

# C언어 Term Project

## sudoku 만들기

조 원 : 1606212 김 계 림  
1651723 강 정 우  
1721935 정 보 근

# 목차

- ① 요구사항 분석
- ② 설계
- ③ 제작(코딩)
- ④ 테스트
- ⑤ 참고문헌
- ⑥ 총평
- ⑦ 사진

# 요구사항 분석

## ◆ 목표

9X9 스도쿠 만들기

## ◆ Requirement analysis

1. 0의 개수를 조사하여 스도쿠값 도출
2. 백트래킹 (1. 현재 자리 (x,y)에 1~9를 넣을 수 있나 확인해보고, 가능하면 그 자리에 숫자를 넣는다.

# 설계

## ◆필요한 함수 및 자료구조

1. If else문 – 선언한 함수가 맞거나 틀리면 return으로 반환하기위해

Ex)

```
if ((col + 1) < 9) return fillSudoku(puzzle, row, col + 1);  
else if ((row + 1) < 9) return fillSudoku(puzzle, row + 1, 0);  
else return 1;
```

# 설계

## ◆필요한 함수 및 자료구조

2. For문 – 선언 되어있는 배열에 숫자를 넣기 위해서

Ex)

```
for (i = 1; i<10; i++)  
{  
    for (j = 1; j<10; j++) printf("%d", (*(puzzle + i - 1) + j - 1));  
    printf("\n");  
}
```

# 설계

## ◆필요한 함수 및 자료구조

3. 함수와 포인터 – 메모리를 적게 쓰기위해서

Ex)

```
int isAvailable(int(*puzzle)[9], int row, int col, int num)
{
    return !(*(*puzzle + row) + col);
}
```

```
// arr[i] = *(arr+i) 응용
```

# 설계

## ◆필요한 함수 및 자료구조

4. fopen/fclose 함수 및 fprintf – 파일에 입출력하기위해서

Ex)

```
FILE *fp = fopen("practice.txt", "r");
```

```
FILE *fs = fopen("result.txt", "w");
```

```
fclose(fp);
```

```
fprintf(fs, "소요 시간  %d분: %f초\n", result_min, result_sec);
```

# 설계

## ◆필요한 함수 및 자료구조

5. fgets/fputs – 문서에서 문장단위로 받고 다시 문서에 문장 단위로 출력하기위해서

Ex)

```
fgets(buf, 10, fp);
```

```
fputs(buf, fs);
```

// 이것을 쓸때 문서로 숫자가 표현되는 방식은 아스키코드이므로 아스키코드 0x30(0)을 더하거나 빼야한다.



# 설계

◆필요한 함수 및 자료구조

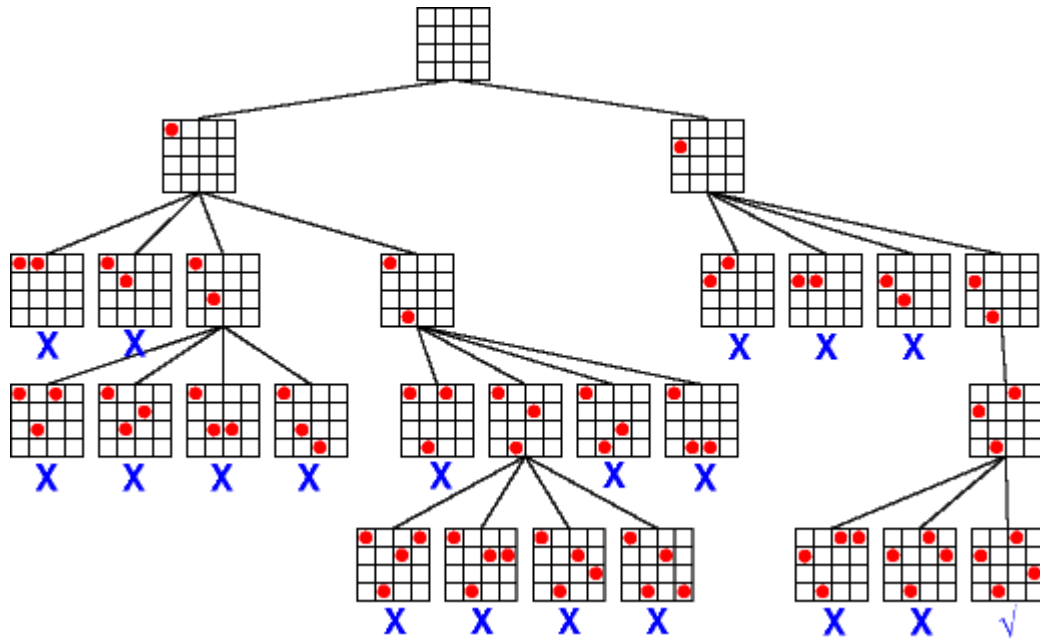
6. Gettickcount()- 실행시간을 구하기위해서  
EX)

```
int startTickCount = GetTickCount();
```

```
int currentTickCount;
```

```
// #include<windows.h> 해야함
```

