Arrays

1 Introduction

Through this training, to understand why arrays are necessary in programming. Grasp how to declare, create, initialize and operate arrays. Grasp how to use multidimensional arrays.

1.1 Evaluation

- Code Correctness: 60%
- Experimental Report: 40%

1.2 Knowledge Points

- Arrays Declaring and Creating
- Arrays Processing
- Arrays Initializing
- Arrays Passing
- Multidimensional Arrays

2 Demonstration

2.1 Declaring and Creating

(1) Declare an array variable using `dataType[] arrayName`. For example:

```
int[] intArray;
```

(2) Create an array using `new dataType[arraySize]`. For example:

```
intArray = new int[10];
```

2.2 Processing Arrays

(1) Initialize an array using `for-loop` statement. For example:

```
Scanner input = new Scanner(System.in);
for (int i = 0; i < intArray.length; i++){
    intArray[i] = input.nextInt();
}
```

(2) Initialize an array using initializer `dataType[] arrayName = {v1,v2,...,vk}`. For example:

```
int[] intArray = {1,2,3,4,5,6};
```

(3) Display an array:

```
for (int i = 0; i < intArray.length; i++){
    System.out.printf("%-3d", intArray[i]);
}
```

(4) Find the largest element:

```
double max = myList[0];
for (int i = 1; i < myList.length; i++) {
    if (myList[i] > max) max = myList[i];
}
```

(5) Accessing elements of Arrays using `foreach`:

```
for (for e: intArray){
    System.out.printf("%-3d", e);
}
```

2.3 Copying Arrays

(1) Copying arrays using `loop`:

```
int[] sourceArray = {2, 3, 1, 5, 10};
int[] targetArray = new int[sourceArray.length];
for (int i = 0; i < sourceArray.length; i++) {
    targetArray[i] = sourceArray[i];
}
```

(2) Copying arrays using `System.arraycopy(srcArr, srcPos, tarArr, tarPos, length)`:

```
int[] sourceArray = {2, 3, 1, 5, 10};
int[] targetArray = new int[sourceArray.length];
System.arraycopy(sourceArray, 0, targetArray, 0, sourceArray.length);
```

2.4 Passing Arrays to Methods

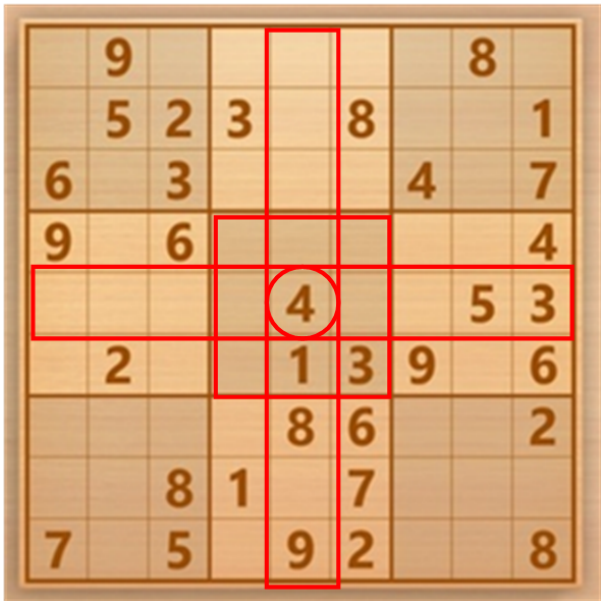
For an argument of an array type, the value of the argument is a `reference` to an array; this `reference value` is passed to the method. Semantically, it can be best described as `pass-by-sharing`, that is, the array in the method is the same as the array being passed. Thus, if you change the array in the method, you will see the change outside the method.

```
public class Test {
    public static void main(String[] args) {
        int x = 1; // x represents an int value
        int[] y = new int[10]; // y represents an array of int values
        m(x, y); // Invoke m with arguments x and y
        System.out.println("x is " + x);
        System.out.println("y[0] is " + y[0]);
    }
    public static void m(int number, int[] numbers) {
        number = 1001; // Assign a new value to number
        numbers[0] = 5555; // Assign a new value to numbers[0]
    }
}
```

