

<컴퓨터 알고리즘 과제 2>

제출인: 한아림

학번: 2022111354

분반: 01

학과: 정보보호학과

<목차>

1. 문제 1

- 1-1. 문제 정의
- 1-2. 순서도와 설명
- 1-3. 프로그램 구현
- 1-4. 실행 화면

2. 문제 2

- 2-1. 문제 정의
- 2-2. 순서도와 설명
- 2-3. 프로그램 구현
- 2-4. 실행 화면

3. 문제 3

- 3-1. 문제 정의
- 3-2. 순서도와 설명
- 3-3. 프로그램 구현
- 3-4. 실행 화면

1. 문제 1

1-1. 문제 정의

1	3	6	10	15
2	5	9	14	19
4	8	13	18	22
7	12	17	21	24
11	16	20	23	25

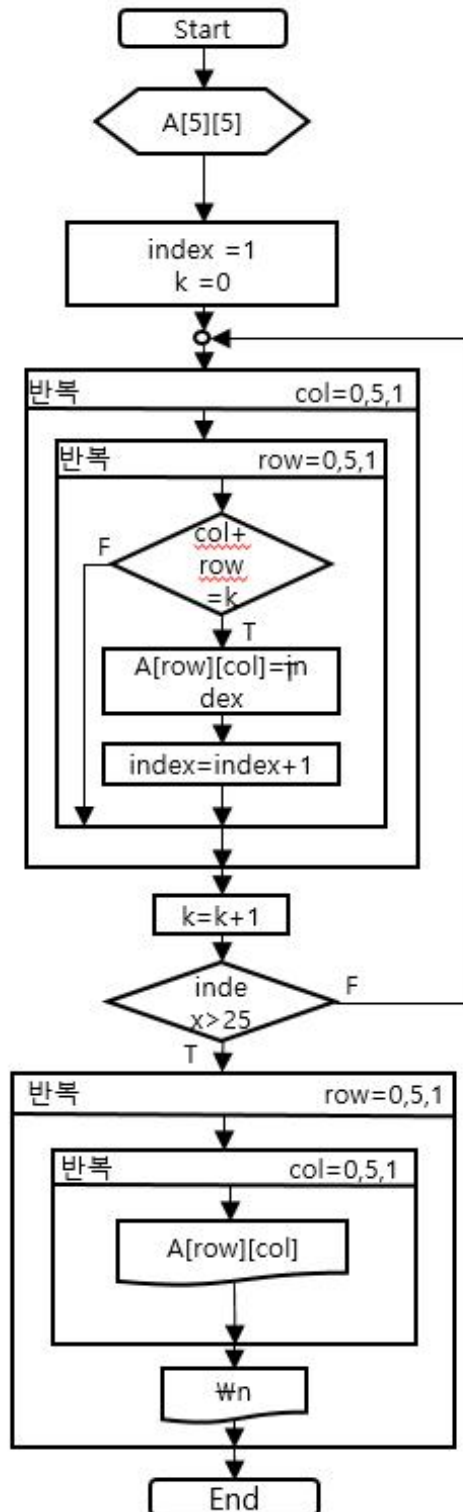
첫 번째 문제는 25개의 원소를 가지는 배열 A에 대하여, 위와 같은 모양으로 배열 원소를 저장해주는 프로그램을 구현하는 문제이다. 배열 A는 5행 5열, 즉 25개의 원소를 가지는 배열로, A[5][5]로 정의된다.

위 그림처럼 배열 A의 원소들은 대각선 방향으로 차례로 저장되어 있는 모습이다. 배열의 원소는 1부터 1씩 증가하여 25까지 증가한다. 배열 원소값의 증가 및 저장 방향은 우상향이다. 이동 방향이 하나밖에 존재하지 않으므로, 별도의 방향 지정 변수는 필요하지 않음을 알 수 있다. 단, 이전 대각선에서 다음 대각선으로 넘어가도록 프로그램을 설정해야 하기에, 어느 부분에서 다음 대각선으로 넘어갈 것인지에 대한 조건을 정립할 필요가 있다.

먼저 중심 배열 A를 정의하고, 반복문을 통해 값을 순서대로 배정할 계획이다. 값들의 배정은 열을 우선하여 채워진다. 또 행과 열 변환을 위한 변수를 별도로 설정해야 한다. 이 변수의 값은 0부터 시작하여 반복문의 실행이 1회 될 때마다 1씩 증가한다. 그리고 반복문 실행 도중 열 번호와 행 번호의 합이 행, 열 변환 변수의 값과 같아진다면, 배열에 인덱스 값을 삽입하도록 할 것이다. 인덱스값을 삽입한 후에는 인덱스값을 1씩 증가시키 인덱스값이 25를 넘지 않는 범위까지 반복을 지속할 계획이다.

이 다음 완성된 배열을 출력하는 과정을 거치면 문제의 풀이가 끝난다. 배열 출력은 행을 우선으로 출력한다.

1-2. 순서도 설계와 설명



좌측의 순서도는 문제 1을 해결하기 위한 순서도이다. 먼저 배열 A를 선언한 후, 배열의 인덱스값을 나타내는 변수 index를 선언한다. 이어 행과 열 변환을 위한 변수 k를 선언한 후, 초기값을 0으로 지정한다. 이때 index의 경우 1부터 시작해야 함에 유의하며, 초기값을 1로 지정한다.