# 설계



7. 백트래킹 알고리즘을 구현하기 위한 c코드

Ex)

```
if (*(puzzle + row) + i) == num) return 0;
```

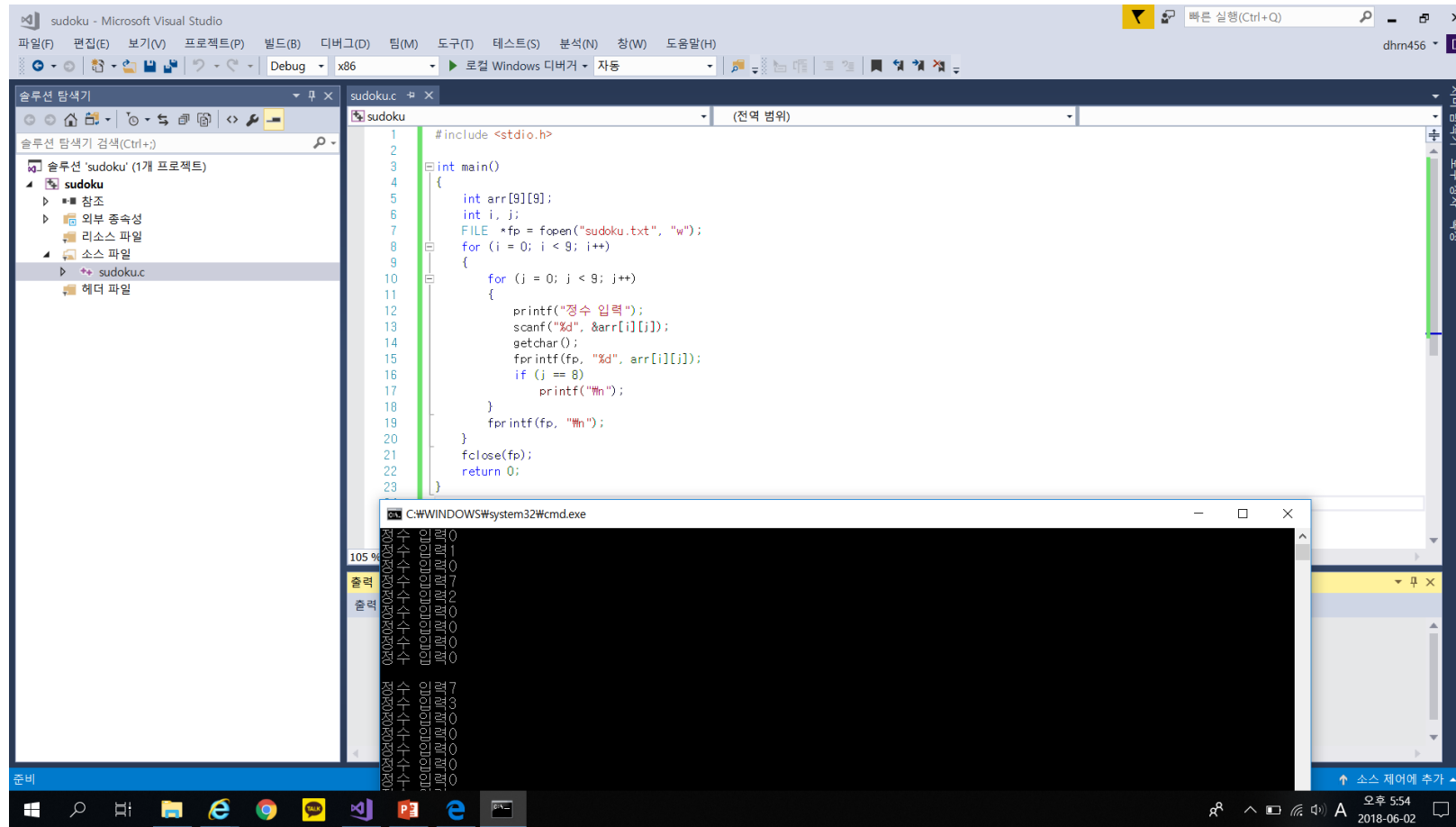
```
if (*(puzzle + i) + col) == num) return 0;
```

```
if (*(puzzle + rowStart + i % 3) + colStart + i / 3)  
== num) return 0;
```

# 테스트케이스

1. sudoku.txt - 파일 생성코드로 스도쿠를 만든다
2. result.txt – sudoku.txt로 부터 파일을 받아들여 스도쿠 문제를 풀어 result.txt를 만든다

# 제작(코딩결과)



The image shows a screenshot of the Microsoft Visual Studio IDE. The main window displays the source code for a C program named `sudoku.c`. The code is as follows:

