[디지털 컨버전스]
스마트 콘텐츠와 웹 융합 응용 SW
개발자 양성과정

강사 : 이상훈
학생 : 김원석
6회차 수업
2021/05/14 금요일

1. 스위치스트링 테스트(Switch문)
    - 이해 되었습니다.

```java
import java.util.Scanner;

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

        System.out.print("문자열도 switch 처리가 되나요 ? 일단 입력해봅시다: ");
        Scanner scan = new Scanner(System.in);    // 스캐너 클래스로 문자를 입력하기 위해 준비한다.

        String str = scan.nextLine();        // 문자열을 쓸수 있도록하는 문법 "String"

        // 주의점: 앞선 예에서도 적었지만 switch에 사용되는 데이터타입과
        // case에서 사용하는 데이터타입을 일치시킬 필요가 있다.
        switch (str) {
            case "hi":
                System.out.println("안녕");
                break;
            default:        //case에 없는 상황을 고려하여 default도 만들어 준다.
                System.out.println("으아앜");
                break;
        }
    }
}
        // 스위치 문을 통해서 케이스에 맞는 입력을 넣을경우 해당 케이스의 입력값이 출력이 되고,
        // 없을경우 default로 이동하여 출력됨
```

2. 스위치 테스트(Switch문)

```
public class SwitchTest {
    public static void main(String[] args) {
        System.out.println("저희 상점에 방문해주셔서 감사합니다. 물건을 고르십쇼 호갱님!");

        // Boolean 이란 참, 거짓을 표현할 수 있는 데이터타입이다.      //Boolean을 쓰면 두가지 참(true) or 거짓(false) 으로밖에 표현이 안됨.
        Boolean isTrue = true;                                  // isTrue라는 변수명과 함께 참을 만들어줌

        Scanner scan = new Scanner(System.in);  //Scanner 클래스를 이용하여 문자를 쓸 준비
        int num;                                // int 변수명 준비

        while (isTrue) {                        // While문법을 사용하여 입력값 작성
            System.out.print("숫자를 눌러 물건을 담으세요: ");

            num = scan.nextInt();

            String str = "hi";                      // 이부분은 왜 필요한지? (hi)가 문자열을 쓰기위한 용도 인가?

            // 입력된 키보드 값에 따라 적절한 처리를 하게 된다.
            // 키보드 값에 따라 처리하는 루틴은 case x에 해당한다.
            // 0번이 눌렸다면 case 0, 1번이라면 case 1과 같은 형식이다.

            // switch에서 판정에 사용하는 것이 String이라면
            // case에서 사용하는것도 String으로 맞춰서 동작시킬 수 있다.
```

```java
switch (num) {
    // 문자 낱개로는 가능함(홑따옴표)
    // 현재는 숫자값이라서 현재는 문장 여러개의 문자열(쌍따옴표는 불가능함)
    case 0:
        System.out.println("탈출합니다.");
        isTrue = false;
        break;
    case 1:
        System.out.println("비누를 장바구니에 담았습니다.");
        // break;
        // break는 더 이상 밑으로 내려가지 않고
        // 이 시점에서 종료할 수 있게 도와주는 역할을 한다.
        break;
    case 2:
        System.out.println("신발을 장바구니에 담았습니다.");
        break;
    case 3:
        System.out.println("에어팟을 장바구니에 담았습니다.");
        break;
    default:
        // 이 default라는 녀석은 말 그대로 기본값에 해당함
        // 우리가 예상치 못한 입력이 존재할 수 있음
        // 이 경우에 활용하는것이 default라고 보면 됩니다.
        System.out.println("그런건 없습니다!");
        break;
}
```

다 이해 되었습니다.

// break 사용하여 더이상 다음 입력값이 출력되지 않고, 종료 되게 해준다.

// default는 case상에 존재하지 않은 입력값이 있을 경우를 대비해 만들어 준다.

25번문제 복습

3. 25번 문제풀이

```
public class Prob25 {
    public static void main(String[] args) {
        /*
            1 ~ 100 까지의 숫자중 2의 배수는 모두 더한다.
            여기서 5의 배수는 모두 뺀다.
            11의 배수는 더한다.
            중복이 발생할 경우엔 무시
         */

        int sum = 0;

        for (int i = 1; i <= 100; i++) {
            if (i % 11 == 0 && i % 5 == 0 && i % 2 == 0) {
                System.out.println("110의 배수 = " + i);
            } else if (i % 11 == 0 && i % 5 == 0) {
                System.out.println("55의 배수 = " + i);
            } else if (i % 11 == 0 && i % 2 == 0) {
                System.out.println("22의 배수 = " + i);
            } else if (i % 5 == 0 && i % 2 == 0) {
                System.out.println("10의 배수 = " + i);
            } else if (i % 11 == 0) {
                System.out.println("11의 배수 = " + i);
                sum += i;
            } else if (i % 5 == 0) {
                System.out.println("5의 배수 = " + i);
                sum -= i;
            } else if (i % 2 == 0) {
                System.out.println("2의 배수 = " + i);
                // System.out.printf("2의 배수 = %d\n", i);
                sum += i;
            }
        }

        System.out.println("최종 결과 = " + sum);
    }
}
```

//for문을 사용하여 1 ~ 100까지의 숫자범위를 설정

// 이부분은 11의 배수, 2의 배수, 5의 배수가 해당하는
중복부분을 걸러내는 입력값인것 같다.