이어 배열에 인덱스값을 부여하기 위한 반복문을 실행한다. 반복 변수 col의 초기값을 0으로 설정하고, col이 5가 될 때까지 1씩 증가시키며 아래 과정을 반복한다.

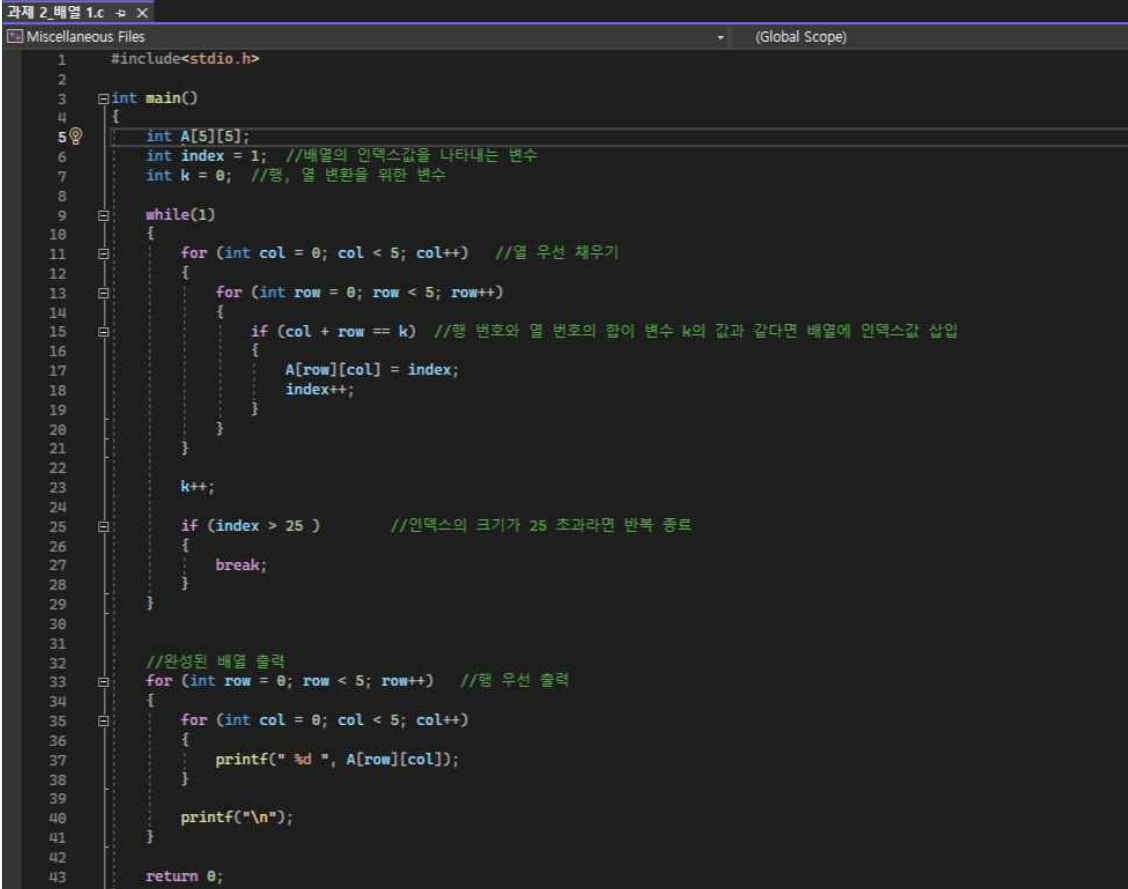
이때 이중 반복이 쓰이는데, 또 다른 반복 변수 row의 초기값을 0으로 지정하고, 5가 될 때까지 1씩 증가시키며 반복문을 수행한다. 만일 변수 row와 col의 값이 k의 값과 같아진다면, 배열 A의 row행 col열에 인덱스값을 삽입한다. 인덱스값을 삽입한 후 인덱스 값을 1 증가시키고, 안쪽 반복문의 반복 변수인 row의 값을 1 증가시킨다. 만일 row, col 변수의 값들의 합이 k와 다르다면, 별도로 인덱스를 삽입하지 않고 곧바로 row의 값을 1 증가시킨다.

안쪽 반복문의 반복이 끝나면 밖 반복문 col 변수의 값을 1 증가시킨다. 안과 밖 반복문의 반복이 모두 끝나면 k의 값을 1 증가시키고, 인덱스의 값이 25 초과인지를 검사한다.

만일 인덱스 값이 25를 넘었다면 반복을 중단하고, 그렇지 않다면 다시 반복문을 처음부터 수행한다.

인덱스의 값이 25를 넘었다면 완성된 배열을 출력하는 과정을 거친다. 배열의 출력 또한 이중 반복을 이용하여 출력하며, 행을 우선으로 하여 출력한다. 또 출력되는 모양이 정사각형과 같아야 하므로 한 행이 모두 출력될 때마다 줄 변환을 해 주어야 한다. 배열이 모두 출력된 경우, 순서도는 끝난다.