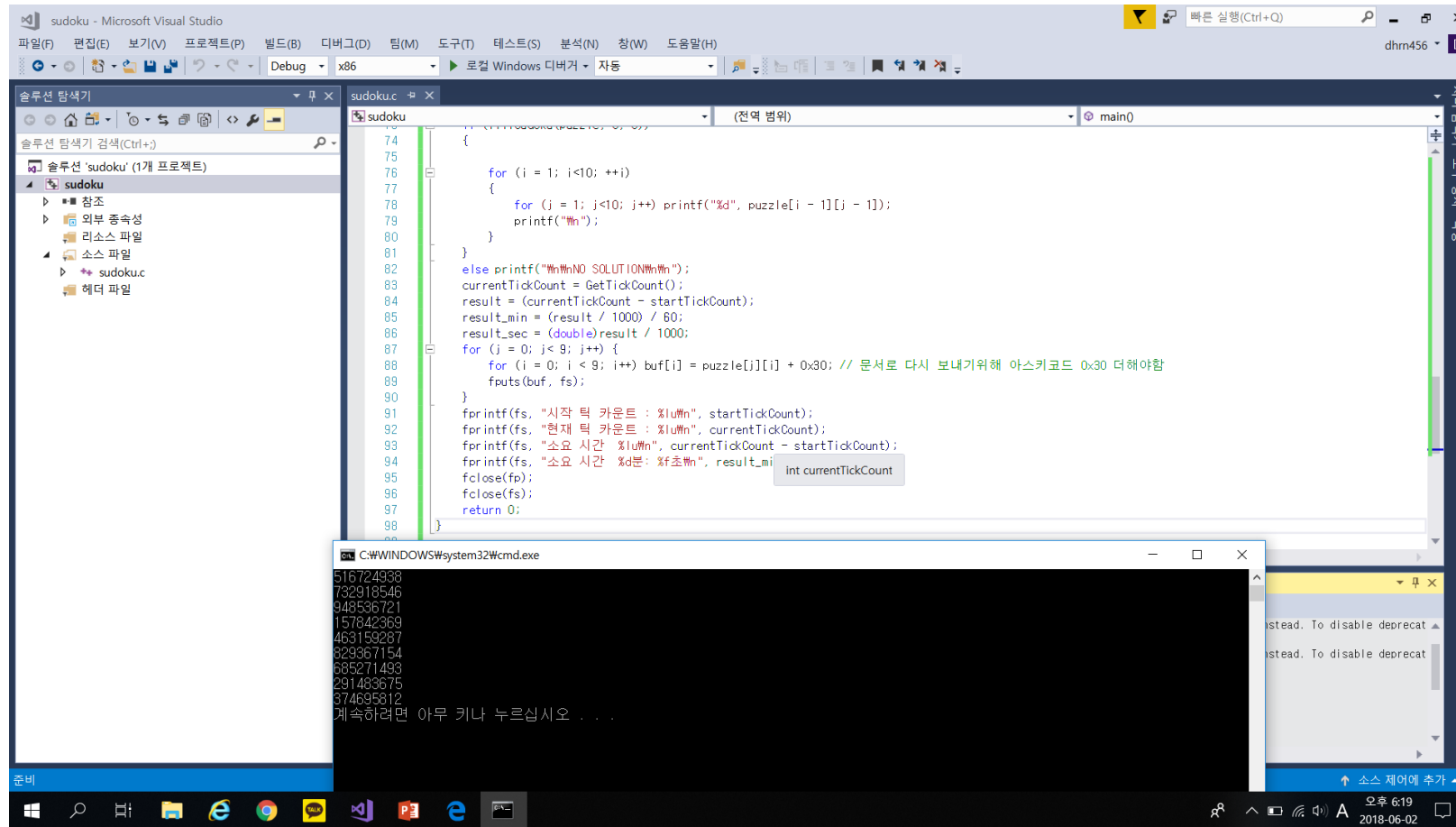
```
1 #include <stdio.h>
2
3 int main()
4 {
5     int arr[9][9];
6     int i, j;
7     FILE *fp = fopen("sudoku.txt", "w");
8     for (i = 0; i < 9; i++)
9     {
10         for (j = 0; j < 9; j++)
11         {
12             printf("정수 입력");
13             scanf("%d", &arr[i][j]);
14             getchar();
15             fprintf(fp, "%d", arr[i][j]);
16             if (j == 8)
17                 printf("\n");
18         }
19         fprintf(fp, "\n");
20     }
21     fclose(fp);
22     return 0;
23 }
```

Below the code editor, a command prompt window titled `C:\WINDOWS\system32\cmd.exe` is open, showing the output of the program's execution. The output consists of a 9x9 grid of numbers, with each row containing a sequence of numbers that appear to be a permutation of 0-9, separated by spaces. The output is as follows:

```
0 1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9 0
2 3 4 5 6 7 8 9 0 1
3 4 5 6 7 8 9 0 1 2
4 5 6 7 8 9 0 1 2 3
5 6 7 8 9 0 1 2 3 4
6 7 8 9 0 1 2 3 4 5
7 8 9 0 1 2 3 4 5 6
8 9 0 1 2 3 4 5 6 7
9 0 1 2 3 4 5 6 7 8
```

The Visual Studio interface also shows the 'Solution Explorer' on the left, indicating the project structure, and the 'Output' window at the bottom right, which is currently empty.

# 제작(코딩결과)



```
sudoku - Microsoft Visual Studio
파일(F) 편집(E) 보기(V) 프로젝트(P) 빌드(B) 디버그(D) 팀(M) 도구(T) 테스트(S) 분석(N) 창(W) 도움말(H)
Debug x86 로컬 Windows 디버거 자동 dhm456 D

솔루션 탐색기
솔루션 탐색기 검색(Ctrl+;)
솔루션 'sudoku' (1개 프로젝트)
  sudoku
    참조
    외부 종속성
    리소스 파일
    소스 파일
      sudoku.c
    헤더 파일

sudoku.c
74
75
76     for (i = 1; i<10; ++i)
77     {
78         for (j = 1; j<10; j++) printf("%d", puzzle[i - 1][j - 1]);
79         printf("\n");
80     }
81
82     else printf("\n\nNO SOLUTION\n\n");
83     currentTickCount = GetTickCount();
84     result = (currentTickCount - startTickCount);
85     result_min = (result / 1000) / 60;
86     result_sec = (double)result / 1000;
87     for (j = 0; j< 9; j++) {
88         for (i = 0; i< 9; i++) buf[i] = puzzle[j][i] + 0x30; // 문서로 다시 보내기 위해 아스키코드 0x30 더해야함
89         fputs(buf, fs);
90     }
91     fprintf(fs, "시작 틱 카운트 : %lu\n", startTickCount);
92     fprintf(fs, "현재 틱 카운트 : %lu\n", currentTickCount);
93     fprintf(fs, "소요 시간 %lu\n", currentTickCount - startTickCount);
94     fprintf(fs, "소요 시간 %d분: %f초\n", result_min, result_sec);
95     fclose(fs);
96     return 0;
97
98 }
```

