Chapter 04. 조건문과 반복문

4.1 코드 실행 흐름 제어

흐름 : main( ) 메소드의 시작 중괄호 { 에서 시작해서 끝 중괄호 } 까지 위에서 부터 아래로 실행하는 것. **흐름 제어** : 제어문(흐름 제어문)을 이용해 실행 흐름을 원하는 방향으로 해준다.

루핑 : 반복문이 흐름을 다시 되돌아가게 하는 것.

4.2 조건문(if문, switch문)

4.2.1 if문

: 조건식의 결과에 따라 블록 실행 여부가 결정. 조건식에서는 true 또는 false 값을 산출할 수 있는 연산식이나, boolean 변수가 올 수 있다.

- **if문 예제 코드**

```
public class IfExample {
    public static void main(String[] args) {
        int score = 93;

        if(score >= 90){
            System.out.println("점수가 90보다 큽니다.");
            System.out.println("등급은 A 입니다.");
        }

        if(score < 90)
            System.out.println("점수가 90보다 작습니다.");
            System.out.println("등급은 B 입니다.");
    }
}
```

실행 결과

```
점수가 90보다 큽니다.
등급은 A 입니다.
등급은 B 입니다.
```

> **System.out.println("등급은 B 입니다.");** 는 **if(score < 90)** 에 무관하여 조건에 만족하지 못하여도 출력이 된다.

4.2.2 if-else 문

- **예제 코드**

```
public class IfElseExample {
    public static void main(String[] args) {
        int score = 85;

        if(score >= 90){
            System.out.println("점수가 90보다 큽니다.");
            System.out.println("등급은 A 입니다.");
        }else{
            System.out.println("점수가 90보다 작습니다.");
            System.out.println("등급은 B 입니다.");
        }
    }
}
```

실행 결과

```
점수가 90보다 작습니다.
등급은 B 입니다.
```

4.2.3 if - else if - else 문

: 여러 개의 조건문

- **예제 코드**

```
public class IfElseIfElseExample {
    public static void main(String[] args) {
        int score = 75;

        if(score >= 90){
            System.out.println("점수가 100~90 입니다.");
            System.out.println("등급은 A 입니다.");
        }else if(score >= 80){
            System.out.println("점수가 80~89 입니다.");
            System.out.println("등급은 B 입니다.");
        }else if(score >= 70){
            System.out.println("점수가 70~79 입니다.");
            System.out.println("등급은 C 입니다.");
        }else{
            System.out.println("점수가 70 미만 입니다.");
            System.out.println("등급은 D 입니다.");
        }
    }
}
```

실행 결과

```
점수가 70~79 입니다.
등급은 C 입니다.
```

- **Math.rando( )**

```
0.0 <= Math.random() < 1.0
0   <= (int)(Math.random()*10) < 10
```

- **주사위의 번호를 뽑는 예제**

```
public class IfDiceExample {
    public static void main(String[] args) {
        // 1 부터 6 까지 랜덤 수 받기
        int num = (int)(Math.random()*6) + 1;

        if(num==1){
            System.out.println("1");
        }else if(num==2){
            System.out.println("2");
        }else if(num==3){
            System.out.println("3");
        }else if(num==4) {
            System.out.println("4");
        } else if(num==5) {
            System.out.println("5");
        } else {
            System.out.println("6");
        }
    }
}
```

실행 결과

```
5
```

4.2.4 중첩 if문

: if문의 블록 내부에 또 다른 if문을 사용하는 것

- **예제 코드**

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

```
        int score = (int)(Math.random()*20) + 81;
        System.out.println("점수 : " + score);

        String grade;

        if(score>=90){
            if(score>=95){
                grade = "A+";
            }else{
                grade = "A";
            }
        } else {
            if(score>=85){
                grade = "B+";
            }else{
                grade = "B";
            }
        }

        System.out.println("학점 : " + grade);
    }
}
```

실행 결과

```
점수 : 84
학점 : B
```

4.2.5 switch 문

: 변수의 값에 따라서 실행문이 결정. 경우의 수가 많을 때 if문 보다 간결하다.