1-3. 프로그램 구현



The screenshot shows a C program in a code editor. The program is titled '과제 2 배열 1.c' and is in the 'Global Scope'. It includes the standard header `<stdio.h>`. The `main` function starts by declaring a 5x5 integer array `A`, an `index` variable starting at 1, and a `k` variable starting at 0. A `while(1)` loop begins, containing a nested `for` loop for columns (`col` from 0 to 4). Inside this, another nested `for` loop for rows (`row` from 0 to 4) is present. Within the row loop, an `if` statement checks if `col + row == k`. If true, it assigns `A[row][col] = index` and increments `index`. After the row loop, `k` is incremented. An `if` statement checks if `index > 25`, and if true, it breaks the `while` loop. Finally, a `for` loop prints the array `A` row by row, and the program returns 0.

```
1  #include<stdio.h>
2
3  int main()
4  {
5      int A[5][5];
6      int index = 1; //배열의 인덱스값을 나타내는 변수
7      int k = 0; //행, 열 변환을 위한 변수
8
9      while(1)
10     {
11         for (int col = 0; col < 5; col++) //열 우선 채우기
12         {
13             for (int row = 0; row < 5; row++)
14             {
15                 if (col + row == k) //행 번호와 열 번호의 합이 변수 k의 값과 같다면 배열에 인덱스값 삽입
16                 {
17                     A[row][col] = index;
18                     index++;
19                 }
20             }
21         }
22
23         k++;
24
25         if (index > 25 ) //인덱스의 크기가 25 초과라면 반복 종료
26         {
27             break;
28         }
29     }
30
31     //완성된 배열 출력
32     for (int row = 0; row < 5; row++) //행 우선 출력
33     {
34         for (int col = 0; col < 5; col++)
35         {
36             printf(" %d ", A[row][col]);
37         }
38
39         printf("\n");
40     }
41
42     return 0;
43 }
```

+ 소스 코드는 보고서 파일과 함께 Lms에 참조

1-4. 실행 화면



The screenshot shows a Microsoft Visual Studio Debug console window. The title bar reads "Microsoft Visual Studio Debug" with standard window controls. The console output displays five lines of numbers, each line containing five space-separated integers. Below the numbers, a message indicates the program has exited with code 0, followed by a prompt to press any key to close the window.

```
1 3 6 10 15  
2 5 9 14 19  
4 8 13 18 22  
7 12 17 21 24  
11 16 20 23 25  
  
C:\Users\USER\Desktop\Computer Algorithm Assignment 2_Array 1\x64\Debug\Computer Algorithm Assignment 2.exe (process 21260) exited with code 0.  
Press any key to close this window . . .|
```

2. 문제 2

2-1. 문제 정의

1	3	4	10	11
2	5	9	12	19
6	8	13	18	20
7	14	17	21	24
15	16	22	23	25

두 번째 문제는 25개의 원소를 가지는 배열 B에 대하여 위와 같은 순서로 원소를 저장해주는 알고리즘을 구현하는 문제이다. 배열 B는 5행 5열로, 25개의 배열 원소를 가지는 B[5][5]로 정의된다.

배열 B의 경우 대각선별로 원소를 저장하는 방향이 상이하다. 홀수 번째 대각선들의 경우 좌하향으로 배열 원소가 저장된 양상을 보이고, 반대로 짝수 번째 대각선들의 경우 문제 1의 배열 A와 마찬가지로 우상향으로 배열 원소가 저장되어 있다.

필자의 경우는 우선적으로 25개의 배열을 모두 우상향 저장 형식으로 채워넣은 뒤, 특정 대각선들만 골라 좌하향으로 배열 원소를 새롭게 채워넣는 방식을 선택했다.

먼저 문제 1의 경우와 마찬가지로 이중 반복문을 사용하여 배열을 채울 계획이고, 행을 나타내는 변수의 값과 열을 나타내는 변수의 값의 합이 행,열 변환 변수의 값과 같을 경우에만 배열에 값을 저장해줄 것이다. 만일 인덱스값을 나타내는 변수의 값이 25 초과라면 반복을 종료하고 좌하향으로 배열 원소를 새롭게 채워넣는다.

배열 원소에 새로운 값을 저장하는 과정은 2 부분으로 구성되는데, 크게 중앙 대각선을 기준으로 위 부분의 값을 채우는 과정과 이 아랫부분의 값을 채우는 과정으로 나뉘어진다. 중앙 대각선 위 부분의 값 대입의 경우 새롭게 원소들을 채워넣기 위해 행과 열을 나타내는 변수들을 각각 정의하고, 이중 반복문을 통해 값들을 새로이 채워넣는다. 단, 홀수 번째 배열들에 한해서만 값을 새로 배정해야 하므로, 반복 변수는 1이 아닌 2씩 증가한다. 배열 원소에 새 값을 저장한 후 배열 인덱스의 값을 1 증가시키고, 다음 대각선으로 넘어갈 때는 행 번호에 2를 더한 값만큼 증가시킨다.

