

3장. 제어문(조건문, 반복문)



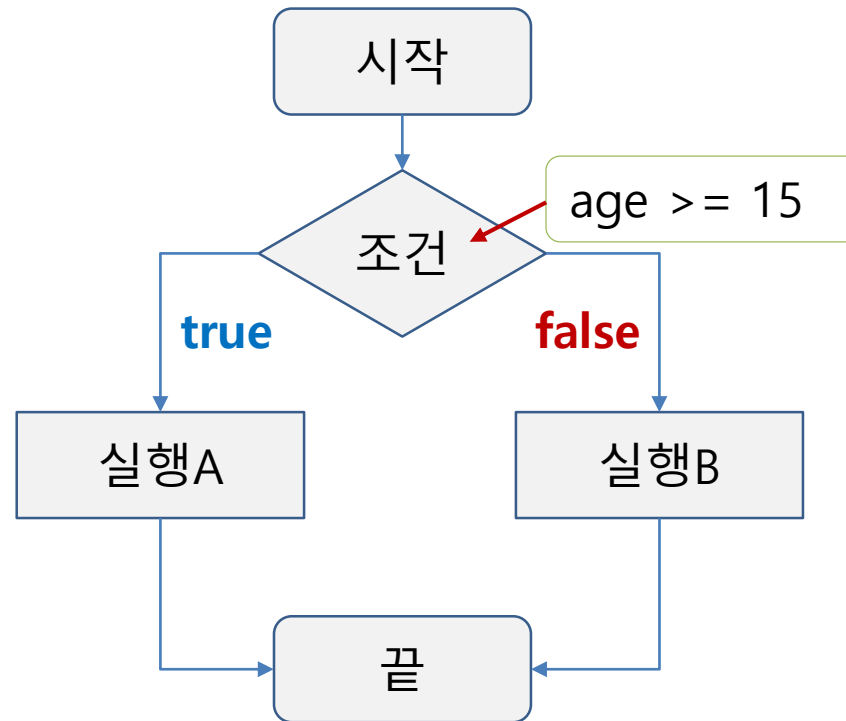
if / loop



조건문(if문)

❖ 조건문

- 주어진 조건에 따라 다른 수행문이 실행되도록 한 프로그래밍 구문
- if문, switch 문이 대표적이다.



조건문(if문)

▪ if문

```
if(조건식){  
    수행문;  
}
```

//조건식이 참이면 수행문 실행

▪ if-else 문

```
if(조건식){  
    수행문1;  
}else{  
    수행문2  
}
```

//조건식이 참이면 수행문1 실행,
아니면 수행문2 실행



조건문(if문)

▪ if문 예제

```
package choice;

public class IfElse1 {

    public static void main(String[] args) {
        // if문
        // 나이가 15세 이상이면 "관람가" 출력
        int age = 16;

        if(age >= 15) {
            System.out.println("관람가");
        }

        System.out.println("나이는 " + age + "세 입니다.");

        // if ~ else문
        // 나이가 15세 이상이면 "관람가"이고, 아니면 "관람불가" 출력
        if(age >= 15) {
            System.out.println("관람가");
        } else {
            System.out.println("관람불가");
        }

        System.out.println("나이는 " + age + "세 입니다.");
    }
}
```



조건문(if문)

■ 조건문과 조건 연산자

- 간단한 if ~ else 조건문은 조건 연산자로 구현할 수 있음
- 두 수 중 큰 값을 출력하는 프로그램

```
if(a > b) {  
    max = a;  
}  
else {  
    max = b;  
}
```



```
max = (a > b) ? a : b;
```

If~else 문

조건 연산자



조건문(if문)

■ 조건문과 조건 연산자

```
// 두 수 중 큰수 비교
Scanner sc = new Scanner(System.in);
System.out.print("첫번째 수 입력 : ");
int n1 = sc.nextInt();
System.out.print("두번째 수 입력 : ");
int n2 = sc.nextInt();

int max;
max = n1 > n2 ? n1 : n2; //조건 연산자 구문

/*
if(n1 > n2) {
    max = n1;
}
else {
    max = n2;
}
*/

System.out.println("두 수 중 큰 수는 " + max + "입니다.");
```



조건문(if문)