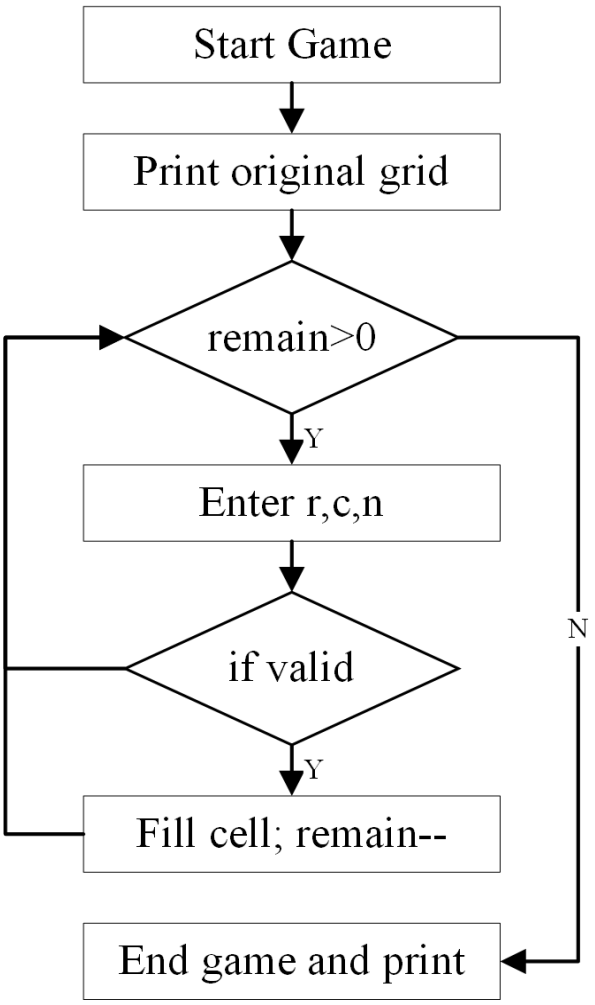
3 Experiment Content

3.1 Sudoku

The objective is to fill a 9×9 grid with digits so that each column, each row, and each of the nine 3×3 sub-grids that compose the grid contains all of the digits from 1 to 9.



(1) Design the main business processes.

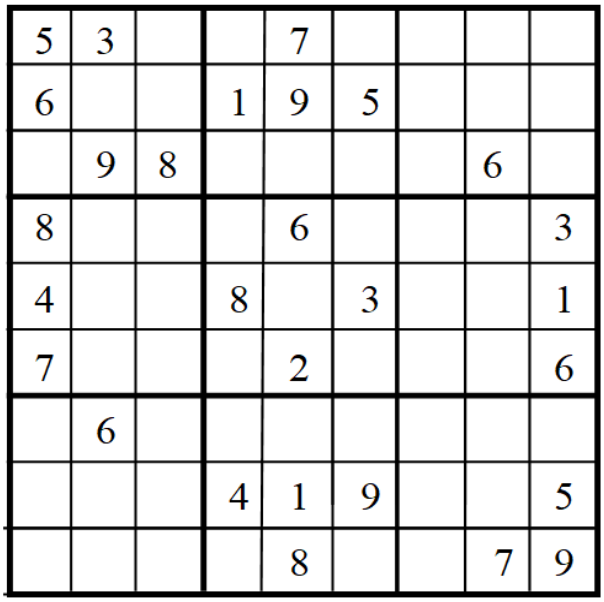


Main process code framework:

```
public static void main(String[] args){
    Scanner input = new Scanner(System.in);

    // Use a two-dimensional array to represent the game board

    // Print the initial state of the game

    // Count the number of cells need to fill

    // Play the game
    while(remain > 0){
        System.out.print("Enter row, column and number([1-9] [1-9] [1-9]):");
        int r = input.nextInt() - 1;
        int c = input.nextInt() - 1;
        int n = input.nextInt();

        if(grid[r][c] != 0){
            System.out.println("The cell is not empty!");
        }else{
            if(isValid(r,c,n,grid)){
                grid[r][c] = n;
                remain--;
```

```
                System.out.println("The current state of grid:");
                printGrid(grid);
            }else{
                System.out.println("The number is irrational.");
            }
        }
    }

    // End the game and print the final solution

    // Close the input stream
    input.close();
}
```

(2) Declare a two-dimensional array to represent a grid and initialize the grid using array initializer.