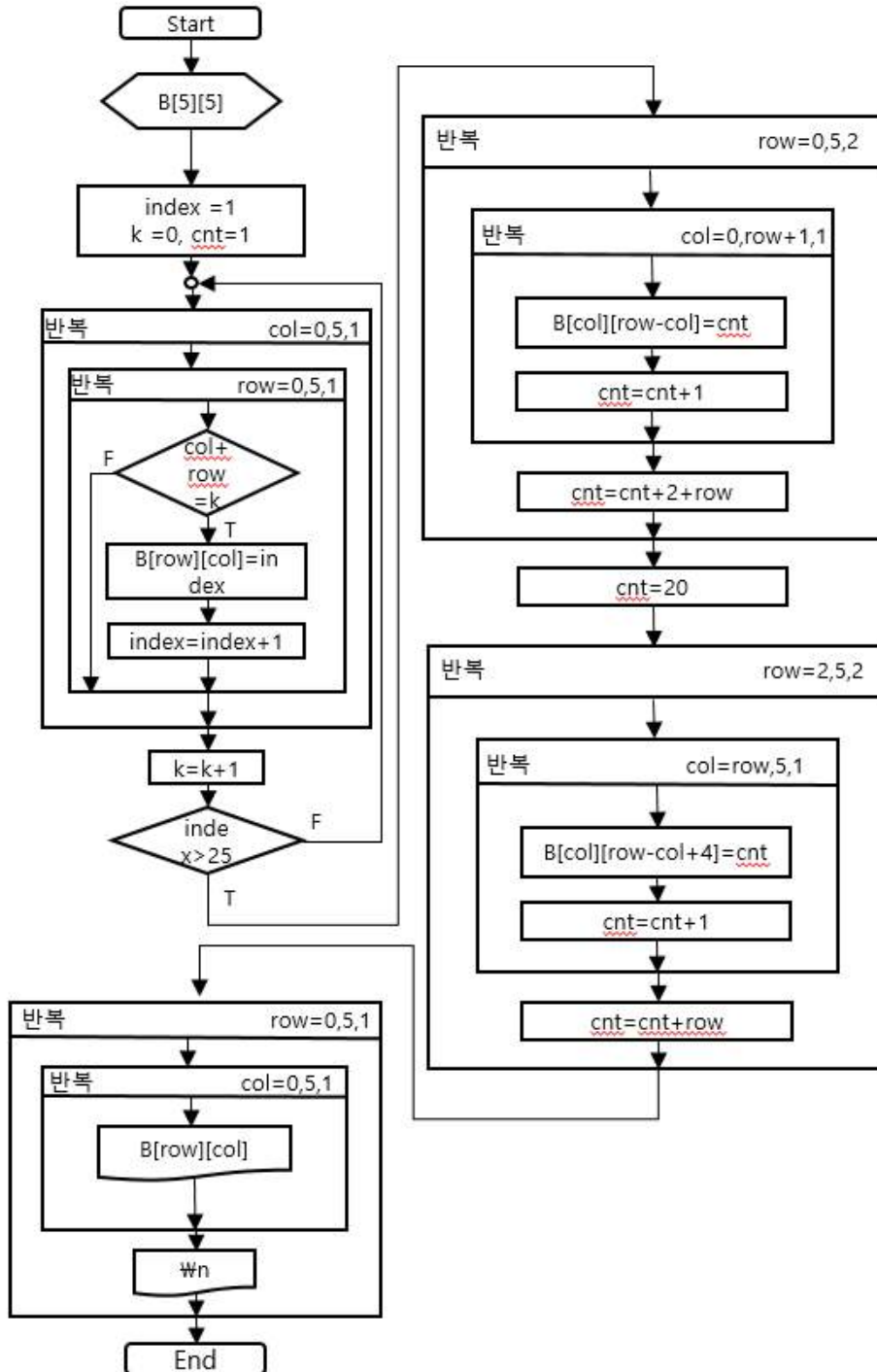
중앙 대각선 아랫 부분의 경우 인덱스 값을 20으로 초기화한 후 중첩 반복을 진행한다. 여기서 유의해야 할 점은 행을 나타내는 변수의 초기값을 2로 지정해야 한다. 또 배열에 인덱스값

을 저장할 때, 행을 나타내는 변수값에서 열을 나타내는 변수값을 뺀 값에 4를 더한 값을 배열의 열로 하여야 한다. 마찬가지로 다음 대각선으로 이동할 때 반복 변수는 1이 아닌 2씩 증가한다.

또 반복이 1회 실행될 때마다 인덱스값은 인덱스값에 행을 나타내는 변수 값을 더한 값만큼 증가함에 유의해야 한다. 모든 반복이 완료되면 배열에 값을 채워넣는 과정이 끝난다.

이 다음 완성된 배열을 출력하는 과정을 거치면 문제의 풀이가 끝난다. 배열 출력은 행을 우선으로 출력한다.

2-2. 순서도 설계와 설명



위의 순서도는 문제 2를 해결하기 위한 순서도이다. 먼저 25개의 원소를 가지는 배열 B를 선언한다. 그 다음 인덱스값을 나타내는 변수 index와 cnt를 선언해주고, 초기값을 1로 설정

한다. 마지막으로 행과 열 변환을 위한 변수인 k 를 초기값을 0으로 하여 선언해준다. 변수를 선언한 후 먼저 대각선 우상향 방향으로 배열을 채우기 위한 과정을 진행한다.