C:\WINDOWS\system32\cmd.exe

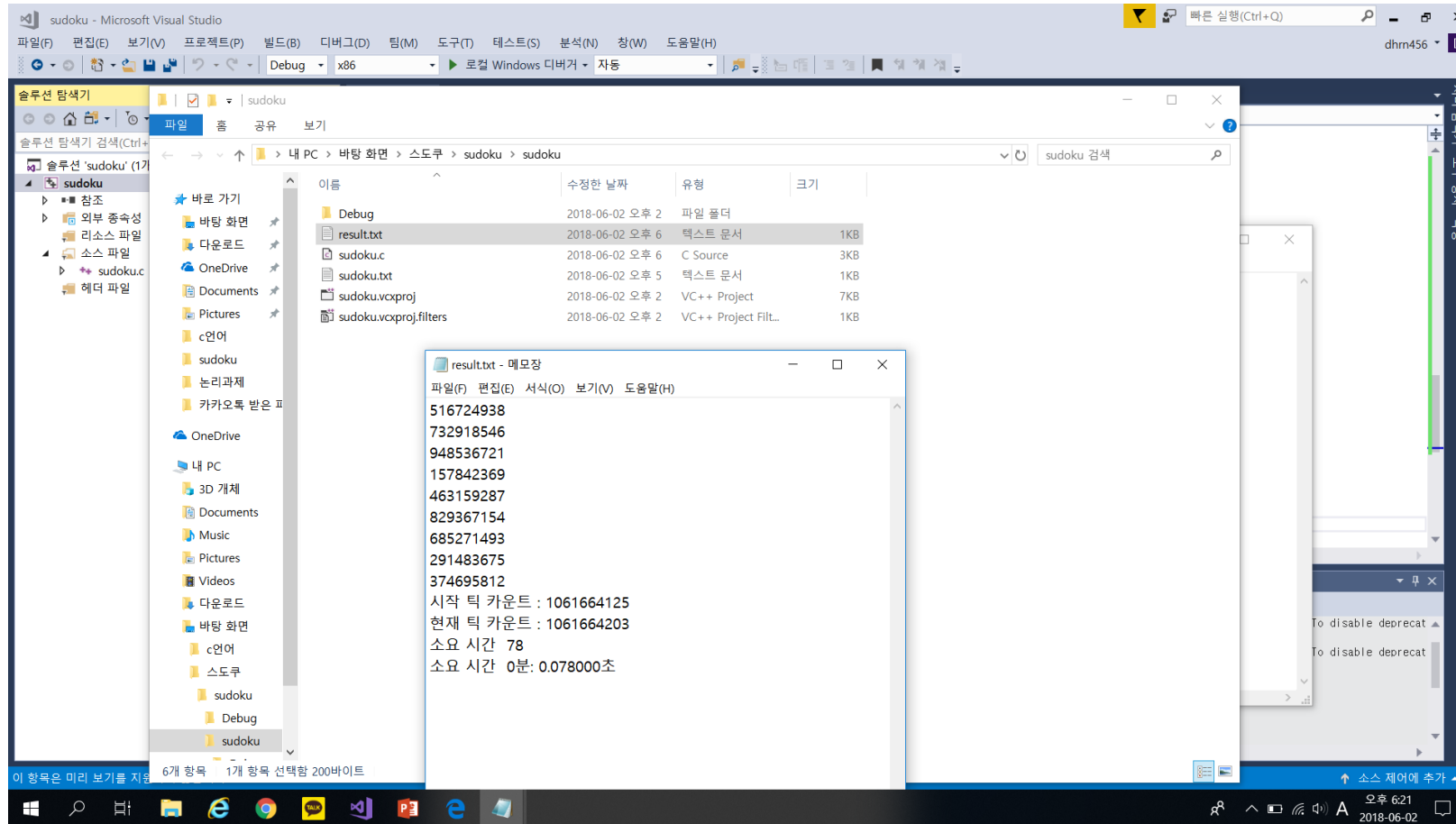
```
516724938
732918546
948536721
157842369
463159287
829367154
685271493
291463675
374695812
계속하려면 아무 키나 누르십시오 . . .
```

