Project #1 - Sudoku

Meng-Hsun Tsai
CSIE, NCKU

Presented by Jingfei

2016.03.22

An easy question…

Bonus +0.5

What is Sudoku?

- Play with digits from 1 to 9 in a 9x9 grid

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

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

Solve it! !

- It should contain all of the digits from 1 to 9 in
  1. Each row
  2. Each column
  3. Each of the nine 3x3 sub-grids

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

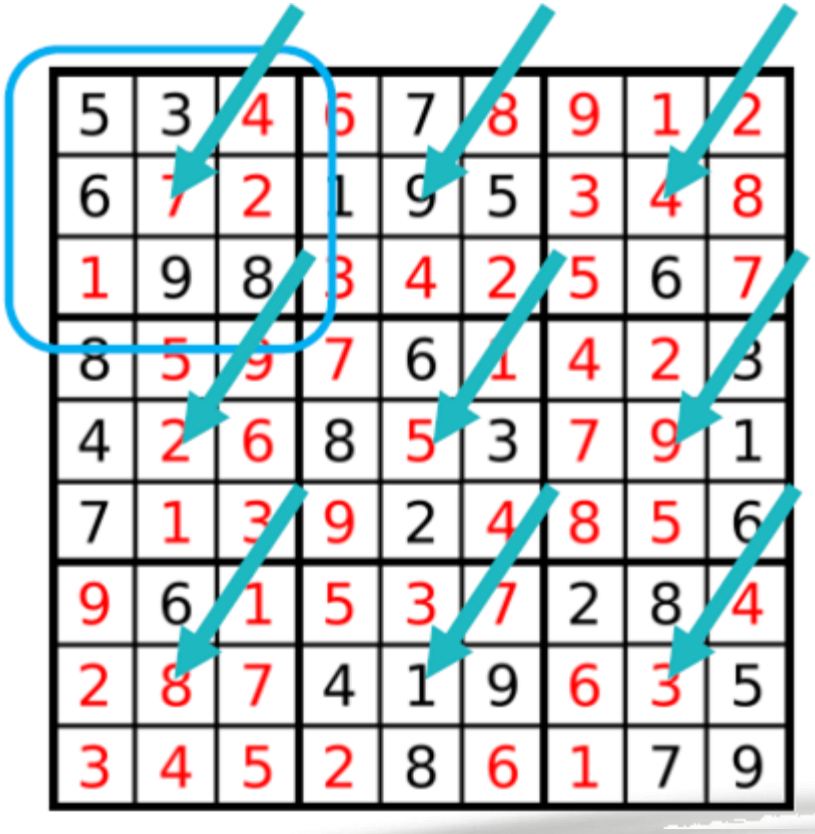
Solve it! !

- It should contain all of the digits from 1 to 9 in
  1. Each row
  2. Each column
  3. Each of the nine 3x3 sub-grids

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

Solve it! !

- It should contain all of the digits from 1 to 9 in
  1. Each row
  2. Each column
  3. Each of the nine 3x3 sub-grids



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

After warm up…

Bonus +0.5

How to make a Sudoku board?

- Put digits 1 to 9 and blank in a board randomly

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

How to make a solvable Sudoku board?

- I do not know, either. Google it !

How to make a solvable Sudoku board?

- How about… if we have a board with solution.

Transform it!

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

How to make a solvable Sudoku board?

- How about… if we have a board with solution.
  - Rotate the grid 90, 180, or 270 degrees. (clockwise, counter-clockwise)
  - Flip the board horizontally or vertically
  - Or even flip by diagonal axis
  - Permute the rows 1-3, 4-6, or 7-9.
  - Permute the columns 1-3, 4-6, or 7-9.
  - Permute the 3x9 blocks of rows.
  - Permute the 9x3 blocks of columns.
  - Permute the digits.

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

How to make a solvable Sudoku board?

- How about… if we have a board with solution. (We will use …)
  - Rotate the grid 90, 180, or 270 degrees. (clockwise, counter-clockwise)
  - Flip the board horizontally or vertically
  - Or even flip by diagonal axis
  - Permute the rows 1-3, 4-6, or 7-9.
  - Permute the columns 1-3, 4-6, or 7-9.
  - Permute the 3x9 blocks of rows.
  - Permute the 9x3 blocks of columns.
  - Permute the digits.

