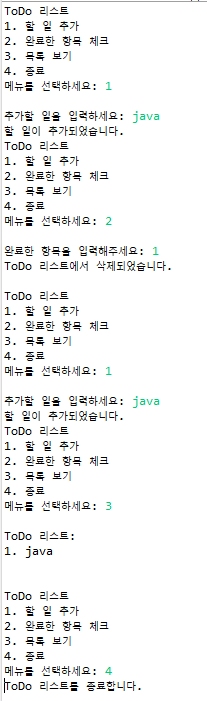
**JAVA 프로그래밍 LAB 2주차**

20234014 컴퓨터공학과 이은정

1. ToDo 리스트 작성하기

package ToDo\_list;

import java.util.ArrayList;

import java.util.Scanner;

public class ToDo\_list {

public static void main(String[] args) {

// TODO Auto-generated method stub

int input;

Scanner sc = new Scanner(System.in);

ArrayList<String> list = new ArrayList<String>();

int num;

do {

System.out.println("ToDo 리스트");

System.out.println("1. 할 일 추가");

System.out.println("2. 완료한 항목 체크");

System.out.println("3. 목록 보기");

System.out.println("4. 종료");

System.out.print("메뉴를 선택하세요: ");

input = sc.nextInt();

if (input == 1) {

System.out.print("\n추가할 일을 입력하세요: ");

String todo;

todo = sc.next();

list.add(todo);

System.out.println("할 일이 추가되었습니다.");

}

else if (input == 2) {

System.out.print("\n완료한 항목을 입력해주세요: ");

num = sc.nextInt();

list.remove(num - 1);

System.out.println("ToDo 리스트에서 삭제되었습니다.\n");

}

else if (input == 3) {

System.out.println("\nToDo 리스트:");

for (int i = 0; i < list.size(); i++) {

System.out.println((i + 1) + ". " + list.get(i));

}

System.out.println("\n");

} else if (input == 4)

break;

} while (input != 4);

System.out.println("ToDo 리스트를 종료합니다.");

sc.close();

}

}

}한 번 이상의 입력을 받아 반복해야 하기 때문에 do-while문을 사용했다. 리스트는 무한정으로 추가될 수 있기 때문에 ArrayList를 사용한다. Input에 사용자가 입력한 숫자를 저장한다.

1을 누를 경우 임시로 todo에 입력값을 받고, 그를 list에 추가한다. 제대로 실행되었음을 확인해주기 위해 추가되었다는 메시지를 추가로 출력한다.

만약 2를 입력했을 경우 완료한 항목의 번호를 입력해달라는 메시지를 출력한다. 다만 주의해야 할 것은 리스트는 0부터 인덱스가 시작한다. 내가 2번째로 입력한 값은 리스트 상 1의 인덱스를 가지고 있다. 따라서 list에서 그 항목을 삭제하려면 입력값-1의 인덱스를 빼야 한다. 따라서 위 유의사항에 유념하여 코드를 작성한다. 마찬가지로 코드가 제대로 실행되었는지 확인하기 위해서 제대로 삭제되었다는 메시지를 출력한다.

3을 입력할 시 list의 모든 항목을 출력한다. List 크기만큼 반복해야 하기 때문에 일반적인 for문을 사용한다. 다만 사용자의 편의성을 위해 순번은 i+1을 사용했다. List의 값을 가져오기 위해 list.get(i)를 사용해 그 항목을 가져온다.

4를 입력하면 종료해야 한다. 따라서 while문을 벗어나기 위해 break문을 사용해 해당 반복문을 탈출한다. Do-while문은 4를 입력 시 반복문을 멈추게 되어있지만, 혹시 모를 다른 경우를 차단하기 위해 input이 4가 아닐 경우에만 위 반복문을 반복한다.