- **예제 코드**

```
public class SwitchExample {
    public static void main(String[] args) {
        int num = (int)(Math.random()*6) + 1;

        switch (num) {
            case 1:     // num == 1 일때
                System.out.println("1");
                break;
            case 2:     // num == 2 일때
                System.out.println("2");
                break;
            case 3:     // num == 3 일때
                System.out.println("3");
                break;
            case 4:     // num == 4 일때
                System.out.println("4");
```

```
                break;
            case 5:     // num == 5 일때
                System.out.println("5");
                break;
            default:    // 나머지
                System.out.println("6");
                break;
        }
    }
}
```

실행 결과

```
5
```

- **break문이 없는 case 예제 코드**

```
public class SwitchNoBreakExample {
    public static void main(String[] args) {
        int time = (int)(Math.random()*4) + 8;
        System.out.println("현재기간 : " + time + " 시");

        switch (time){
            case 8:
                System.out.println("출근합니다.");
            case 9:
                System.out.println("회의를 합니다.");
            case 10:
                System.out.println("업무를 봅니다.");
            default:
                System.out.println("외근을 나갑니다.");
        }
    }
}
```

실행 결과

```
현재기간 : 9 시
회의를 합니다.
업무를 봅니다.
외근을 나갑니다.
```

> break가 없다면 다음 case가 연달아 실행된다.

- **char 타입의 Switch문 예제 코드**

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

```
        char grade = 'B';

        switch (grade) {
            case 'A':
            case 'a':
                System.out.println("우수 회원입니다.");
                break;
            case 'B':
            case 'b':
                System.out.println("일반 회원입니다.");
                break;
            default:
                System.out.println("손님입니다.");
        }
    }
}
```

실행 결과

```
일반 회원입니다.
```

- **String 타입의 Switch문 예제**

```
public class SwitchStringExample {
    public static void main(String[] args) {
        String position = "과장";

        switch (position) {
            case "부장":
                System.out.println("700만원");
                break;
            case "과장":
                System.out.println("500만원");
                break;
            default:
                System.out.println("300만원");
        }
    }
}
```

실행 결과

```
500만원
```

4.3 반복문(for문 , while문 , do-while문)

for문 : 반복 횟수를 알고 있을 때 주로 사용

while문 : 조건에 따라 반복할 때 주로 사용(조건을 먼저 검사)

do-while문 : 조건을 나중에 검사할 때 주로 사용

4.3.1 for문

- **예제 코드**

```
public class ForPrintFrom1To10Example {
    public static void main(String[] args) {
        for(int i=1; i<=10; i++){
            System.out.println(i);
        }
    }
}
```

실행 결과

```
1
2
3
4
5
6
7
8
9
10
```

- **1부터 100까지 합 출력 예제**

```
public class ForSumFrom1To100Example {
    public static void main(String[] args) {
        int sum = 0;

        for(int i=1; i<=100; i++){
            sum += i;
        }

        System.out.println("1~100 합 : " + sum);
    }
}
```

실행 결과

```
1~100 합 : 5050
```

> 위의 코드에

> **System.out.println("1~" + i + " 합 : " + sum);** 을 추가하면 컴파일 에러가 발생한다.
> 이유는 for문에 선언된 i는 for문을 벗어나면 사용할 수 없기 때문이다.
> 변수 i를 불러오기 위해서는 for문 전에 선언되어야 한다.

- **부동소수점 타입 반복문**
  : 초기화식에서 루프 카운트 변수를 선언할 때 부동소수점 타입은 사용하면 안된다.

예제 코드

```
public class ForFloatCounterExample {
    public static void main(String[] args) {
        for(float x = 0.1f; x<=1.0f; x+=0.1f){
            System.out.println(x);
        }
    }
}
```

실행 결과

```
0.1
0.2
0.3
0.4
0.5
0.6
0.70000005
0.8000001
0.9000001
```

> 0.1 은 float 타입으로 정확하게 표현할 수 없기 때문에 x에 더해지는 실제값은 0.1보다 약간 크다. 그러므로 루프는 9번만 실행된다.

- **구구단 예제 코드**

```
public class ForMultiplicationTableExample {
    public static void main(String[] args) {
        for(int m=2; m<=9; m++){
            System.out.println("***" + m + "단 ***");
            for(int n=1; n<=9; n++){
                System.out.println(m + " x " + n + " = " + (m*n));
            }
        }
    }
}
```

실행결과

```
***2단 ***
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18

...

***9단 ***
9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
```

4.3.2 while문

- **1부터 10까지 출력 예제**

```java
public class WhilePrintFrom1To10Example {
    public static void main(String[] args) {
        int i = 1;
        while (i<=10) {
            System.out.println(i);
            i++;
        }
    }
}
```

