Fortify Source Code Analyzer Software Engineer Candidate Homework

Project 1

You will define, implement and test a Sudoku solver.

Sudoku rules:

The objective is to fill a 9×9 grid with digits so that each column, each row, and each of the nine 3×3 subgrids that compose the grid (also called "boxes", "blocks", or "regions") contains all of the digits from 1 to 9.

Sudoku examples:

Easy

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

Difficult

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

Requirements

- Write an architecture document explaining the choices you have made to implement this solver.
- Implement the solver in Java.
  - Document the issues you have encountered and how you resolved them
  - Include the source code and other necessary files, setup instructions in your response.
- Check your solver with at least the 2 examples provided above

Project 2

The following code in C language contains some vulnerabilities. Analyze the code, report in the code as comments the vulnerabilities you have found and explain why these are vulnerabilities.

```
#include <stdio.h>
#include <stdlib.h>
#include <wchar.h>
#define PASSWORD "ABCD1234!"
/*You need not worry about other include statements if at all any are missing */

void func1()
{
    char * data;
    char * dataBuffer = (char *)ALLOCA(100*sizeof(char));
    memset(dataBuffer, 'A', 100-1);
    dataBuffer[100-1] = '\0';
    data = dataBuffer - 8;
    {
        char source[100];
        memset(source, 'C', 100-1);
        source[100-1] = '\0';
        strcpy(data, source);
        if(data != NULL)
        {
            printf("%s\n", data);
        }
    }
}

void func2()
{
    char * data;
    data = NULL;
    data = (char *)calloc(100, sizeof(char));
    strcpy(data, "A String");
    if(data != NULL)
    {
        printf("%s\n", data);
    }
}

void func3()
{
    char * password;
    char passwordBuffer[100] = "";
    password = passwordBuffer;
    strcpy(password, PASSWORD);
    {
        HANDLE pHandle;
        char * username = "User";
        char * domain = "Domain";
        /* Let's say LogonUserA is a custon authentication function*/
        if (LogonUserA(
                    username,
                    domain,
                    password,
                    &pHandle) != 0)
        {
            printf("User logged in successfully.\n");
            CloseHandle(pHandle);
        }
        else
        {
            printf("Unable to login.\n");
        }
    }
```

```
}

static void func4()
{
    char * data;
    data = NULL;
    data = (char *)calloc(20, sizeof(char));
    if (data != NULL)
    {
        strcpy(data, "Initialize");
        if(data != NULL)
        {
            printf("%s\n", data);
        }
        free(data);
    }
}

void func5()
{
    int i = 0;
    do
    {
        printf("%d\n", i);
        i = (i + 1) % 256;
    } while(i >= 0);
}

void func6()
{
    char dataBuffer[100] = "";
    char * data = dataBuffer;
    printf("Please enter a string: ");
    if (fgets(data, 100, stdin) < 0)
    {
        printf("fgets failed!\n");
        exit(1);
    }
    if(data != NULL)
    {
        printf("%s\n", data);
    }

}

void func7()
{
    char * data;
    data = "Fortify";
    data = NULL;
    printf("%s\n", data);
}

int main(int argc, char * argv[])
{
    printf("Calling func1\n");
    func1();

    printf("Calling func2\n");
    func2();

    printf("Calling func3\n");
    func3();

    printf("Calling func4\n");
    func4();

    printf("Calling func5\n");
    func5();

    printf("Calling func6\n");
    func6();
    printf("Calling func7\n");
    func7();

    return 0;
}
```