2. 로또 번호 자동 생성기

**package** lottery;

**import** java.util.Arrays;

**public** **class** lottery {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**boolean**[] numbers = **new** **boolean**[46];

**int**[] lot = **new** **int**[6];

Arrays.*fill*(numbers, **true**);

**for** (**int** i = 0; i < 6; i++) {

**int** num = (**int**) (Math.*random*() \* 46 + 1);

**if** (numbers[num] == **false**)

**continue**;

numbers[num] = **false**;

lot[i] = num;

}

System.***out***.print("선택된 번호: ");

**for** (**int** i = 0; i < 6; i++) {

System.***out***.print(lot[i] + " ");

}



}

}

로또 번호는 1~45까지이다. 나는 boolean에서 46까지의 항목을 true로 모두 채우고, 랜덤으로 6개의 항목을 false로 바꾼 후에 true인 항목들을 로또 번호로 사용하려고 한다. 다만 같은 수가 중복 될 경우, 새로운 수를 뽑도록 다시 수를 뽑는다.

boolean에서 46개의 항목을 저장할 수 있는 배열을 만든다. Boolean[] 배열명 = new Boolean[배열 항목 개수]형으로 배열을 제작 시 자동으로 모든 항은 false로 채워진다. 또한 로또 번호를 저장할 lot이라는 이름의 int형 배열 또한 제작해준다.

Boolean 배열을 fill 매서드를 사용하여 전부 true로 채워준다. 이후에 for문을 사용해 랜덤으로 6개의 수를 뽑는다. Math.random()을 실행할 시 0~1 미만까지의 랜덤한 double형 수를 만든다. 즉슨 내가 1 이상 max 이하의 랜덤한 수를 뽑고 싶다면 random()을 통해 수를 랜덤하게 뽑은 다음 해당 수에 max 범위의 수를 곱해주고 +1을 해주어야 한다는 뜻이다. 나는 Boolean 리스트에서 1~46 인덱스를 사용할 것이므로 45를 곱해주고 거기에 1을 더해준다. 뽑은 수는 false로 다시 그 값을 바꿔주는데, 만약 내가 이전에 그 수를 뽑은 적이 있다면 이미 그 값은 false 처리가 되어 있을 것이다. Numbers에서 그 인덱스의 값이 false인지를 검사하고, false일 경우 continue하여 다시 수를 뽑을 수 있도록 처리한다. 또한 로또는 뽑은 순서가 매우 중요하기 때문에, lot에 그 순서대로 수를 저장해준다. 마지막으로 print를 사용해 lot의 모든 항목을 출력해주면 로또 자동 숫자 뽑기 코드가 완성된다.

3. 계산기

**package** Cal;

**import** java.util.Scanner;

**public** **class** Cal {

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("연산을 입력하세요: ");

String operator = sc.next();

System.***out***.print("피 연산자 2개를 입력하세요: ");

**int** a = sc.nextInt();

**int** b = sc.nextInt();

**if** (operator.equals("\*"))

System.***out***.println(a + "\*" + b + "=" + (a \* b));

**else** **if** (operator.equals("+"))

System.***out***.println(a + "+" + b + "=" + (a + b));

**else** **if** (operator.equals("-"))

System.***out***.println(a + "-" + b + "=" + (a - b));

**else** **if** (operator.equals("/")) {

**if** (b == '0')

System.***out***.println("분모가 0이므로 계산할 수 없습니다.");

**else**

System.***out***.println(a + "/" + b + "=" + (a / (b - '0')));

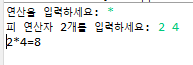
sc.close();

}

}

}



수 2개를 입력 받고 이를 계산한다. 연산자는 문자다. 따라서 string에 해당 문자를 입력 받아 저장한다. 그 다음으로 int a, b에 각각의 수를 저장한다. c언어와 비슷하게 nextInt()식으로 값을 입력받을 경우 띄어쓰기를 기준으로 수를 각각 앞 뒤로 인식하여 수를 저장할 수 있다. 그 이후는 연산자에 따라 결과값을 출력한다.