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

Our tasks in this project

at least 9 public `void` functions

Task 1: give question

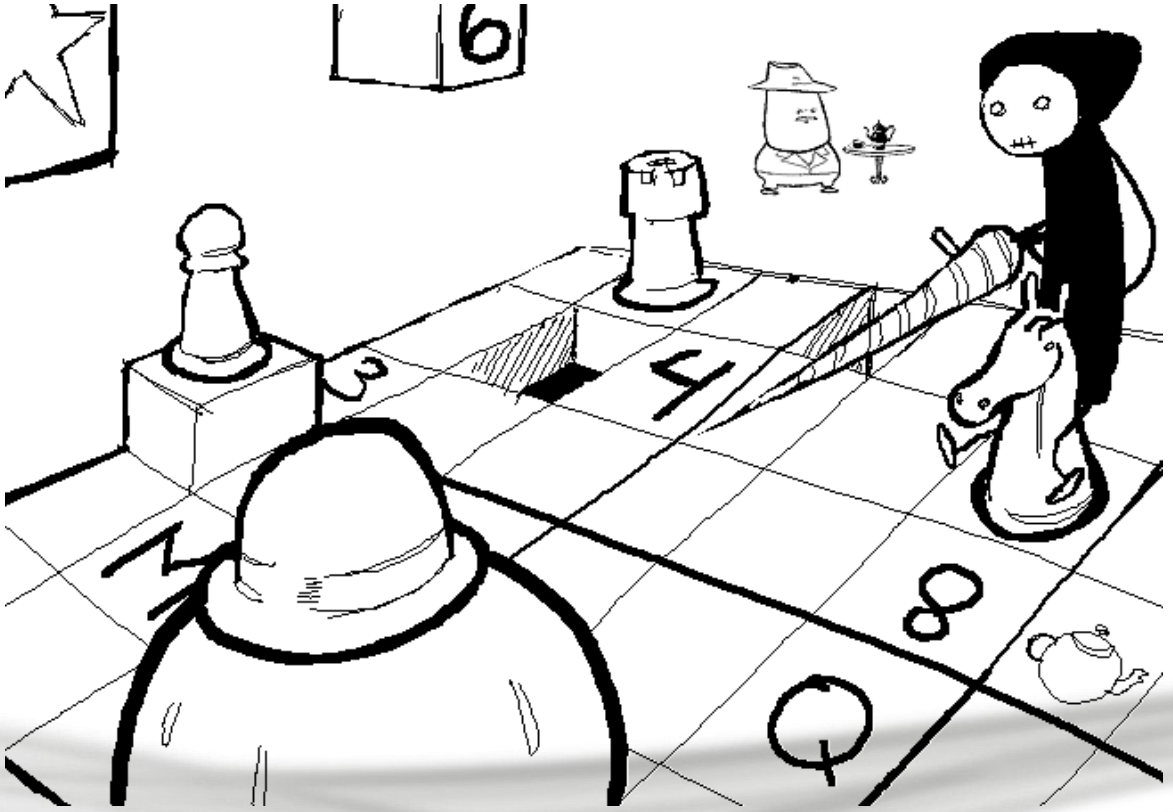
1. **`giveQuestion()`**: Create your own Sudoku board.
   - Use ‘0’ character to represent the blanks
   - Any two digits are separated by a space

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

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

Task 2-1: read in Sudoku board

2. **`readIn()`**: Read in Sudoku board.
   - clue: cin, scanf, 81 ditgits …



Task 2-2: judge and solve it !

3. **`solve()`**: Judge if the board read in by `readIn()` function is solvable; then solve it, and print it out.

- Unsolvable: output a single character '0'
- Exactly one solution: output a single character '1' in the first line. The next 9 lines are the solution, for example:

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

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

- More than one solution: output a single character '2'

Task 3-1: transform

4. Use following functions to transform the board
   - **`changeNum(int a, int b)`**: Exchange number a and b in the board. (1 <= a, b <= 9)
   - **`changeRow(int a, int b)`**: Exchange row set a and b in the board, each row set include three continuous rows. (0 <= a, b <= 2)
     For example, if a=0 and b=1, you should exchange the whole first three rows (row 0~2) and second three rows (row 3~5).
     That is: row0 ↔ row3, row1 ↔ row4, row2 ↔ row5.
   - **`changeCol(int a,int b)`**: Concept is the same as ChangeRow. This time we exchange columns. (0 <= a, b <= 2)
   - **`rotate(int n)`**: Rotate the board 90 degrees n times in clockwise direction. (0 <= n <= 100)
   - **`flip(int n)`**: If n equals to 0, flip the board vertically. Otherwise, flip it horizontally. (0 <= n <= 1)

Task 3-2: transform

5. **`transform()`** function:

