

자바프로그래밍및실습 과제 2

214823 박종현

April 2022

1 실습 과제

1.1 실습 과제 1

```
import java.util.Scanner;

public class Main {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.print(" 정수를 입력하시오 >> ");
        int n = sc.nextInt(); // 입력
        for (int i = n; i > 0; i--) { // n회 반복
            System.out.println("*".repeat(i)); // repeat 메서드는 인자만큼 문자열 객체를
            반복함
            // for (int j = 0; j < i; j++) System.out.print("*");
            // System.out.println();
        }
    }
}
```

1.2 실습 과제 2

```
import java.util.Scanner;

public class Main {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.print(" 정수를 입력하시오 >> ");
        int n = sc.nextInt();
        // n만큼 반복
        for (int i = 0; i < n; i++) {
            System.out.println("*".repeat(2*i+1)); // 별이 2*i+1회 찍혀야함 (i = 반복 횟수 ,
            0 부터 시작 )
            // for (int j = 0; j < 2*i+1; j++) System.out.print("*");
            // System.out.println();
        }
    }
}
```

1.3 실습 과제 3

```
import java.util.Scanner;

public class Main {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.print(" 소문자 알파벳 하나를 입력하시오 >> ");
        // charAt 메서드는 인자로 받아오는 인덱스 번호를 통해 문자열을
        인덱싱할 수 있도록 함
        // (= String 형 값을 char 값으로 캐스팅 )
        char gets = sc.next().charAt(0);
        // 'a'는 97임, gets에서 시작하여 'a' 까지 반복
        for (int i = gets; i >= 97; i--) {
            // 각 열마다 알파벳 출력
            for (int j = 97; j <= i; j++) {
                System.out.print((char)j);
            }
            System.out.println(); // 개행
        }
    }
}
```

1.4 실습 과제 4

```
import java.util.Scanner;
import java.lang.Math;

public class Main {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        // 문제 제시 값
        int n = (int)(Math.random()*50.0);
        int tries = 0; // 시도 횟수 기록
        while (true) { // 무한 루프
            tries++;
            System.out.print(" 숫자를 추측하여 보세요 : ");
            int gets = sc.nextInt(); // 입력
            if (gets < n) System.out.println("UP");
            else if (gets > n) System.out.println("DOWN");
            else {
                // 정답일 경우 시도 횟수를 출력하고 반복문 종료
                System.out.println(String.format(" 정답입니다 . 시도횟수 = %d", tries));
                break;
            }
        }
    }
}
```

2 과제

2.1 과제 1

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        int gets = -1; // do-while문을 사용하지 않는 대신 음수로 초기화
        Scanner sc = new Scanner(System.in);
        System.out.print("양의 정수를 입력하십시오 : ");
        while (gets <= 0) { // 양의 정수가 입력될 때까지 반복
            gets = sc.nextInt();
            if (gets > 0) break; // 입력값이 양수이면 아래 라인을 수행하지 않고
                                // 반복 종료
            System.out.print("양의 정수가 아닙니다 . 다시 입력하세요 : ");
        }
        System.out.println(); // 개행
        // 출력 내용을 덜 복잡하게 확인할 수 있도록 문자열 포매팅 수행
        System.out.println(String.format("%d의 약수는 다음과 같습니다 .", gets));

        // 가장 러프한 형태의 약수 알고리즘 구현
        // 시간복잡도 : O(n)
        for (int i = 1; i <= gets; i++) {
            if (gets % i == 0) System.out.print(String.format("%d ", i));
        }

        // 터미널 환경에서 프로세스를 실행하면
        // 종료 시 개행되어야 다음 터미널 명령을 깔끔하게 입력할 수 있음 .
        System.out.println();
    }
}
```

2.2 과제 2

```

import java.util.Scanner;

public class Main {
    public static boolean primes[];
    public static void main(String[] args) {
        int gets = -1; // do-while문을 사용하지 않는 대신 음수로 초기화
        Scanner sc = new Scanner(System.in);
        System.out.print("양의 정수를 입력하시오 : ");
        while (gets <= 0) { // 양의 정수가 입력될 때까지 반복
            gets = sc.nextInt();
            if (gets > 0) break; // 입력값이 양수이면 아래 라인을 수행하지 않고
                                // 반복 종료
            System.out.print("양의 정수가 아닙니다 . 다시 입력하세요 : ");
        }
        System.out.println(); // 개행

        // 에라토스테네스의 체
        // 시간복잡도 : O(n log log n)
        primes = new boolean[gets+1]; // gets까지 인덱싱할 수 있도록 gets+1 크기의
        // 부울 배열 생성
        for (int i = 0; i < gets+1; i++) primes[i] = true; // 부울 배열 기본 초기값은
        // false이므로 true로 초기화
        primes[0] = false; primes[1] = false; // 0과 1은 소수가 아님
        System.out.print("결과: ");
        for (int i = 2; i < gets+1; i++) {
            if (primes[i]) { // 만약 primes[i]가 true라면 i가 소수임을 의미함
                System.out.print(String.format("%d ", i));
                for (int j = i; j < gets+1; j+=i) primes[j] = false; // i의 배수를 전부
                // false(합성수)로 처리
            }
        }

        // 터미널 환경에서 프로세스를 실행하면
        // 종료 시 개행되어야 다음 터미널 명령을 깔끔하게 입력할 수 있음 .
        System.out.println();
    }
}

```

2.3 과제 3