■ 숫자와 문자 비교

```
// 숫자 비교
int n1 = 100;
int n2 = 100;

if(n1 == n2){
    System.out.println("두 수가 일치합니다.");
}else {
    System.out.println("두 수가 일치하지 않습니다.");
}

// 문자 비교
String str1 = "apple";
String str2 = "banana";

if(str1.equals(str2)) {
    System.out.println("두 단어가 일치합니다.");
}else {
    System.out.println("두 단어가 일치하지 않습니다.");
}
```



조건문(if문)

▪ if - else if – else 문

```
If(조건 1){  
    수행문1;  
}  
else if(조건 2)  
    수행문2  
}  
else{  
    수행문3  
}
```

//조건 1이 참이면 수행문1 실행, 조건2가 참이면 수행문2
실행, 조건1,2가 모두 거짓이면 수행문3 실행



조건문(if ~ else if ~ else)

▪ 다중 if문 예제

점수	학점
90 ~ 100	A
80 ~ 89	B
70 ~ 79	C
70 미만	F

```
import java.util.Scanner;

public class IfElse3 {
    public static void main(String[] args) {
        //점수에 따른 학점 출력하기
        Scanner sc = new Scanner(System.in);

        System.out.print("점수를 입력하세요: ");
        //int score = 87;
        int score = sc.nextInt(); //점수 입력
        char grade; //학점을 저장할 변수

        if(score >= 90 && score <= 100) {
            grade = 'A';
        }else if(score >= 80) {
            grade = 'B';
        }else if(score >= 70) {
            grade = 'C';
        }else {
            grade = 'F';
        }

        System.out.println("학점은 " + grade + "입니다");
        sc.close(); //Scanner 닫기
    }
}
```



조건문(SWITCH – CASE)

▪ switch ~ case문

조건식의 결과가 정수 또는 문자열의 값이고 그 값에 따라 수행문이 결정될때 사용되는 구문

```
switch(변수){  
    case 변수값:  
        실행문  
        break;  
    ...  
    default:  
        실행문  
        break;  
}
```



조건문(SWITCH – CASE)

▪ switch ~ case문에 문자열 사용하기

```
public class medalColor {  
  
    public static void main(String[] args) {  
  
        String medalColor = "Gold";  
  
        switch(medalColor){  
            case "Gold":  
                System.out.println("금메달 입니다.");  
                break;  
            case "Silver":  
                System.out.println("은메달 입니다.");  
                break;  
            case "Bronze":  
                System.out.println("동메달 입니다.");  
                break;  
            default:  
                System.out.println("메달이 없습니다.");  
                break;  
        }  
    }  
}
```



조건문(SWITCH – CASE)

■ case문 동시에 사용하기

```
// case문 동시 사용
int month = 6;
int day = 0;

switch(month) {
case 1: case 3: case 5: case 7: case 8: case 10: case 12:
    day = 31;
    break;
case 4: case 6: case 9: case 11:
    day = 30;
    break;
case 2:
    day = 28;
    break;
default:
    System.out.println("지원되지 않는 기능입니다.");
    return; //즉시 종료
}
System.out.println(month + "월은 " + day + "일까지 있습니다.");
```



반복문(while문)