준비

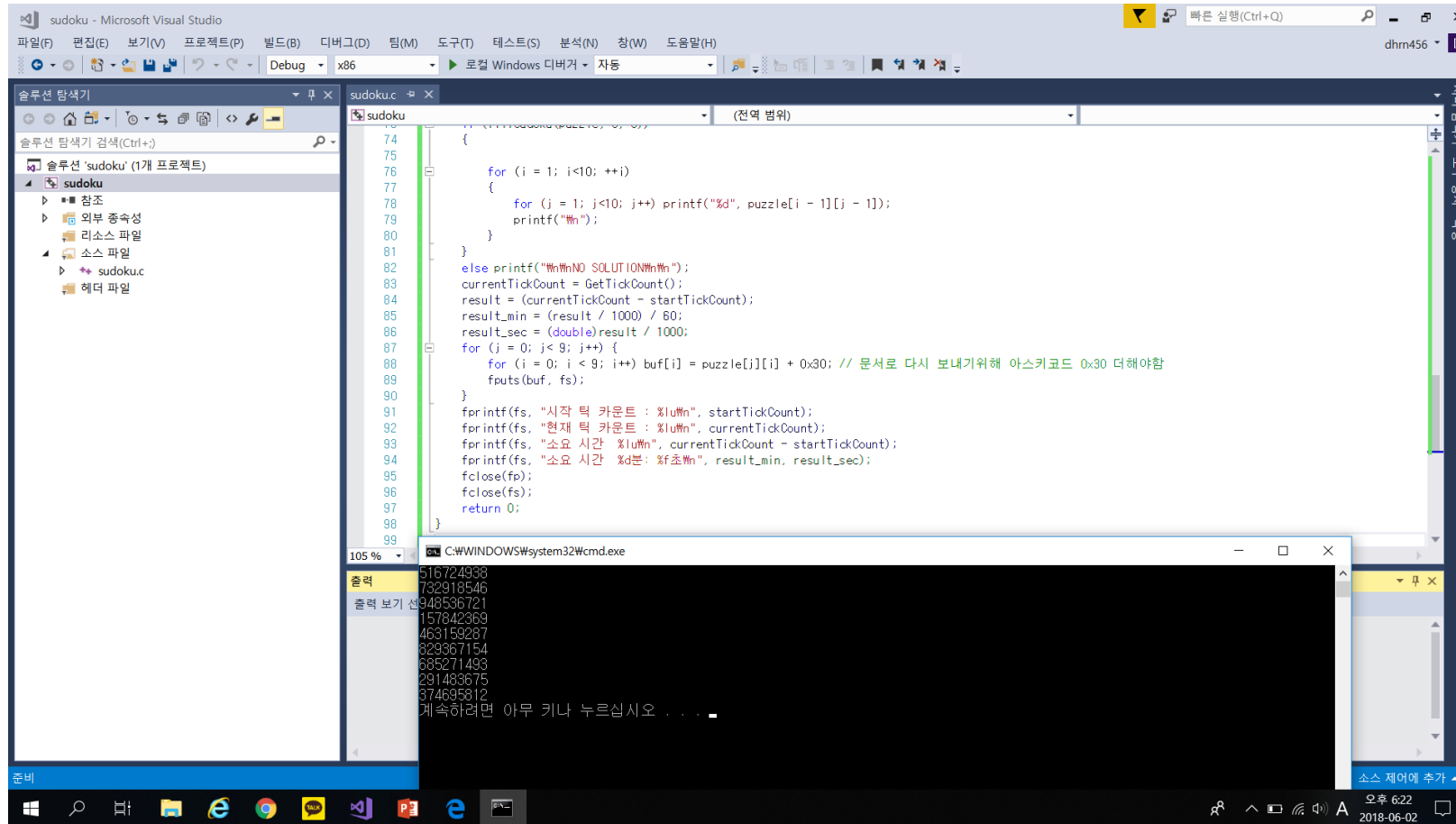
소스 제어에 추가

오후 6:19  
2018-06-02

# 제작(코딩결과) 문서생성



# 테스트 예제 1 번



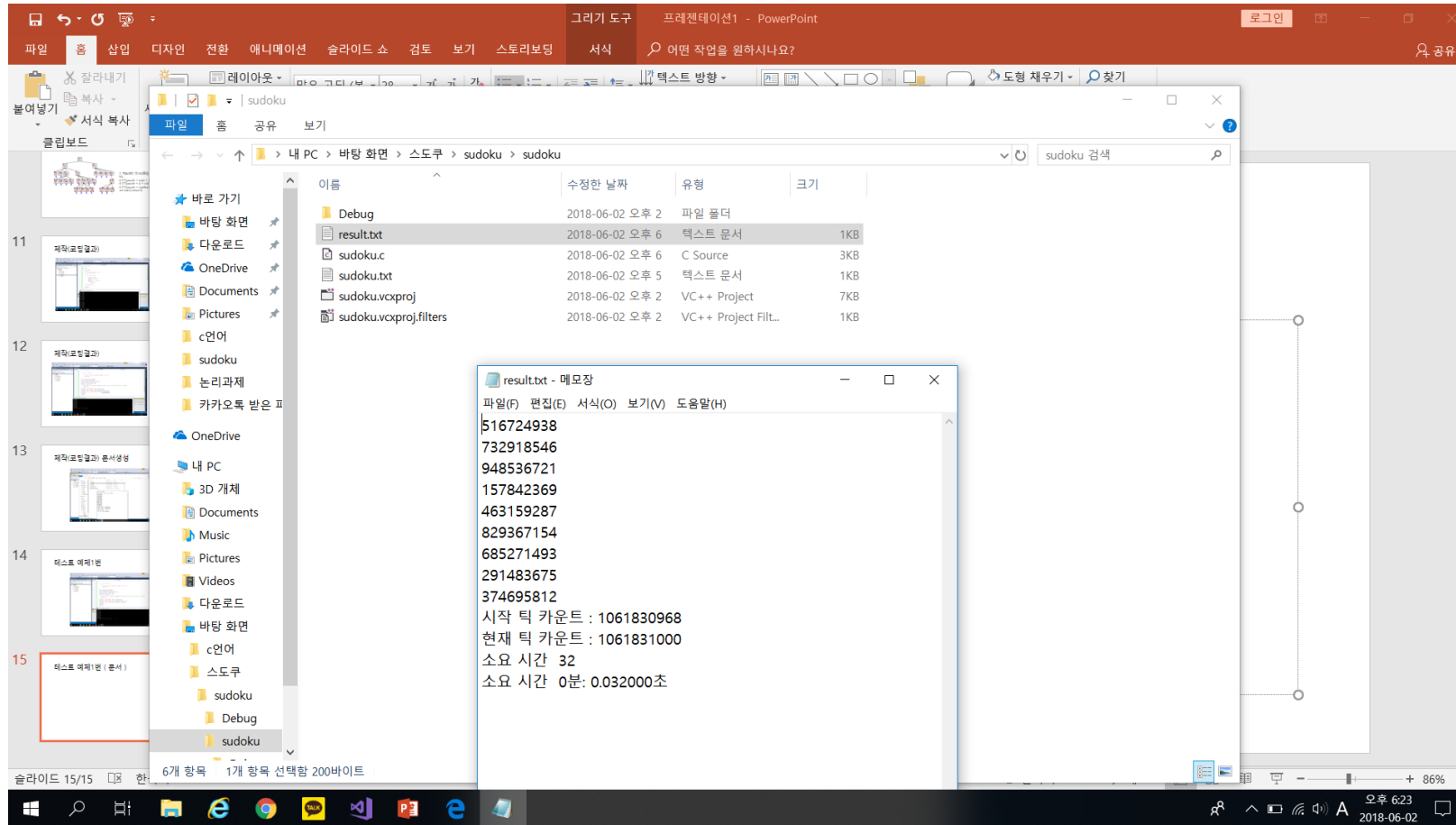
The screenshot displays the Microsoft Visual Studio IDE with a C program named `sudoku.c` open. The program is designed to solve a 10x10 Sudoku puzzle. The code includes a function `findSolution` that recursively searches for a solution, a `main` function that initializes the puzzle and calls `findSolution`, and a `printSolution` function that prints the solution. The program also includes timing logic to measure the execution time.

```
74 void findSolution(int puzzle[10][10], int row, int col)
75 {
76     for (i = 1; i < 10; ++i)
77     {
78         for (j = 1; j < 10; j++) printf("%d", puzzle[i - 1][j - 1]);
79         printf("\n");
80     }
81 }
82 else printf("\n\nNO SOLUTION\n\n");
83 currentTickCount = GetTickCount();
84 result = (currentTickCount - startTickCount);
85 result_min = (result / 1000) / 60;
86 result_sec = (double)result / 1000;
87 for (j = 0; j < 9; j++) {
88     for (i = 0; i < 9; i++) buf[i] = puzzle[j][i] + 0x30; // 문서로 다시 보내기 위해 아스키코드 0x30 더해야함
89     fputs(buf, fs);
90 }
91 fprintf(fs, "시작 틱 카운트 : %lu\n", startTickCount);
92 fprintf(fs, "현재 틱 카운트 : %lu\n", currentTickCount);
93 fprintf(fs, "소요 시간 : %lu\n", currentTickCount - startTickCount);
94 fprintf(fs, "소요 시간 : %d분 : %f초\n", result_min, result_sec);
95 fclose(fp);
96 fclose(fs);
97 return 0;
98 }
99 }
```