이 때 주의해야 할 점은 연산자가 문자라는 것이다. 문자라는 말은 ==으로 비교를 할 수 없다는 말이다. c언어에서는 문자는 strcmp를 통해 비교했었다. 여기서는 문자열.equals를 통해 그 값을 비교한다. 비교해보고, 각각의 값에 맞도록 그 값을 출력한다. 분자는 분모 0값을 가질 수 없으므로 이는 if문을 통해 처리한다.

4. 자동 기사 작성하기

**package** news;

**import** java.util.Scanner;

**public** **class** news {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("경기장은 어디입니까?: ");

String area = sc.next();

System.***out***.print("이긴 팀은 어디입니까?: ");

String win = sc.next();

System.***out***.print("진 팀은 어디입니까?: ");

String lose = sc.next();

System.***out***.print("우수 선수는 누구입니까?: ");

String MVP = sc.next();

System.***out***.print("스코어는 몇 대 몇입니까?: ");

String score = sc.next();

System.***out***.println("=======================");

System.***out***.println("오늘 " + area + "에서 야구 경기가 열렸습니다.");

System.***out***.println(win + "과 " + lose + "은 치열한 공방전을 펼쳤습니다.");

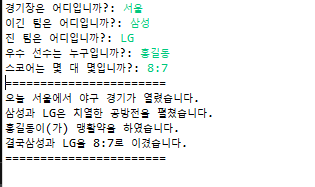
System.***out***.println(MVP + "이(가) 맹활약을 하였습니다.");

System.***out***.println("결국" + win + "이 " + lose + "을 " + score + "로 이겼습니다.");

System.***out***.println("=======================");

}

}

뉴스 기사를 자동으로 입력하는데 가장 큰 변수는 경기장, 이긴 팀, 진 팀, 우수 선수, 스코어다. 이는 그 때 그 때마다 입력을 받고, 이를 기사에 바로 적용시켜 출력한다. 다만 이 때 기사에는 변수를 사용하여 불러온다. String 변수명에 각각의 정보를 모두 입력 받는다. 그리고 기사를 출력한다.

기사는 크게 변수, 일반 기사로 나뉘어 있다. 변수와 일반 기사 혹은 변수와 변수를 연결해 출력하기 위해서는 각 요소를 +로 연결해주어야 한다. 따라서 각 기사와 변수를 적절한 위치에 놓고 서로를 +로 연결해주면 각각의 변수에 저장된 값을 한 줄로 출력할 수 있다.

5. 그림 그리기 프로그램

package draw;

import java.util.Arrays;

import java.util.Scanner;

public class draw {

public static void main(String[] args) {

// TODO Auto-generated method stub

int[][] can = new int[5][5];

Scanner sc = new Scanner(System.in);

do {

System.out.println("그림 그리기 프로그램");

System.out.println("1. 직선 그리기");

System.out.println("2. 정사각형 그리기");

System.out.print("원하는 그림을 선택하세요(1-2, 종료: 0): ");

int input = sc.nextInt();

if (input == 0)

break;

else if (input == 1) {

System.out.print("첫 번째 점의 좌표: ");

int a = sc.nextInt();

int b = sc.nextInt();

System.out.print("두 번째 점의 좌표: ");

int c = sc.nextInt();

int d = sc.nextInt();

if (a == c) { // 수직 선

for (int j = Math.min(b, d); j <= Math.max(b, d); j++) {

can[a][j] = 1;

}

} else if (b == d) { // 수평 선

for (int i = Math.min(a, c); i <= Math.max(a, c); i++) {

can[i][b] = 1;

}

} else if (Math.abs(a - c) == Math.abs(b - d)) { // 45도 기울기 선

int dx = (a < c) ? 1 : -1;

int dy = (b < d) ? 1 : -1;

int x = Math.min(a, c);

int y = Math.min(b, d);

while (x != c) {

can[x][y] = 1;

x += dx;

y += dy;

}

can[c][d] = 1;