- Use data read in by `readIn()` function and any function in the last page inside `transform()` function, then print it out.

```
void Sudoku::transform(){
    readIn();
    change();
    printOut(false);
}

void Sudoku::change(){
    srand(time(NULL));
    changeNum(rand()%sudokuNum+1, rand()%sudokuNum+1);
    changeRow(rand()%3, rand()%3);
    changeCol(rand()%3, rand()%3);
    rotate(rand()%101);
    flip(rand()%2);
}
```

```
void Sudoku::printOut(bool isAns){
    int i;
    if(!isAns)
        for(i=0; i<sudokuSize; ++i)
            printf("%d%c",map[i], (i+1)%9==0?'\n':' ');
    else
        for(i=0; i<sudokuSize; ++i)
            printf("%d%c",ans[i], (i+1)%9==0?'\n':' ');
}
```

code author: zeroplusone

Conclusion: at least 9 public `void` functions

1. **`giveQuestion()`:** no input / output 81 digits
2. **`readIn()`:** input 81 digits / no output
3. **`solve()`:** no input (use the one in `readIn()`) / output your answer
4. **`changeNum(int a, int b)`:** no input / no output
5. **`changeRow(int a, int b)`:** no input / no output
6. **`changeCol(int a,int b)`:** no input / no output
7. **`rotate(int n)`:** no input / no output
8. **`flip(int n)`:** no input / no output
9. **`transform()`:** no input (use the one in `readIn()`) / output 81 digits

Bonus I

Using `makefile`

5 points bonus

Example

- I will use these three functions to check your makefile:

```
int main(){
  Sudoku ss;
  ss.giveQuestion();
  return 0;
}
```

giveQuestion.cpp

```
int main(){
  Sudoku ss;
  ss.readIn();
  ss.solve();
  return 0;
}
```

solve.cpp

```
int main(){
  Sudoku ss;
  ss.readIn();
  ss.transform();
  return 0;
}
```

transform.cpp

You can write a Makefile to make compile easier

```
all: Sudoku.o giveQuestion.cpp solve.cpp transform.cpp
  g++ -o giveQuestion giveQuestion.cpp Sudoku.o
  g++ -o solve solve.cpp Sudoku.o
  g++ -o transform transform.cpp Sudoku.o

Sudoku.o: Sudoku.cpp Sudoku.h
  g++ -c Sudoku.cpp -o Sudoku.o
```

Then type make in terminal ☺

Bonus II

Redirection

No point~

I don’t want to type 81 digits every time…

```
int main(){
  Sudoku ss;
  ss.readIn();
  ss.solve();
  return 0;
}
```

solve.cpp

Use redirection

Live DEMO

Bonus III

A website to check your code

0~8 points bonus

Sudoku tournament website

- Use this website to check your code: http://judge.imslab.org/
- Here are the statements:
  - Accepted: your code is correct, congratulations
  - Wrong Answer
  - Compile Error
  - Presentation Error
  - Time limited exceed: Exceed the time limit, 30 seconds.
  - Error: there is something wrong in this process, I hope it will not happen
- Please do not submit your code again when your another code is in "pending" statement. (Otherwise, you will get Error.)
- If there is any problem or question, please post your question here: http://moodle.ncku.edu.tw/mod/forum/view.php?id=484743

Sudoku tournament contest

- During 4/5 to 4/11, we will have two version of contest: Basic and Advanced
  - Basic:
    - You will need to solve your opponent's Sudoku board and give him a board to solve it.
    - The winner is the faster.
    - Functions used: giveQuestion(), readIn(), solve()
  - Advanced:
    - Two board need to be solved: one is your opponent's, the another is yours; However, the second one is transformed by your opponent.
    - It means you need to give your opponent a board to solve, transform the board he gave you and return to him.
    - The time during two solving process will be added up, and the winner is the faster.
    - Functions used: giveQuestion(), readIn(), solve(), transform()

Sudoku tournament contest

- You can choose only one of the two platforms.
- 5 challenge times per day.
- You can not challenge others when your code is not correct.
- Score bonus:
  - After contest, people in the 1st to 5th place will get 5 points in basic version and 8 points in advanced version.
  - People in the 6th to 10th place will get 4 points in basic version and 6 points in advanced version, and so on…
  - Until the 25th place in basic version and 20th place in advanced version.
- Be a White Hat instead of Black Hat, please…

The last but not the least

Requirements

- All input/output is stdin/stdout.
- Hand in two files `Sudoku.h` and `Sudoku.cpp`
  - Makefile is bonus, if you want to get it, there are four additional files: `giveQuestion.cpp, solve.cpp, transform.cpp, Makefile`
- In the class `Sudoku`, at least 9 public functions: `giveQuestion, readIn, solve, changeNum, changeRow, changeCol, rotate, flip, transform` should be defined.
- You need to consider both correctness and efficiency.

Evaluation

- You should upload source code to your github repository “pd2-sudoku” before 2016/04/03 23:59.
- Grading Policy
  - Correctness 80 points
  - Efficiency (speed) 20 points
  - Bonus I: Makefile 5 points
  - Bonus III: Sudoku Tournament 0~8 points (04/05 to 04/11)

Q&A

Thank you for listening