The console window shows the output of the program, which includes the solution to the puzzle and the execution time.

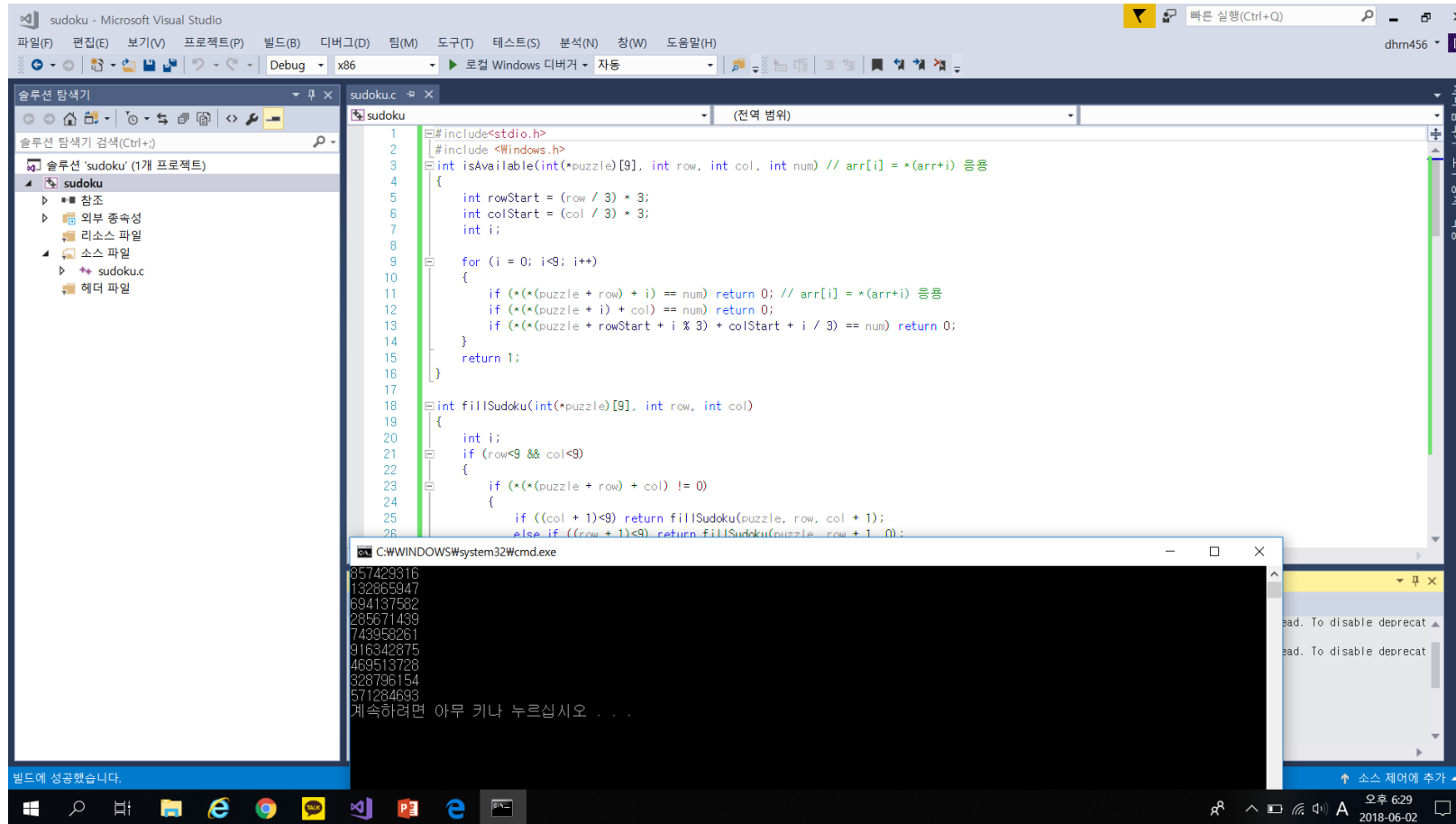
```
516724938
732918546
348536721
157842369
463159287
829367154
685271493
291483675
374695812
계속하려면 아무 키나 누르십시오 . . .
```

# 테스트 예제 1번 ( 문서 )





# 테스트 예제 2번



The screenshot shows the Microsoft Visual Studio IDE with a C program for solving a Sudoku puzzle. The code is as follows:

