라.

- (1) 책에 있는 코드를 구현하고 테스트 할 것.
- (2) 243쪽 ~ 244쪽의 고찰 내용 생각해 볼 것. 특히 (4)의 내용에 대해서는 코드에서 사용된 부분을 찾고, 사용 이유를 보고서에 정리할 것.

[문제분석 및 해결방법]

(4) 많은 문법들이 코드에 사용되었다. 각 부분을 찾아 의미를 생각해보라.

- 참조형 매개변수와 참조자 변환

```
inline int& mask(int x, int y) { return MineMapMask[y][x]; }
inline int& label(int x, int y) { return MineMapLabel[y][x]; }
// mask()와 label() 함수의 반환형을 참조형으로 함으로써 다음 아래와 같은 코드 사용이 가능하다.
label(x, y) = countNbrBombs(x, y);
mask(x, y) = Open;
mask(x, y) = Flag;
/* 위 코드는 반환형이 참조형이기 때문에 가능하다. 참조형을 사용하여 메모리 공간에 별명을 붙여 별명을 사용해 값을 넣음으로써 포인터를 사용해 주소에 의해 호출 및 반환하는 함수보다 사용하기 편하고 코드가 간결해지도록 하였다 */
static bool getPos(int& x, int& y) {...}
/* 위 코드의 경우에도 매개변수를 참조자형으로 변경함으로써 인수 값을 매개변수로 복사하는 것이 아닌 변수 자체를 호출할 수 있게 할 수 있다. */
```

- 인라인 함수와 디폴트 매개변수

```
inline int& mask(int x, int y) { return MineMapMask[y][x]; }
inline int& label(int x, int y) { return MineMapLabel[y][x]; }
inline bool isValid(int x, int y) { return (x >= 0 && x < nx && y >= 0 && y < ny); }
inline bool isBomb(int x, int y) { return isValid(x, y) && label(x, y) == Bomb; }
inline bool isEmpty(int x, int y) { return isValid(x, y) && label(x, y) == Empty; }
/* 위 코드에서 각 함수들은 인라인 함수를 사용하여 매크로 함수와 같이 전처리에서 빠르게 처리가 되면서
동시에 일반 함수와 동일한 완전한 함수의 형태를 가지도록 하였다. 또한 인라인 함수를 사용함으로써 매크
로 함수와 달리 매개변수의 자료형을 철저히 검사하기 때문에 함수를 잘못 사용할 수 있는 위험을 줄인다.*/
extern void playMineSweeper(int nBomb = 13);
static void init(int total = 9)
/* 디폴트 매개변수를 사용하여 매개변수 nBomb에 디폴트 값 13, total에 디폴트 값 9를 저장한다. 따라서
지뢰의 개수가 입력되지 않으면 각 디폴트값으로 대체된다. */
```

- 재귀호출

```
static void dig(int x, int y) {      //(x, y)를 파는 함수
    if (isValid(x, y) && mask(x, y) != Open) {
        mask(x, y) = Open;
        if (label(x, y) == 0) {
            dig(x - 1, y - 1);
            dig(x - 1, y);
            dig(x - 1, y + 1);
            dig(x, y - 1);
            dig(x, y + 1);
            dig(x + 1, y - 1);
            dig(x + 1, y);
            dig(x + 1, y + 1);
        }
    }
}
/* dig()함수에서는 해당 영역을 파내는 함수로, 재귀호출을 사용하여 해당 영역의 주변 인접한 자리의 값이
0일 때 0이 없어질 때까지 반복하여 파낸다. */
```

- 정적 함수와 정적 변수

```
<정적 전역변수>
static int MineMapMask[DIM][DIM];      //Hide, Open, Flag
static int MineMapLabel[DIM][DIM];     //0~8, 9(bomb)
static int nx = DIM, ny = DIM;
static int nBomb = DIM;
<정적 함수>
static void dig(int x, int y)          //(x, y)를 파는(여는) 함수
static void mark(int x, int y)         //(x, y)에 깃발을 꽂는 함수
static int getBombCount()              //깃발의 수를 계산하는 함수
static void print()                    //지뢰 맵 화면 출력 함수
static int countNbrBombs(int x, int y) //인접한 지뢰의 수 계산 함수
```

```
static void init(int total = 9)
static bool getPos(int& x, int& y)      //키보드 좌표 입력 함수
static int checkDone()                //게임 종료 검사 함수
/* 위의 변수와 함수를 static으로 선언함으로써 이들의 가시범위를 해당파일 내로 국한시킨다. 따라서 다른
외부 파일에 영향이 미치지 않도록 한다. */
```

- 나열형(enum)

```
enum LabelType { Empty = 0, Bomb = 9 };
enum MaskType { Hide = 0, Open, Flag };
/* 나열형 선언을 통해 Hide = 0에 따라 Open = 1, Flag = 2로 설정하였다, Open과 Flag가 1, 2로 설정
되는 이유는 나열형 enum은 바로 앞에 나온 요소의 값보다 하나 더 큰 값이 정수값이 할당되기 때문에
Hide의 초깃값이 0으로 설정되었으므로 Open과 Flag가 차례대로 1, 2가 되는 것이다. 그러나 Empty나
Bomb처럼 직접 요소들의 값을 지정할 수도 있다. 본 코드에서는 이 나열형의 요소들을 사용하여 다른 함수
에 대입하고 비교하는 데에 사용한다. */
```

- bool 반환함수 및 문자처리 함수 toupper()