배열에 값을 저장할 때는 열을 우선으로 채워지는데, col 변수의 초기값을 0으로 하여 col 변수의 값이 5가 될 때까지 1씩 증가시켜가며 중첩 반복문을 실행한다. 또 row 반복변수를 이용, 초기값을 0으로 설정한 이후 5가 될 때까지 1씩 증가시켜가며 과정을 반복한다. 만일 col , row 변수의 합이 변수 k 와 같은 경우, 배열 B 에 인덱스값을 저장하고, 같지 않을 경우에는 저장하지 않는다. 값을 저장한 후 인덱스 값을 1씩 증가시키며 반복문을 이어서 수행한다. 이중 반복문이 모두 끝나면 k 의 값을 1 증가시키고, $index$ 의 값이 25를 넘지 않는지 검사한다. 만일 인덱스의 크기가 25 초과일 경우는 반복을 종료한다.

우상향 배열을 모두 채우고 난 이후에는 좌하향 대각선 배열에 새로운 값들을 저장해야 한다. 이때도 마찬가지로 중첩 반복문이 사용된다. 먼저 중앙 대각선 윗부분을 먼저 채워넣는다.

중앙 대각선 윗부분을 채워넣기 위해서는 먼저 반복 변수 row 의 초기값을 0으로 설정한 후 5가 될 때까지 1씩 증가시켜가며 반복한다. 또 내부 반복 변수 col 의 초기값을 0으로 지정하고, col 변수의 값이 row 변수의 값과 같거나 작을때까지만 col 변수의 값을 1씩 증가시켜가며 배열 B 에 인덱스 값을 저장하는 과정을 반복한다. 이때 유의해야 할 점은 B 의 col 행 $row-col$ 열에 cnt 변수의 값을 저장해야 한다는 것이다. 내부 반복이 모두 완료될 때마다 cnt 의 값은 cnt 에 $row+2$ 값을 더한 만큼 증가한다.

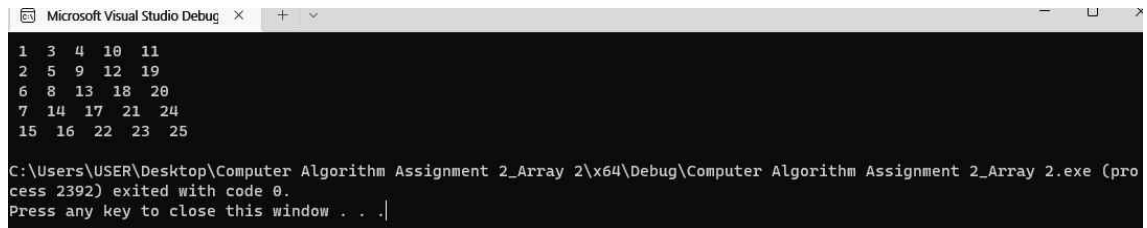
중앙 대각선의 윗부분의 배열 원소에 값이 모두 들어갔다면, 이제 중앙 대각선 아랫부분의 배열 원소를 채워넣어야 한다. 중첩 반복 과정을 수행하기에 앞서 cnt 변수의 값을 20으로 초기화해주는 과정을 거친다. 이후, 앞의 과정과 마찬가지로 외부 반복변수 row 와 내부 반복변수 col 을 사용한다. row 와 col 의 초기값은 각각 2, row 이고, 모두 5가 될 때까지 2,1씩 증가하며 반복을 수행한다. 내부 반복이 완료될 때마다 cnt 변수는 row 변수값을 더한 만큼 증가한다. 반복문을 수행하며 배열 B 의 col 행 $row-col-4$ 열에 cnt 변수의 값을 저장하고, 값을 저장한 후 cnt 변수의 값을 1 증가시킨다.

중앙 대각선의 아랫 부분까지 모두 채운 후에는 완성된 배열을 출력한다. 배열은 중첩 반복을 수행하며 열 우선으로 출력된다. 배열의 모양을 갖추어 출력하기 위해 한 열이 모두 출력될 때마다 줄바꿈 기호를 넣어주었다. 배열의 출력이 완료되면 순서도는 끝난다.

2-3. 프로그램 구현

```
Computer Algorithm Assignment 2_Array 2 (Global Scope)
1  #include <stdio.h>
2
3  int main()
4  {
5      int B[5][5] = { 0 };
6      int row, col, cnt = 1;
7      int index = 1; //배열의 인덱스값을 나타내는 변수
8      int k = 0; //행, 열 변환을 위한 변수
9
10     //대각선 - 우상향 배열 채우기
11     while (1)
12     {
13         for (int col = 0; col < 5; col++) //열 우선 채우기
14         {
15             for (int row = 0; row < 5; row++)
16             {
17                 if (col + row == k) //행 번호와 열 번호의 합이 변수 k의 값과 같다면 배열에 인덱스값 삽입
18                 {
19                     B[row][col] = index;
20                     index++;
21                 }
22             }
23
24             k++;
25
26             if (index > 25) //인덱스의 크기가 25 초과라면 반복 종료
27             {
28                 break;
29             }
30         }
31     }
32
33     //대각선 - 좌하향 배열 채우기
34     for (row = 0; row < 5; row=row+2)
35     {
36         for (col = 0; col <= row; col++)
37         {
38             B[col][row - col] = cnt;
39             cnt++;
40         }
41
42         cnt=cnt+row+2;
43     }
44
45     cnt = 20;
46     for (row = 2; row < 5; row=row+2)
47     {
48         for (col = row; col < 5; col++)
49         {
50             B[col][row - col + 4] = cnt;
51             cnt++;
52         }
53
54         cnt = cnt + row;
55     }
56
57     //완성된 배열 출력
58     for (row = 0; row < 5; row++)
59     {
60         for (col = 0; col < 5; col++)
61         {
62             printf(" %d ", B[row][col]);
63         }
64         printf("\n");
65     }
66 }
```