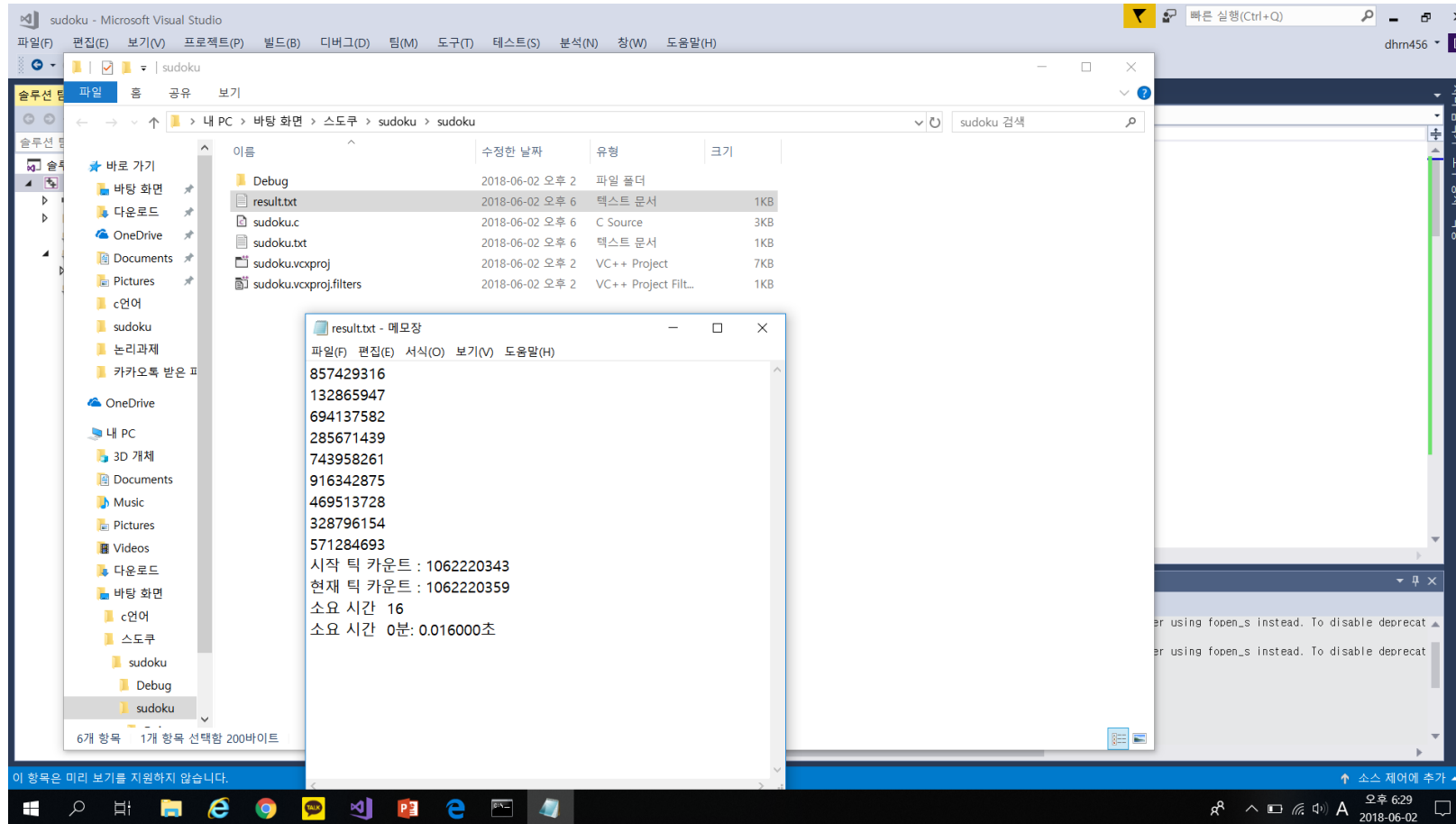
```
1 #include<stdio.h>
2 #include <Windows.h>
3 int isAvailable(int(*puzzle)[9], int row, int col, int num) // arr[i] = *(arr+i) 응용
4 {
5     int rowStart = (row / 3) * 3;
6     int colStart = (col / 3) * 3;
7     int i;
8
9     for (i = 0; i<9; i++)
10     {
11         if (*(puzzle + row) + i == num) return 0; // arr[i] = *(arr+i) 응용
12         if (*(puzzle + i) + col == num) return 0;
13         if (*(puzzle + rowStart + i % 3) + colStart + i / 3 == num) return 0;
14     }
15     return 1;
16 }
17
18 int fillSudoku(int(*puzzle)[9], int row, int col)
19 {
20     int i;
21     if (row<9 && col<9)
22     {
23         if (*(puzzle + row) + col != 0)
24         {
25             if ((col + 1)<9) return fillSudoku(puzzle, row, col + 1);
26             else if ((row + 1)<9) return fillSudoku(puzzle, row + 1, 0);
27         }
28     }
```

In the foreground, a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe" displays the following output:

```
857429316
132865947
694137582
285671439
743958261
916342875
469513728
328796154
571284693
계속하려면 아무 키나 누르십시오 . . .
```

The status bar at the bottom of Visual Studio indicates "빌드에 성공했습니다." (Build succeeded).