```

import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int gets = -(int)2e9, max = gets; // gets를 입력, max를 최댓값으로 지정 , -INF로
        // 초기화
        String[] arr = {"첫", "두", "세"}; // 문장 일부분을 배열로 초기화
        for (String var: arr) { // foreach 구문 사용, arr.length 만큼 반복
            System.out.print(var+"번째 정수를 입력하세요 : "); // 문장 병합
            gets = sc.nextInt(); // 입력
            max = max > gets ? max : gets; // max값 갱신
        }
        System.out.println();
        System.out.println(String.format("Max값은 %d입니다.", max));
    }
}

```

```

import java.util.Scanner;

public class Main {
    public static void main(String[] args) {

```

```

Scanner sc = new Scanner(System.in);
int gets = (int)2e9, min = gets; // gets를 입력, min을 최솟값으로 지정, INF로 초기화
String[] arr = {"첫", "두", "세"}; // 문장 일부분을 배열로 초기화
for (String var: arr) { // foreach 구문 사용, arr.length 만큼 반복
    System.out.print(var+"번째 정수를 입력하세요 : "); // 문장 병합
    gets = sc.nextInt(); // 입력
    min = min < gets ? min : gets; // min값 갱신
}
System.out.println();
System.out.println(String.format("Min값은 %d입니다.", min));
}
}

```

2.4 과제 4

```

public class Main {
    public static void main(String[] args) {
        // 문제 제시 값
        int n[][] = {{1},{1,2,3},{1},{1,2,3,4},{1,2}};

        // 배열을 행렬로 생각했을 때
        // 첫번째 for문으로 열 조회
        for (int i = 0; i < n.length; i++) {
            // 두번째 for문으로 행 조회
            for (int j = 0; j < n[i].length; j++) {
                // (i, j)번 요소 조회
                System.out.print(String.format("%d ", n[i][j]));
            }
            // 하나의 열을 모두 조회하면 개행
            System.out.println();
        }
    }
}

```

2.5 과제 5

```

import java.util.Scanner;

public class Main {
    public static void main (String[] args) {
        Scanner sc = new Scanner(System.in);

        // 문제 제시 값
        int[] unit = {50000, 10000, 1000, 500, 100, 50, 10};

        // 잔액 변수 생성
        int remains = -1;
        System.out.print("금액을 입력하시오 >> ");
        remains = sc.nextInt();
        for (int u: unit) { // foreach문 사용, u는 unit의 요소 (단위)
            int n = remains / u; // 잔액을 단위로 나눈 몫 구하기 (= 지폐 혹은 화폐의 개수)
            if (n == 0) continue; // 몫이 0이면 다음 단위 처리
            System.out.println(String.format("%d원: %d개", u, n));
            remains -= u*n;
        }
    }
}

```

2.6 과제 6

```
import java.lang.Math;

public class Main {
    public static void main(String[] args) {
        // 배열 초기화
        int[][] arr = new int[4][4];
        // 이중 포문으로 배열의 각 값 접근
        for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 4; j++) {
                arr[i][j] = (int)(Math.random()*10+1); // 문제 제시 값
            }
        }
        for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 4; j++) {
                // 행 출력
                System.out.print(String.format("%d ", arr[i][j]));
            }
            // 하나의 열을 모두 출력하면 개행
            System.out.println();
        }
    }
}
```

2.7 과제 7

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        // 값 초기화
        int gets = -1, arr[];
        /*
        puts는 값이 이미 생성되었는지 확인하는 용도로 사용됨
        puts[val] = true라면 val은 이미 출력 대상인 것
        boolean[]은 디폴트 초기화값이 false임
        참조 : https://www.tutorialspoint.com/how-can-we-initialize-a-boolean-array-in-java
        */
        boolean puts[] = new boolean[101];
        puts[0] = true; // 출력 범위가 1 부터 100 사이 정수이므로 0은 사용되는
                       // 값이 아님 : 다른 용도로 사용
        System.out.print("정수 몇 개? >> ");
        gets = sc.nextInt();
        // 입력값만큼의 크기를 가진 배열 생성
        arr = new int[gets];

        // 초기화
        for (int i = 0; i < gets; i++) {
            int val = 0; // 윗줄의 puts[0] = true;를 활용하기 위해 val을 0 으로 초기화
            /*
            puts[0] = true이므로 while문 진입 가능 .
            이후 랜덤값으로 갱신되었을 때
            * puts[val]이 false라면 사용되지 않은 값임을 의미하므로
              while문이 종료되고 그대로 값을 사용함
            * puts[val]이 true라면 이미 사용된 값이므로
              while문은 종료되지 않고 난수 생성을 재시도함 .
            */
            while (puts[val]) { val = (int)(Math.random()*100+1); }
            puts[val] = true;
            arr[i] = val;
        }
    }
}
```

```

// 각 열에서 몇개 값을 출력했는지 기록함
int lnCounts = 0;
for (int i = 0; i < gets; i++) {
    System.out.print(String.format("%d ", arr[i]));
    lnCounts++;
    // 만약 값을 10 번 출력했다면 개행
    if (lnCounts % 10 == 0) System.out.println();
}

// 터미널 환경에서 프로세스를 실행하면
// 종료 시 개행되어야 다음 터미널 명령을 깔끔하게 입력할 수 있음 .
System.out.println();
}
}

```