2-4. 실행 화면



```
Microsoft Visual Studio Debug  X + -
1 3 4 10 11
2 5 9 12 19
6 8 13 18 20
7 14 17 21 24
15 16 22 23 25
C:\Users\USER\Desktop\Computer Algorithm Assignment 2_Array 2\x64\Debug\Computer Algorithm Assignment 2_Array 2.exe (process 2392) exited with code 0.
Press any key to close this window . . .|
```

3. 문제 3

3-1. 문제 정의

25	10	11	12	13
24	9	2	3	14
23	8	1	4	15
22	7	6	5	16
21	20	19	18	17

3번째 문제는 달팽이 모양으로 배열의 원소를 채워넣는 프로그램을 구현하는 것이다. 필자는 먼저 일반 달팽이 모양으로 배열 원소를 채워넣은 후, 배열을 좌우반전 처리하였다. 이어 좌우반전이 완료된 배열을 좌측으로 90도 회전하여 위와 같은 그림의 배열을 완성하였다.

배열 C는 5행 5열, 즉 25개의 배열 원소를 가지는 배열로, C[5][5]로 정의된다. 또 배열을 좌측으로 회전시키는 과정에서 사용할 배열 TEMP 또한 미리 선언한다. 배열 TEMP 또한 5행 5열의 배열로, 25개의 배열 원소를 가지고 있다. 추가적으로 배열 원소에 저장할 인덱스 값을 나타내는 변수 index와 오른쪽, 왼쪽으로 이동하기 위한 변수 a와 b를 선언한다. 또 배열의 행과 열을 나타내는 변수를 선언하고, 반복변수들과 배열의 좌우반전 시 사용할 스와핑 변수 또한 선언한다. 이어 배열을 채워넣는 과정에서 방향이 바뀌어야 하기에, 배열을 채워넣는 방향을 나타내는 방향 변수까지 정의해야 한다.

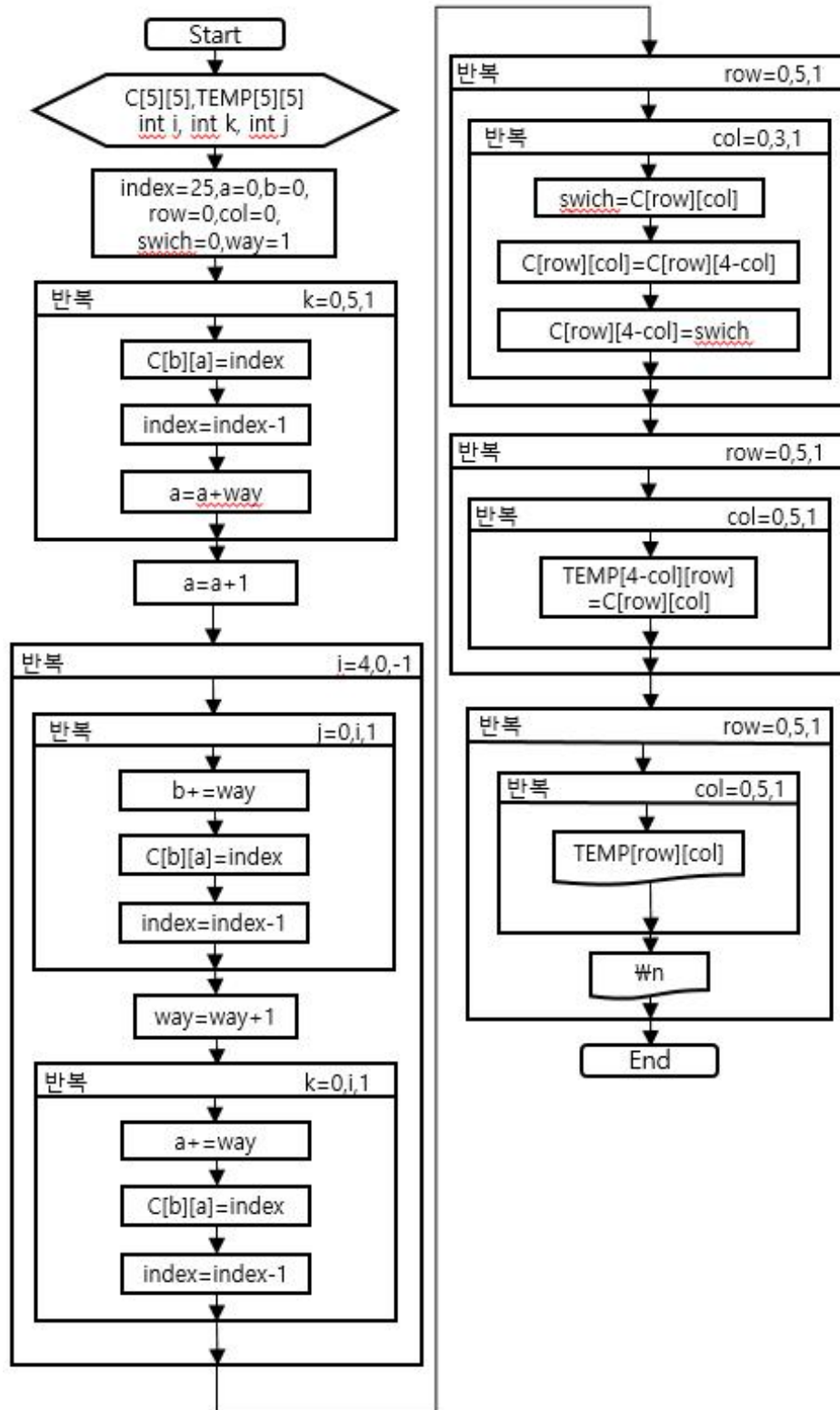
배열 안에 달팽이 모양으로 인덱스를 채워넣는 과정에서는 반복문이 사용되며 반복변수가 5가 될 때까지 인덱스값을 1씩 줄여가며 배열에 채워넣는다. 한 방향으로의 값 저장이 끝나면 방향 변수의 부호를 바꾸어 다른 방향으로 나머지 원소에 값을 채워넣는다.