# 테스트 예제2번( 문서 )



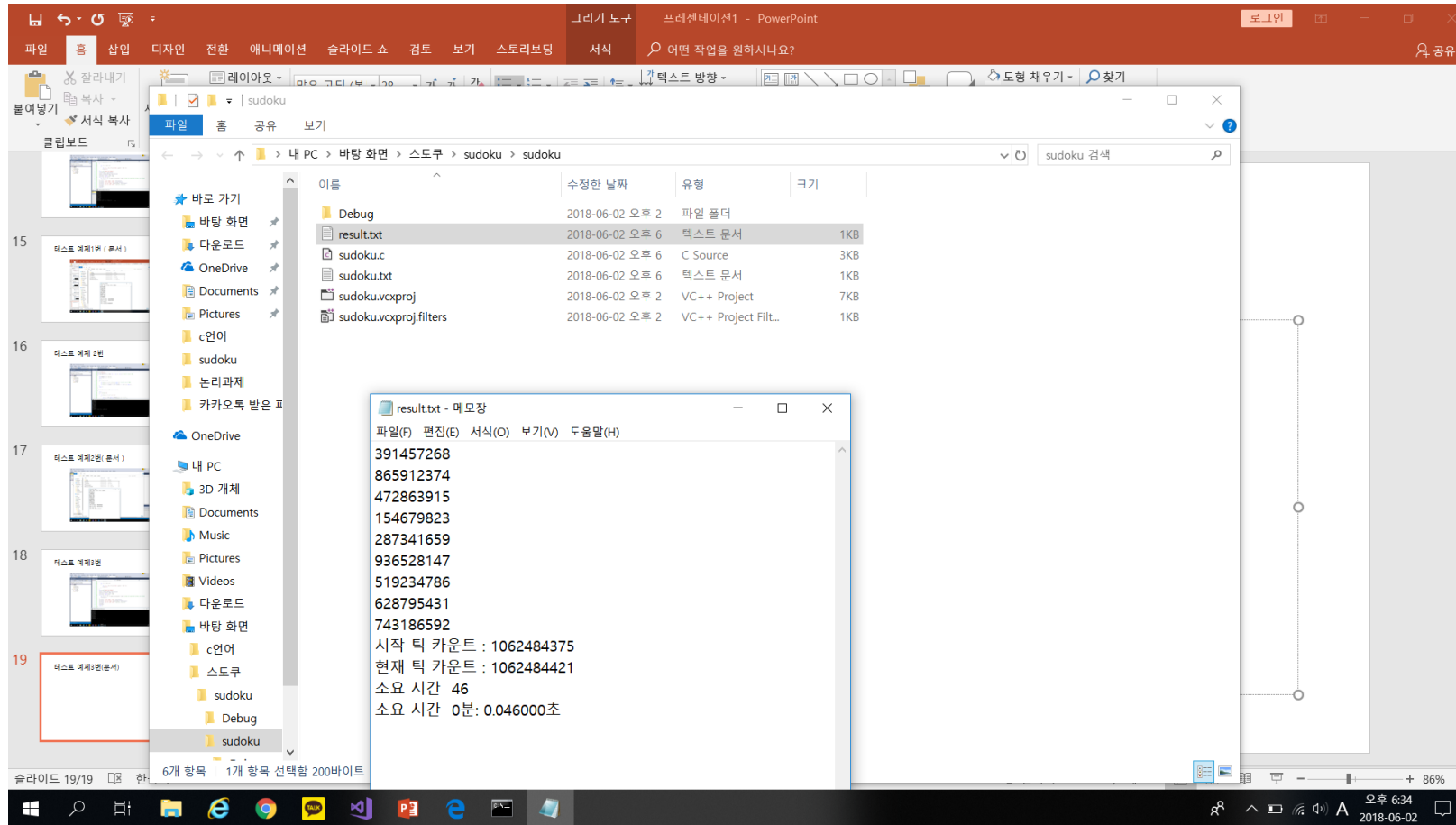
# 테스트 예제 3번

The image shows a screenshot of the Microsoft Visual Studio IDE. The main window displays the source code for a C program named 'sudoku.c'. The code is for solving a Sudoku puzzle. It includes a function 'find\_solution' that recursively finds a solution. The main function 'main' initializes the puzzle, calls 'find\_solution', and prints the solution and timing information. A terminal window at the bottom shows the output of the program, displaying a list of numbers and a message to continue.

```
74 // ...
75
76
77 for (i = 1; i < 10; ++i)
78 {
79     for (j = 1; j < 10; j++) printf("%d", puzzle[i - 1][j - 1]);
80     printf("\n");
81 }
82
83 else printf("\n\nNO SOLUTION\n\n");
84 currentTickCount = GetTickCount();
85 result = (currentTickCount - startTickCount);
86 result_min = (result / 1000) / 60;
87 result_sec = (double)result / 1000;
88 for (j = 0; j < 9; j++) {
89     for (i = 0; i < 9; i++) buf[i] = puzzle[j][i] + 0x30; // 문서로 다시 보내기 위해 아스키코드 0x30 더해야함
90     fputs(buf, fs);
91 }
92 printf(fs, "시작 틱 카운트 : %lu\n", startTickCount);
93 printf(fs, "현재 틱 카운트 : %lu\n", currentTickCount);
94 printf(fs, "소요 시간 : %lu\n", currentTickCount - startTickCount);
95 printf(fs, "소요 시간 : %d분: %f초\n", result_min, result_sec);
96 fclose(fp);
97 fclose(fs);
98 return 0;
99 }
```

391457268  
865912374  
472863915  
154679823  
287341659  
936528147  
519234786  
628795431  
743186592  
계속하려면 아무 키나 누르십시오 . . .

# 테스트 예제3번(문서)



# 참고문헌

[https://codereview.stackexchange.com/questions/37430/sudoku-solver-in-c?utm\\_medium=organic&utm\\_source=google\\_rich\\_qa&utm\\_campaign=google\\_rich\\_qa](https://codereview.stackexchange.com/questions/37430/sudoku-solver-in-c?utm_medium=organic&utm_source=google_rich_qa&utm_campaign=google_rich_qa) 스도쿠 알고리즘

<http://blog.naver.com/PostView.nhn?blogId=highkrs&logNo=220195938896&categoryNo=16&parentCategoryNo=0&viewDate=&currentPage=1&postListTopCurrentPage=1&from=postView>  
fgets fputs

# 과제 수행 총평

한 학기동안 C언어를 공부하면서 처음에는 막막 할 것만 같던 스도쿠가 조원과 함께 머리를 맞대며 의논을 하고 토의를 하며 스도쿠소스를 최종적으로 만들게 되었을 때 자신감과 C언어 뿐 아니라 다른 알고리즘도 이해하는 계기가 되었습니다.