실행결과

```
1
2
3
4
5
6
7
8
9
10
```

- **1부터 100까지 합을 출력**

```
public class WhileSumFrom1To100Example {
    public static void main(String[] args) {
        int sum = 0;

        int i = 1;

        while(i<=100){
            sum += i;
            i++;
        }

        System.out.println("1~" + (i-1) + " 합 : " + sum);
    }
}
```

실행결과

```
1~100 합 : 5050
```

- **키보드로 while문 제어**

```
public class WhileKeyControlExample {
    public static void main(String[] args) throws Exception {
        boolean run = true;
        int speed = 0;
        int keyCode = 0;

        while(run){
            if(keyCode!=13 && keyCode!=10){
                System.out.println("----------------");
                System.out.println("1. 증속 | 2. 감속 | 3. 중지");
                System.out.println("----------------");
                System.out.println("선택 : ");
            }
```

```
            keyCode = System.in.read(); // 키보드 키 코드를 읽음

            if (keyCode == 49) {
                speed++;
                System.out.println("현재 속도= " + speed);
            } else if (keyCode == 50){
                speed--;
                System.out.println("현재 속도= " + speed);
            } else if (keyCode == 51){
                run = false;
            }
        }

        System.out.println("프로그램 종료");
    }
}
```

실행결과

```
---------------
1. 증속 | 2. 감속 | 3. 중지
---------------
선택 :
1
현재 속도= 1
---------------
1. 증속 | 2. 감속 | 3. 중지
---------------
선택 :
2
현재 속도= 0
---------------
1. 증속 | 2. 감속 | 3. 중지
---------------
선택 :
3
프로그램 종료
```

4.3.3 do-while문

- **입력된 문자열 한 번에 읽기**

```
Scanner scanner = new Scanner(System.in);   // Scanner 객체 생성
String inputString = scanner.nextLine();    // nextLine() 메소드 호출
```

> nextLine( ) 메소드를 호출하면 콘솔에 입력된 문자열을 한 번에 읽을 수 있다.

- **do-while문 예제**

```
import java.util.Scanner;
```

```
public class DoWhileExample {
    public static void main(String[] args) {
        System.out.println("메시지를 입력하세요.");
        System.out.println("프로그램을 종료하려면 q를 입력하세요.");

        Scanner scanner = new Scanner(System.in);
        String inputString;

        do{
            System.out.print(">");
            inputString = scanner.nextLine();
            System.out.println(inputString);
        }while( !inputString.equals("q"));

        System.out.println();
        System.out.println("프로그램 종료");
    }
}
```

실행결과

```
메시지를 입력하세요.
프로그램을 종료하려면 q를 입력하세요.
>hello
hello
>q
q

프로그램 종료
```

4.3.4 break 문

- **예제 코드**

```
public class BreakExample {
    public static void main(String[] args) {
        while(true){
            int num = (int)(Math.random()*6)+1;
            System.out.println(num);
            if(num == 6){
                break;
            }
        }
        System.out.println("프로그램 종료");
    }
}
```

실행결과

```
2
3
6
프로그램 종료
```

> break문은 가장 가까운 반복문만 종료한다.
>
> 중첩된 반복문에서 바깥쪽 반복문까지 종료시키려면 바깥쪽 반복문에 **이름(라벨)을 붙이고**, **"break 이름;"** 을 사용하면 된다.

- **바깥쪽 반복문 종료 예제**

```java
public class BreakOutterExample {
    public static void main(String[] args) {
        Outter: for(char upper='A'; upper<='Z'; upper++){
            for(char lower='a'; lower<='z'; lower++){
                System.out.println(upper + "-" + lower);
                if(lower=='g'){
                    break Outter;
                }
            }
        }
        System.out.println("프로그램 종료");
    }
}
```

실행결과

```
A-a
A-b
A-c
A-d
A-e
A-f
A-g
프로그램 종료
```

4.3.5 continue문

- **예제 코드**

```java
public class ContinueExample {
    public static void main(String[] args) {
        for(int i=1; i<=10; i++){
            if(i%2 != 0){
                continue;
            }
            System.out.println(i);
        }
    }
}
```

실행결과

```
2
4
6
8
10
```

확인문제

1. 조건문과 반복문의 종류를 채워 넣으세요.
    1. 조건문 : if문, switch문
    2. 반복문 : for문 , while문 , do-while 문

2. 조건문과 반복문을 설명한 것 중 틀린 것은 무엇입니까?
    1. if문은 조건식의 결과에 따라 실행 흐름을 달리할 수 있다.
    2. **switch문에서 사용할 수 있는 변수의 타입은 int, double이 될 수 있다. (X , 부동소수점 타입은 쓰면 안 된다.)**
    3. for문은 카운터 변수로 지정한 횟수만큼 반복시킬 때 사용할 수 있다.
    4. break문은 switch문, for문, while문을 종료할 때 사용할 수 있다.