25개의 원소에 모두 값을 저장하고 난 후, 위 그림과 같은 모습을 연출하기 위해 완성된 배열을 좌우반전한 뒤 좌측으로 90도 회전시킨다. 배열을 좌우반전하는 과정에서 스위치 변수를 사용하며, 중첩 반복문을 사용한다.

배열의 좌우반전이 끝나면 마지막으로 배열을 좌측으로 90도 회전시키는 과정을 수행한다. 이 또한 중첩 반복을 수행하며, 과정 수행을 위한 새로운 배열이자 최종적으로 출력될 배열을 의미할 TEMP를 사용한다.

위의 모든 처리가 마무리된 후 완성된 배열을 출력해주면 알고리즘은 끝이 난다.

3-2. 순서도 설계와 설명



위 순서도는 문제 3을 해결하기 위한 순서도이다. 먼저 배열 C, TEMP를 정의하고, 이어 필요한 변수들을 정의한다. 변수 index의 경우 배열 원소에 저장될 인덱스 값을 나타내는 변수로, 초기값을 25로 설정한다. a와 b 변수는 오른쪽과 왼쪽으로 이동하기 위한 변수로, 모두 초기값을 0으로 설정한다. row 변수와 col 변수의 경우 배열의 행과 열을 나타내는 변수이며, 모두 초기값은 0이다. 이어 배열을 좌우반전시킬 때 사용할 변수 swich를 초기값을 0으로 하여 선언한다. 마지막으로 배열을 채워나갈 방향을 결정하기 위한 변수 way를 초기값을 1로 하여 정의한다.

배열과 변수들의 정의가 끝나면 달팽이 모양으로 배열을 채우는 작업이 실시된다. 반복 변수 k의 초기값을 0으로 하여 5가 될 때까지 k의 값을 1씩 증가시키며 반복을 수행한다. 반복이 수행될때마다 index 변수의 값을 배열 C의 b행 a열에 저장한다. 배열에 값을 저장한 후 index 변수의 값을 1 감소시키고, 변수 a의 값은 way 변수의 값에 a를 더한 값이 된다.

첫 번째 반복이 모두 끝나면 변수 a의 값을 1 감소시키고, 다음 반복을 수행한다. 외부 반복의 경우, 반복 변수 i의 초기값을 4로 하여 0이 될 때까지 i의 값을 1씩 감소시키며 반복을 진행한다. 내부 반복의 경우 가장 먼저 반복 변수 j의 초기값을 0으로 하여 j가 변수 i의 값과 같아질 때까지 j의 값을 1씩 증가시키며 반복을 수행한다. 첫 번째 내부 반복에서 변수 b의 값은 b의 원래 값에 way 변수의 값을 더한 값이 되며, 배열 C의 b행 a열에는 index의 값이 들어간다. 배열에 인덱스값이 저장되었다면, index 변수의 값을 1 감소시킨다.

첫 번째 내부반복이 모두 끝나면 변수 way의 부호를 반전시키고 이어 배열에 원소를 채우기 위한 두 번째 반복을 수행한다. 반복 변수 k의 초기값을 0으로 하여 1씩 증가시키며 k의 값이 변수 i의 값과 같아질 때까지 반복을 수행한다. 반복을 수행할 때 변수 a의 값은 변수 a의 값에 변수 way의 값을 더한 값이 되며, 배열 C의 b행 a열에는 index 변수의 값이 저장된다. 배열에 값이 저장되었다면 index의 값은 1씩 감소한다.