}

} else if (input == 2) {

System.out.print("첫 번째 점의 좌표: ");

int a = sc.nextInt();

int b = sc.nextInt();

System.out.print("두 번째 점의 좌표: ");

int c = sc.nextInt();

int d = sc.nextInt();

int xmin = Math.min(a, c);

int xmax = Math.max(a, c);

int ymin = Math.min(b, d);

int ymax = Math.max(b, d);

for (int i = xmin; i <= xmax; i++) {

for (int j = ymin; j <= ymax; j++) {

can[i][j] = 1;

}

}

}

for (int i = 0; i < 5; i++) {

for (int j = 0; j < 5; j++) {

if (can[i][j] == 0)

System.out.print("-");

else

System.out.print("\*");

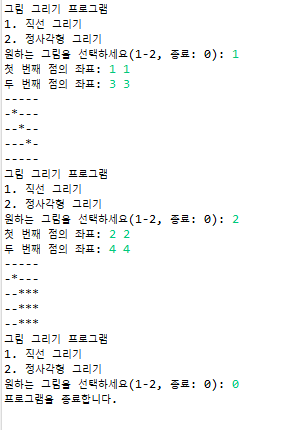
}

System.out.print("\n");

}

} while (true);

System.out.print("프로그램을 종료합니다.");

 sc.close();

}

}

그림을 그리는 경우는 2가지다. 선을 긋거나 사각형을 그리는 것이다. Can[5][5]를 모두 0으로 채우고, 내가 그림을 그려야 하는 부분만을 1로 채운다. 그리고 최종 단계에서 1을 \*로 나머지는 -로 출력해 그림을 그린다. 그림을 그리기 위해 nextInt 함수를 통해 내가 어떤 선택을 할지 값을 입력 받는다. 그림을 계속해서 그릴 수 있도록 do-while(ture) 구조를 사용해 무한반복 하도록 구조를 작성한다.

1번을 눌러 선을 그리는 것을 선택한다면 첫 번째 좌표와 두 번째 좌표를 입력 받는다. nextInt()를 2개 연속으로 사용하여 띄어쓰기를 기준으로 값을 2개씩 입력 받는다. 선을 그리는 방법은 크게 3가지이다. 수평, 수직, 45도 각도를 그리는 선이다. 두 좌표의 x값, y값을 비교해 만약 같다면 이중 배열에서 다른 부분만을 for문을 이용해 1을 대입한다. 이 때 Math.max 혹은 Math.min을 사용해 만약 4,4 2,2처럼 비교적 수가 작은 좌표를 나중에 입력하더라도 제대로 그림을 그릴 수 있도록 한다. 가장 복잡한 기능은 대각선으로 그림을 그리는 것이다. 대각선을 그리는 조건은 각각의 x, y좌표의 차이가 같다는 것이다. 즉슨 정사각형인지를 먼저 확인해야 하므로 Math.ads를 사용해 x, y의 절댓값이 같은지를 먼저 검사한다. 대각선으로 그림을 그릴 때에도 앞서 말한 것처럼 어느 좌표를 입력하더라도 대각선을 그릴 수 있어야 하므로 각각의 x, y좌표중 누가 더 큰지를 알아야 한다. 만약 두 번째로 입력한 좌표가 더 크다면 1을, 작다면 -1을 대입한다. c언어에서도 사용했던 ?문을 java에서도 사용할 수 있기 때문에 사용하면 쉽게 각각의 변수에 대입할 값을 구할 수 있다. 이후에는 각각의 수를 하나씩 키워가며 can[x][y]에 1을 대입한다.

정사각형 그리기 프로그램은 더 간단하다. 마찬가지로 두 좌표에서 x, y를 입력 받고, 더 큰 좌표-작은 좌표의 x, y값을 Math.max, Math.min을 통해 계산한다. 이후에는 위에 대각선을 그렸던 것 처럼 for문을 사용해 그림을 그리면 된다.