3. for문을 이용해서 1부터 100까지의 정수 중에서 3의 배수의 총합을 구하는 코드를 작성해보세요.

```java
public class Exercise03 {
    public static void main(String[] args) {
        int sum = 0;

        for(int i=1; i<=100; i++){
            if( i % 3 == 0){
                sum += i;
            }
        }
        System.out.println("3의 배수의 합 : " + sum);
    }
}
```

실행결과

```
3의 배수의 합 : 1683
```

4. while문과 Math.random( ) 메소드를 이용해서 두 개의 주사위를 던졌을 때 나오는 눈을 (눈1, 눈2) 형태로 출력하고, 눈의 합이 5가 아니면 계속 주사위를 던지고, 눈의 합이 5이면 실행을 멈추는 코드를 작성해보세요.

```java
public class Example04 {
    public static void main(String[] args) {
        int dice1 = (int)(Math.random()*6 + 1);
        int dice2 = (int)(Math.random()*6 + 1);
        System.out.println("(" + dice1 + "," + dice2 + ")");

        while((dice1 + dice2 != 5)){
            dice1 = (int)(Math.random()*6 + 1);
            dice2 = (int)(Math.random()*6 + 1);
            System.out.println("(" + dice1 + "," + dice2 + ")");
        }
    }
}
```

실행결과

```
(2,6)
(5,4)
(2,6)
(5,1)
(2,6)
(5,3)
(3,6)
(6,5)
(4,6)
(5,4)
(5,3)
(1,4)
```

5. 중첩 for문을 이용하여 방정식 4x + 5y = 60의 모든 해를 구해서 (x, y) 형태로 출력해보세요.

   단, x와 y는 10 이하의 자연수 입니다.

```
public class Exercise05 {
    public static void main(String[] args) {
        for(int i=1; i<=10; i++){
            for(int j=1; j<=10; j++){
                if((4*i + 5*j == 60)){
                    System.out.println("(" + i + "," + j + ")");
                }
            }
        }
    }
}
```

실행결과

```
(5,8)
(10,4)
```

6. for문을 이용해서 실행 결과와 같은 삼각형을 출력하는 코드를 작성해보세요.

```
public class Exercise06 {
    public static void main(String[] args) {
        for(int i=0; i<=5; i++){
            for(int j=0; j<i; j++){
                System.out.print('*');
            }
            System.out.println();
        }
    }
}
```

실행 결과

```
*
**
***
****
*****
```

7. while문과 Scanner를 이용해서 키보드로부터 입력된 데이터로 예금, 출금, 조회, 종료 기능을 제공하는 코드를 작성해보세요.

```
import java.util.Scanner;
```

```java
public class Exercise07 {
    public static void main(String[] args) {
        boolean run = true;

        int balance = 0;

        Scanner scanner = new Scanner(System.in);
        int num;

        while(run){
            System.out.println("--------------------");
            System.out.println("1.예금 | 2.출금 | 3.잔고 | 4.종료");
            System.out.println("--------------------");
            System.out.print("선택> ");

            num = scanner.nextInt();
            if(num == 1){
                System.out.print("예금액> ");
                num = scanner.nextInt();
                balance += num;
            } else if (num == 2){
                System.out.print("출금액> ");
                num = scanner.nextInt();
                balance -= num;
            } else if (num == 3){
                System.out.println("잔고> " + balance);
            } else{
                System.out.println("프로그램 종료");
                break;
            }
        }
    }
}
```

실행결과

```
--------------------
1.예금 | 2.출금 | 3.잔고 | 4.종료
--------------------
선택> 1
예금액> 10000
--------------------
1.예금 | 2.출금 | 3.잔고 | 4.종료
--------------------
선택> 2
출금액> 2000
--------------------
1.예금 | 2.출금 | 3.잔고 | 4.종료
--------------------
선택> 3
잔고> 8000
--------------------
```

```
1.예금 | 2.출금 | 3.잔고 | 4.종료
-------------------
선택> 4
프로그램 종료
```