1.

tenth > postTest1.cpp > isSafe(int [GRID_SIZE][GRID_SIZE], int, int, int)

```cpp
#include <iostream>
using namespace std;

// Ukuran grid Sudoku
const int GRID_SIZE = 9;

// Fungsi untuk mencetak grid Sudoku
void printGrid(int grid[GRID_SIZE][GRID_SIZE]) {
    for (int row = 0; row < GRID_SIZE; row++) {
        for (int col = 0; col < GRID_SIZE; col++) {
            cout << grid[row][col] << " ";
        }
        cout << endl;
    }
}

// Fungsi untuk memeriksa apakah angka dapat ditempatkan di sel tertentu
bool isSafe(int grid[GRID_SIZE][GRID_SIZE], int row, int col, int num) {
    // Cek apakah angka sudah ada di baris
    for (int x = 0; x < GRID_SIZE; x++) {
        if (grid[row][x] == num) {
            return false;
        }
    }

    // Cek apakah angka sudah ada di kolom
    for (int x = 0; x < GRID_SIZE; x++) {
        if (grid[x][col] == num) {
            return false;
        }
    }

    // Cek apakah angka sudah ada di subgrid 3x3
    int startRow = row - row % 3;
    int startCol = col - col % 3;
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            if (grid[i + startRow][j + startCol] == num) {
                return false;
            }
        }
    }

    return true;
}
```

```
// Fungsi untuk menyelesaikan Sudoku menggunakan algoritma backtracking
bool solveSudoku(int grid[GRID_SIZE][GRID_SIZE]) {
    int row, col;
    bool emptyCell = false;

    // Temukan sel kosong
    for (row = 0; row < GRID_SIZE; row++) {
        for (col = 0; col < GRID_SIZE; col++) {
            if (grid[row][col] == 0) {
                emptyCell = true;
                break;
            }
        }
        if (emptyCell) break;
    }

    // Jika tidak ada sel kosong, Sudoku selesai
    if (!emptyCell) return true;

    // Coba angka dari 1 hingga 9
    for (int num = 1; num <= 9; num++) {
        if (isSafe(grid, row, col, num)) {
            grid[row][col] = num;

            // Rekursif untuk menyelesaikan grid
            if (solveSudoku(grid)) {
                return true;
            }

            // Jika gagal, kembalikan ke nilai awal (backtracking)
            grid[row][col] = 0;
        }
    }

    return false; // Tidak ada solusi
}
```

```cpp
int main() {
    // Masukan grid Sudoku
    int grid[GRID_SIZE][GRID_SIZE] = {
        {5, 3, 0, 0, 7, 0, 0, 0, 0},
        {6, 0, 0, 1, 9, 5, 0, 0, 0},
        {0, 9, 8, 0, 0, 0, 0, 6, 0},
        {8, 0, 0, 0, 6, 0, 0, 0, 3},
        {4, 0, 0, 8, 0, 3, 0, 0, 1},
        {7, 0, 0, 0, 2, 0, 0, 0, 6},
        {0, 6, 0, 0, 0, 0, 2, 8, 0},
        {0, 0, 0, 4, 1, 9, 0, 0, 5},
        {0, 0, 0, 0, 8, 0, 0, 7, 9}
    };

    if (solveSudoku(grid)) {
        cout << "Solved Sudoku Grid:" << endl;
        printGrid(grid);
    } else {
        cout << "No solution exists for the given Sudoku puzzle." << endl;
    }

    return 0;
}
```

Berikut adalah outputnya:

```
--interpreter=mi' ;5c74ab1e-6a08-4265-b511-ec3eee1d185fSolved Sudoku Grid:
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
```

2.