//해당부분은 11의 배수를 모두 더하고
5의 배수는 빼고, 2의 배수는 더하게 하는
입력값이다.

// 최종결과값은 중복부분은 모두 빠지고 2의 배수와 11의 배수는 모두 더해지고 5의 배수는 빠진 결과 값이다.

4. 배열 Array (arr [] = {};)

```
public class ArrayTest {
    public static void main(String[] args) {
        // 배열은 왜 써야 할까 ?
        // 동일한 데이터 타입의 변수가 여러개 필요할때
        // 일일히 int a, b, c, d, e, f, g, h, i ... z 까지 해봐야 26개 밖에 안됨
        // 만약 회사에서 직원 1000명을 관리해야 한다 가정한다면
        // 이것을 일일히 변수로 선언한다면 죽을 것이다.
        // 당연히 배열을 만들어서 관리해야할 것이다.
        int arr[] = { 1, 2, 3, 4, 5 };
        // int num1 = 1, num2 = 2, num3 = 3, num4 = 4, num5 = 5;
        // 데이터가 많으면 많을수록 단일 변수 선언은 지옥을 체험하게 해줄 것이다.
        // 그러니 우리는 심신의 안정을 위해 배열을 사용해야할 것이다.

        // 배열을 만드는 방법
        // 1. stack에 할당하는 방법(지역 변수)
        //    1-1. 일단은 배열의 데이터 타입(int 같은)을 적는다.
        //    1-2. 배열의 이름이 될 변수명을 적는다.
        //    1-3. 배열임을 알리기 위해 []을 변수 옆에 적어준다.
        //    1-4. 필요하다면 배열의 값들을 초기화한다.
        //         (이때 원소로 지정한 숫자에 따라 배열의 길이가 지정된다)
        //    * 가변으로 구성하고 싶다면 new를 사용해야 하는데 이것은 다음주에 학습하도록 한다.

        // 아래와 같은 데이터를 살펴보자
        // int arr[] = { 1, 2, 3, 4, 5 };
        // 위 데이터는 아래와 같은 형식으로 저장된다.
```

// <<=== 배열식

//<= 여기서 스택이란 {}<- 이부분을 말하는것 같음

Q = 강사님 정확히 지역 변수란 무엇인가요? 다시 설명 해주실 수 있으실까요?

```
        // 아래와 같은 데이터를 살펴보자
        // int arr[] = { 1, 2, 3, 4, 5 };
        // 위 데이터는 아래와 같은 형식으로 저장된다.

        //     ---------------------
        // arr | 1 | 2 | 3 | 4 | 5 |
        //     ---------------------
        //       [0] [1] [2] [3] [4]
        // 배열의 인덱스(방) 번호는 0번부터 시작함에 주의하도록 한다.
        // 그러나 방 번호가 순차적으로 증가하기 때문에
        // for 문이나 while 문등의 반복문과의 혼합구성에 있어 매우 탁월하다.

        for (int i = 0; i < 5; i++) {
            // System.out.printf("arr[%d] = %d\n", i, arr[i]);
            System.out.println("arr[" + i + "] = " + arr[i]);
        }
    }
}

// stack(지역변수)에 할당한다는 것은 지역변수로 처리함을 의미합니다.
// 그렇기 때문에 나중에 매서드나 클래스를 학습한 이후 스택에 할당하면
// 해당 매서드 혹은 클래스 내부에서만 해당 배열이 활성화됩니다.

// arr[0] = x, arr[1] = y, arr[2] = z, arr[3] = k
// System.out.println("arr[" + 0 + "] = " + x + ", arr[" + 1 + "] = " + y + ", arr[ ..... 지옥");
// System.out.printf("arr[%d] = %d, arr[%d] = %d, arr[%d] = %d, arr[%d] = %d\n",
//                   0, arr[0], 1, arr[1], 2, arr[2], 3, arr[3])
```

// 배열을 사용했을때 진행 되는 원리

// 배열은 순차적으로 증가하기 때문에
반복문과의 혼합 (if문, while문)등의
사용에 매우 적합!

5. 컨티뉴 continue test

```
public class ContinueTest {
    public static void main(String[] args) {
        for (int i = 0; i < 10; i++) {
            if (i % 2 == 0) {
                // continue 를 만나면 아래쪽에 진행해야하는 코드가 남아있더라도
                // 무조건 for loop의 최상단으로 이동하게 된다.
                // 그러므로 증감식이 진행된다.
                continue;
            }

            System.out.println("i = " + i);
        }
    }
}
```

6. 27번 챌린지 문제

```java
public class Pro27 {
    public static void main(String[] args) {
        //아래와 같은 형태의 숫자 배치가 있다.
        //1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...    피보나치수열
        //사용자가 15를 입력하면 15번째 값을, 8을 입력하면 8번째 값을 구하도록 프로그래밍 해보자!
        //(n을 입력하면 n 번째 값을 구하도록 프로그래밍 해보자 ~)
        Scanner scan = new Scanner(System.in);
        System.out.println("원하시는 값을 말씀해 주세요.");
        int num = scan.nextInt();

        int a = 1, b = 1, c = 0;

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

            a = b;
            b = c;
            c = a + b;

        }
        System.out.println("원하시는 값의 숫자는 = " + c);

        }
    }
```

// 옆에와 같이 순차적으로 값을 구할수 있도록 만들었는데, 이 방법말고도 다른 간단한 공식이 있는지 궁금합니다.