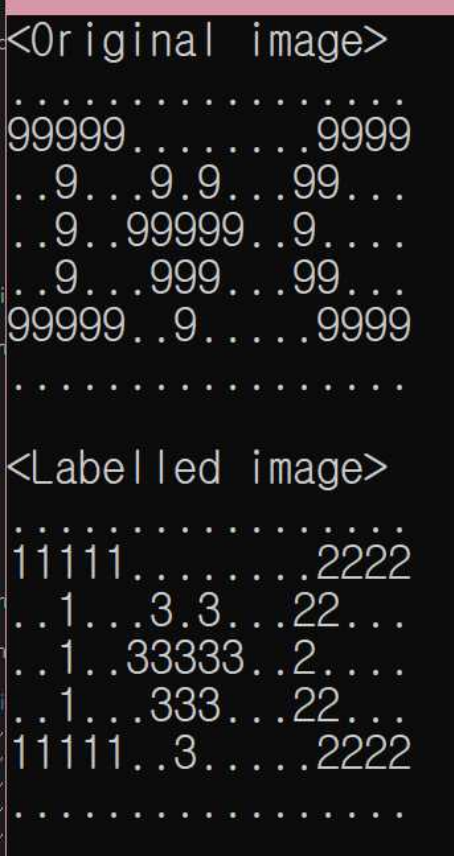
```
static bool getPos(int& x, int& y) {
    printf("\n지뢰(P)행(A-I)열(1-9)\n 입력 --> ");
    bool isBomb = false;
    y = toupper(_getche()) - 'A';
    if (y == 'P' - 'A')
    {
        isBomb = true;
        y = toupper(_getche()) - 'A';
    }
    x = _getche() - '1';
    return isBomb;
}

-----
do {
    print();
    bool isBomb = getPos(x, y);      //위치 입력
    if (isBomb) mark(x, y);          //깃발 위치이면 ==> mark() 호출
    else dig(x, y);                  //아니면 ==> dig() 호출
    status = checkDone();            //게임 종료 상황 검사
} while (status == 0);
/* toupper() 함수를 사용하여 입력받은 문자가 소문자 및 대문자에 관계없이 모두 대문자로 변환하여 반환
되도록 설정한다.
또한 자료형 bool을 사용하여 true(1)면 mark()를 false(0)이면 dig()를 호출한다. */
inline bool isValid(int x, int y) { return (x >= 0 && x < nx&& y >= 0 && y < ny); }
inline bool isBomb(int x, int y) { return isValid(x, y) && label(x, y) == Bomb; }
inline bool isEmpty(int x, int y) { return isValid(x, y) && label(x, y) == Empty; }
/* 위 함수들의 자료형을 bool로 선언하여 isValid는 x, y가 유효한 위치인지 아닌지, isBomb는 지뢰인지
아닌지, isEmpty는 빈칸인지 아닌지를 true(1) 또는 false(0)로 반환하여 게임을 진행시킨다. */
```

(2) 자신이 구현한 주요 코드

<pre>unsigned char image[HEIGHT][WIDTH] = { 0, 9, 9, 9, 9, 9, 0, 0, 0, 0, 0, 0, 0, 0, 9, 9, 9, 9, 9, 9, 9, 0, 0, 9, 0, 0, 0, 9, 0, 9, 0, 0, 0, 9, 9, 9, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 9, 9, 9, 9, 9, 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0, 9, 9, 9, 0, 0, 0, 9, 9, 9, 0, 0, 0, 0, 0, 9, 9, 9, 9, 9, 0, 0, 9, 0, 0, 0, 0, 0, 9, 9, 9, 9, 9, 9, 9, 0, };</pre>	<p>" I ♥ C "를 화면에 출력하였다.</p>
--	------------------------------

(3) 다양한 입력에 대한 테스트 결과

1. 교재 231~232쪽의 영역 채색 프로그램을 구현하고 테스트하라.	
	
2. 교재 237~241쪽의 지뢰 찾기 게임을 구현하고 테스트하라.	

Microsoft Visual Studio 2019

발견:10 전체:10

	①	②	③	④	⑤	⑥	⑦	⑧	⑨	
A				1	α	2	1	1	α	1
B				1	3	α	2	2	2	2
C			1	1	3	α	2	1	α	1
D	1	2	α	2	2	2	2	1	1	
E	α	2	1	1	1	α	1			
F	2	2			1	1	1			
G	α	1								
H	1	1					1	1	1	
I							1	α	1	

성공: 탐색 성공!!!

발견:13 전체:13

	①	②	③	④	⑤	⑥	⑦	⑧	⑨	
A				1	1	1				
B	2	2	2	α	1			1	1	
C	α	α	4	3	2			1	α	
D	3	4	α	α	2	1	2	2	2	
E	1	α	3	2	2	α	2	α	1	
F	2	2	1		1	1	3	2	2	
G	α	1						1	α	1
H	2	2	1				1	2	2	1
I	1	α	1				1	α	1	

성공: 탐색 성공!!!

(4) 코드에 대한 설명 및 해당 문제에 대한 고찰
없음.

(5) 이번 과제에 대한 느낀점
영역 채색 프로그램과 지뢰 찾기 게임의 코드를 처음 보았을 때는 이번 C++에서 배우는 어려운 개념들도 있고 코드도 길어서 접근하기가 두려웠다. 그러나 이번 과제에서는 교수님께서 그동안 배운 C++ 개념들을 영역 채색 프로그램과 지뢰 찾기 게임 코드를 통해 스스로 복습하고 분석해보는 방향으로 과제를 내주셔서, C++에 관한 개념 정리에 도움이 많이 되었다. 그리고 본 강의랑 교재에 이해하기 쉽게 설명되어 있어서 과제 할 때 도움이 많이 되었다.

(6) 궁금한 점이나 건의사항
딱히 없습니다.