tenth > postTest2.cpp > main()

```cpp
#include <iostream>
#include <vector>
#include <cstdlib>
#include <ctime>
#include <algorithm> // Untuk std::shuffle
#include <random>    // Untuk random engine

using namespace std;

const int BOARD_SIZE = 8;
typedef vector<vector<char>> Board;

// Inisialisasi papan catur dengan bidak
void initializeBoard(Board &board) {
    board = {
        {'r', 'n', 'b', 'q', 'k', 'b', 'n', 'r'},
        {'p', 'p', 'p', 'p', 'p', 'p', 'p', 'p'},
        {'.', '.', '.', '.', '.', '.', '.', '.'},
        {'.', '.', '.', '.', '.', '.', '.', '.'},
        {'.', '.', '.', '.', '.', '.', '.', '.'},
        {'.', '.', '.', '.', '.', '.', '.', '.'},
        {'P', 'P', 'P', 'P', 'P', 'P', 'P', 'P'},
        {'R', 'N', 'B', 'Q', 'K', 'B', 'N', 'R'}
    };
}
```

```cpp
// Menampilkan papan catur
void displayBoard(const Board &board) {
    cout << "  ";
    for (int i = 0; i < BOARD_SIZE; i++) cout << i << " ";
    cout << endl;

    for (int i = 0; i < BOARD_SIZE; i++) {
        cout << i << " ";
        for (int j = 0; j < BOARD_SIZE; j++) {
            cout << board[i][j] << " ";
        }
        cout << endl;
    }
}
```

```
42  // Fungsi untuk memeriksa apakah gerakan valid
43  bool isValidMove(const Board &board, int x1, int y1, int x2, int y2, char player) {
44      // Pastikan koordinat dalam batas papan
45      if (x1 < 0 || x1 >= BOARD_SIZE || y1 < 0 || y1 >= BOARD_SIZE ||
46          x2 < 0 || x2 >= BOARD_SIZE || y2 < 0 || y2 >= BOARD_SIZE)
47          return false;
48
49      // Pemain 'P' hanya boleh bergerak maju satu langkah
50      if (player == 'P') {
51          if (x2 == x1 - 1 && y1 == y2 && board[x2][y2] == '.')
52              return true; // Maju satu langkah
53          if (x2 == x1 - 1 && abs(y2 - y1) == 1 && islower(board[x2][y2]))
54              return true; // Menyerang secara diagonal
55      }
56
57      // Pemain 'p' (komputer) hanya boleh maju satu langkah
58      if (player == 'p') {
59          if (x2 == x1 + 1 && y1 == y2 && board[x2][y2] == '.')
60              return true; // Maju satu langkah
61          if (x2 == x1 + 1 && abs(y2 - y1) == 1 && isupper(board[x2][y2]))
62              return true; // Menyerang secara diagonal
63      }
64
65      return false;
66  }
67
```

```
68  // Fungsi untuk memindahkan bidak
69  void makeMove(Board &board, int x1, int y1, int x2, int y2) {
70      board[x2][y2] = board[x1][y1];
71      board[x1][y1] = '.';
72  }
73
74  // Fungsi untuk algoritma lawan (gerakan acak)
75  void algoMove(Board &board) {
76      srand(time(0));
77      while (true) {
78          int x1 = rand() % BOARD_SIZE;
79          int y1 = rand() % BOARD_SIZE;
80          int x2 = x1 + 1;
81          int y2 = y1 + (rand() % 3 - 1); // Gerak lurus atau menyerang
82
83          if (board[x1][y1] == 'p' && isValidMove(board, x1, y1, x2, y2, 'p')) {
84              makeMove(board, x1, y1, x2, y2);
85              cout << "Algo moved from (" << x1 << ", " << y1 << ") to (" << x2 << ", " << y2 << ").\n";
86              break;
87          }
88      }
89  }
90
```

```cpp
// Fungsi utama
int main() {
    Board board;
    initializeBoard(board);

    cout << "Selamat datang di permainan catur sederhana!\n";
    displayBoard(board);

    while (true) {
        // Giliran pemain
        int x1, y1, x2, y2;
        cout << "Masukkan gerakan Anda (format: x1 y1 x2 y2): ";
        cin >> x1 >> y1 >> x2 >> y2;

        if (isValidMove(board, x1, y1, x2, y2, 'P')) {
            makeMove(board, x1, y1, x2, y2);
        } else {
            cout << "Gerakan tidak valid, coba lagi.\n";
            continue;
        }

        displayBoard(board);

```

```
        bool algoDefeated = true;
        for (int i = 0; i < BOARD_SIZE; i++) {
            for (int j = 0; j < BOARD_SIZE; j++) {
                if (board[i][j] == 'p') {
                    algoDefeated = false;
                    break;
                }
            }
        }
        if (algoDefeated) {
            cout << "Selamat, Anda menang!\n";
            break;
        }

        // Giliran algoritma
        algoMove(board);
        displayBoard(board);

        // Periksa kemenangan algoritma
        bool playerDefeated = true;
        for (int i = 0; i < BOARD_SIZE; i++) {
            for (int j = 0; j < BOARD_SIZE; j++) {
                if (board[i][j] == 'P') {
                    playerDefeated = false;
                    break;
                }
            }
        }
        if (playerDefeated) {
            cout << "Algo menang! Coba lagi.\n";
            break;
        }
    }

    return 0;
}
```

Ini adalah outputnya:

```
--interpreter=mi' ;495ca702-52e1-47f7-8ed3-daf12517b9dcSelamat datang di permainan catur sederhana!
  0 1 2 3 4 5 6 7
0 r n b q k b n r
1 p p p p p p p p
2 . . . . . . . .
3 . . . . . . . .
4 . . . . . . . .
5 . . . . . . . .
6 P P P P P P P P
7 R N B Q K B N R
Masukkan gerakan Anda (format: x1 y1 x2 y2): 6 0 5 0
  0 1 2 3 4 5 6 7
0 r n b q k b n r
1 p p p p p p p p
2 . . . . . . . .
3 . . . . . . . .
4 . . . . . . . .
5 P . . . . . . .
6 . P P P P P P P
7 R N B Q K B N R
Algo moved from (1, 3) to (2, 3).
  0 1 2 3 4 5 6 7
0 r n b q k b n r
1 p p p . p p p p
2 . . . p . . . .
3 . . . . . . . .
4 . . . . . . . .
5 P . . . . . . .
6 . P P P P P P P
7 R N B Q K B N R
Masukkan gerakan Anda (format: x1 y1 x2 y2):
```