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

```
int[][] grid = {{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},
                {0,6,0,0,0,0,0,0,0},
                {0,0,0,4,1,9,0,0,5},
                {0,0,0,0,8,0,0,7,9}};
```

(3) Define method `printGrid(int[][] grid)` for printing grid during the game.

```
public static void printGrid(int[][] grid){
    for(int i = 0; i < 9; i++){
        for(int j = 0; j < 9; j++){
            // Print the numbers of the current line
        }
```

```
            System.out.println();
            // Print ----|----|----| at the right time
        }
    }
```

The format of grid printing is as follows:

```
 5 3 4 | 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 | 0 0 0 |
 0 0 0 | 4 1 9 | 0 0 5 |
 0 0 0 | 0 8 0 | 0 7 9 |
-------|-------|-------|
```

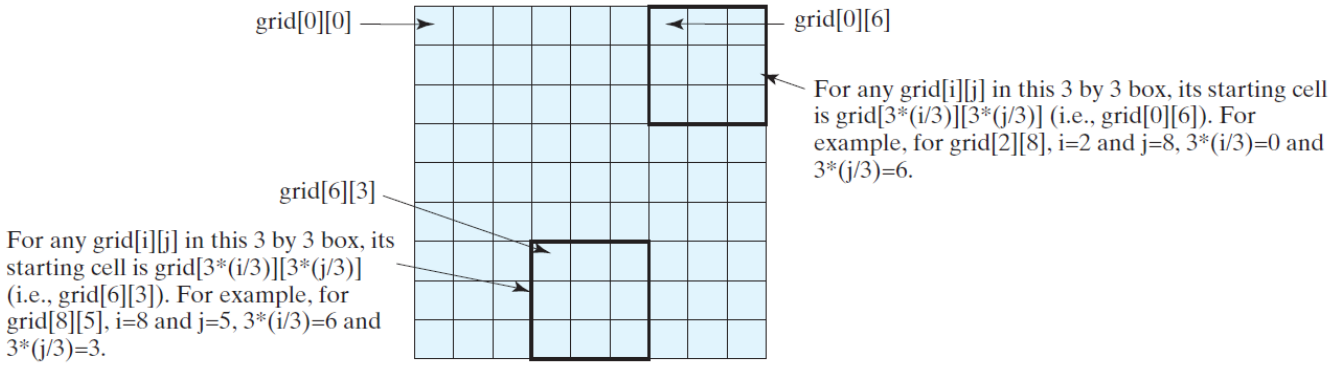
(4) Count the number of cells need to fill.

```
public static int countRemainNum(int[][] grid) {
    int remain = 0;
    // Traverse the grid to count the null cells.
    return remain;
}
```

(5) Check whether filling n in grid[r][c] is valid in the grid.

```
public static boolean isValid(int r, int c, int n, int[][] grid) {
    // Check whether n is valid at the r-1 row
    for (int column = 0; column < 9; column++)
      if (grid[r][column] == n)
        return false;

    // Check whether n is valid at the c-1 column

    // Check whether n is valid in the 3 by 3 box

    return true; // The value n at grid[i][j] is valid
}
```

The location of the first cell in a 3 * 3 box determines the locations of other cells in the box.



4 Experiment Report Requirements

4.1 Think and answer the question

(1) What is the difference between array variables and primitive data type variables?

(2) How to determine the location of the 3 * 3 box in above experiment.

(3) How to understand that `a two-dimensional array is an array of one-dimensional arrays`.

(4) Other experience.

4.2 Experiment report content

(1) Answer the above questions.

(2) All codes.

4.3 Submission method

(1) Upload the report by ftp:(Address:ftp://172.18.5.102; UserName:wangxiaomeng; Password: wangxiaomeng)

(2) File name format: StudentID+Name. For example, `20191234小明.docx`

4.4 Other Instructions

You can obtain experiment course resources through the web platform (URL: https://www.lanqiao.cn; InvitationCode: `ZF0XA4Y1`)