배열 원소에 달팽이 모양대로 값을 부여하는 과정이 끝나면, 완성된 배열을 좌우로 반전시키는 과정을 수행한다. 먼저 외부 반복 변수 row의 초기값을 0으로 하여, row 변수의 값이 5가 될 때까지 1씩 증가시키며 반복을 수행한다. 이때 row 변수는 배열의 행을 의미한다. 반대로 내부 반복의 경우 배열의 열을 나타내는 col 변수가 반복 변수가 된다. col 변수의 경우 초기값은 0이고, 마찬가지로 1씩 증가하며 변수의 값이 3이 될 때까지 반복을 실행한다. 내부 반복에서는 swich 변수를 이용하여 배열 C의 row행 col열의 인덱스와 배열 C의 row행 4-col열의 인덱스값을 바꾼다.

배열의 좌우반전이 끝나면 마지막으로 이 배열을 좌측으로 90도 회전시키는 과정을 거친다. 이 또한 중첩 반복문이 사용되며, 외부 반복의 경우 반복변수 row의 초기값을 0으로 하여 5가 될 때까지 1씩 증가시키며 반복을 수행한다. 내부 반복의 경우 초기값이 0인 col 반복변수를 이용하는데, 이 변수가 5가 될 때까지 1씩 증가시키며 반복을 수행한다. 수행하는 반복의 내용은 최종 배열 TEMP의 4-col행 row열에 배열 C의 row행 col열의 인덱스를 저장하는 것이다.

배열의 회전까지 모두 마무리되었다면, 이어 완성된 배열을 출력하는 과정을 거친다. 완성된 배열 TEMP의 경우 중첩 반복문을 통해 출력하며, 행 우선으로 출력된다. 배열을 일정하게 출력하기 위해 한 열이 모두 출력된 이후에는 줄바꿈 문자를 넣어 문제의 사진과 같은 모양이 되도록 하였다. 완성된 배열을 모두 출력하고 나면 순서도는 마무리된다.

3-3. 프로그램 구현

```
Computer Algorithm Assingment 2_Array 3 (Global Scope)
1  #include <stdio.h>
2
3  int main()
4  {
5      int C[5][5];
6      int TEMP[5][5]; //배열의 좌측 회전 때 사용할 배열
7      int index = 25;
8      int a = 0; int b = 0; //오른쪽, 왼쪽으로 이동하기 위한 변수
9      int row = 0; int col = 0; //배열의 행과 열을 나타내는 변수
10     int i, k, j;
11     int swich = 0; //배열 좌우반전 시 사용할 스와핑 변수
12     int way = 1; //방향을 나타내는 방향변수
13
14
15     //달팽이 모양으로 배열 채우기
16     for (k = 0; k < 5; ++k)
17     {
18         C[b][a] = index;
19         index = index - 1;
20         a += way;
21     }
22
23     a = a - 1;
24
25     for (i = 4; i > 0; --i)
26     {
27         for (j = 0; j < i; ++j)
28         {
29             b += way;
30             C[b][a] = index;
31             index = index - 1;
32         }
33
34         way = way * -1; //변수 way의 부호를 바꾼다.
35
36         for (k = 0; k < i; ++k)
37         {
38             a += way;
39             C[b][a] = index;
40             index = index - 1;
41         }
42     }
43
44     //배열 좌우반전
45     for (row = 0; row < 5; row++)
46     {
47         for (col = 0; col < 3; col++)
48         {
49             swich = C[row][col];
50             C[row][col] = C[row][4 - col];
51             C[row][4 - col] = swich;
52         }
53     }
```

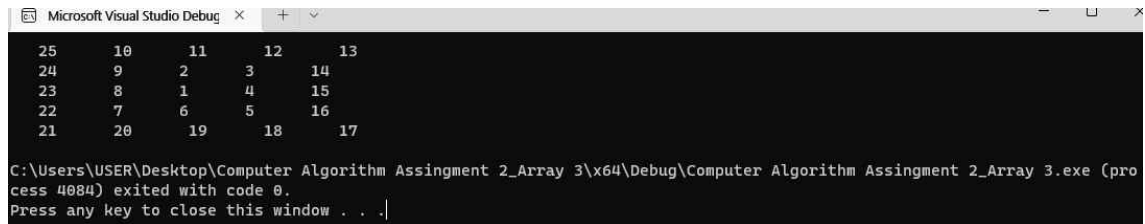
```

55
56 //배열 좌측으로 90도 회전
57 for (row = 0; row < 5; row++)
58 {
59     for (col = 0; col < 5; col++)
60     {
61         TEMP[4 - col][row] = C[row][col];
62     }
63 }
64
65
66 //완성된 배열 출력
67 for (row = 0; row < 5; row++)
68 {
69     for (col = 0; col < 5; col++)
70     {
71         printf("    %d    ", TEMP[row][col]);
72     }
73     printf("\n");
74 }
75
76
77 return 0;
78 }
79

```

+ 소스 코드는 보고서 파일과 함께 Lms에 참조

3-4. 실행 화면



The screenshot shows a Microsoft Visual Studio Debug console window. The output consists of a 5x5 grid of numbers followed by a message indicating the program has exited successfully.

25	10	11	12	13
24	9	2	3	14
23	8	1	4	15
22	7	6	5	16
21	20	19	18	17

C:\Users\USER\Desktop\Computer Algorithm Assingment 2_Array 3\x64\Debug\Computer Algorithm Assingment 2_Array 3.exe (process 4084) exited with code 0.
Press any key to close this window . . .|

< 끝 >