● 반복문

- 주어진 조건이 만족할 때까지 수행문을 반복적으로 수행함
- while, for 문이 있음

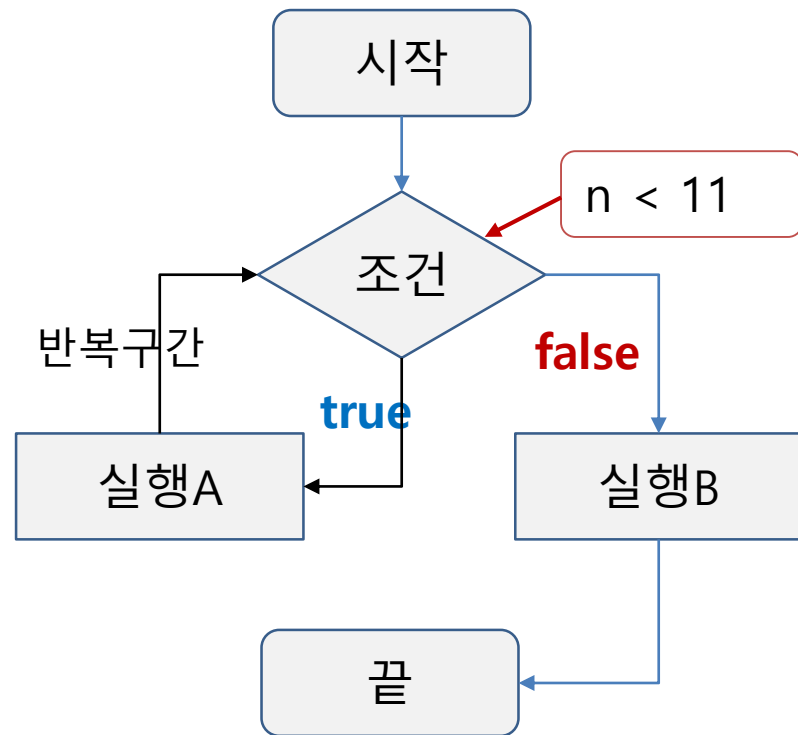
● while 문

- 조건식이 참인 동안 반복 수행

```
while(조건식){  
    수행문1;  
}
```

수행문2;

...



반복문(while문)

- 반복문 예제

```
package loop;

public class WhileEx1 {

    public static void main(String[] args) {
        // "Hello" 10번 출력

        int n = 1;

        while(n <= 10) {
            System.out.println("Hello~ " + n);
            n++;
        }
    }
}
```



반복문(while문)

■ 반복문 예제

```
// 1~10까지의 합계
int x = 1;
int sum = 0;

while(x <= 10) {
    sum += x;
    System.out.println("x=" + x + ", sum=" + sum);
    x++;
}
System.out.println("합계 : " + sum);
}
```



무한반복 - break 문

■ 무한 반복문 - break 문

반복문에서 break 문을 만나면 더 이상 반복을 수행하지 않고 반복문 빠져 나옴

```
while(true){  
    if(조건식){  
        break;  
    }  
    실행문  
}
```



무한반복 – break 문

▪ 반복 조건문 예제

```
public class WhileEx2 {  
  
    public static void main(String[] args) {  
        // 반복 조건문(while ~ if ~ break)  
        // 1~10까지 출력하기  
        int n = 1;  
  
        while(true) {  
            if(n > 10)  
                break;  
            System.out.println(n);  
            n++;  
        }  
    }  
}
```



무한반복 – break 문

▪ 반복 조건문 예제

```
//1~10까지 합계
int x = 1;
int sum = 0;

while(true) {
    if(x > 10)
        break;
    sum += x;
    System.out.println("x=" + x + ", sum=" + sum);
    x++;
}
System.out.println("합계 : " + sum);
}
```



무한반복 – break 문

▪ 반복 조건문 예제

```
public class KeyRepeat {  
    public static void main(String[] args) {  
        //y - "계속 반복합니다", n - "반복을 중단합니다."  
        //그 이외의 키는 "지원하지 않는 키입니다."  
        Scanner sc = new Scanner(System.in);  
  
        while(true) {  
            System.out.print("계속 반복할까요?(y/n) : ");  
  
            String key = sc.nextLine();  
  
            if(key.equals("y") || key.equals("Y")) { //equals() 문자열 비교함수  
                System.out.println("계속 반복합니다.");  
            }  
            else if(key.equals("n") || key.equals("N")) {  
                System.out.println("반복을 중단합니다.");  
                break;  
            }  
            else {  
                System.out.println("지원하지 않는 키입니다.");  
            }  
        }  
        System.out.println("프로그램을 종료합니다.");  
        sc.close();  
    }  
}
```



커피 자동판매기

■ 커피 자동판매기 프로그램

- 동전 500원을 넣으면 커피가 나온다.
- 500원을 초과하면 거스름돈이 나오고 커피가 나온다.
- 500원보다 작으면 커피가 나오지 않음
- 커피는 총 5개이고 모두 소진되면 판매를 중지한다.

돈을 넣어주세요: 500
커피가 나옵니다.
남은 커피의 양은 4개입니다.
돈을 넣어주세요: 600
거스름돈 100원을 돌려주고 커피가 나옵니다.
남은 커피의 양은 3개입니다.
돈을 넣어주세요: 400
돈을 돌려주고 커피는 나오지 않습니다.
남은 커피의 양은 3개입니다.
돈을 넣어주세요: 500
커피가 나옵니다.
남은 커피의 양은 2개입니다.
돈을 넣어주세요: 500
커피가 나옵니다.
남은 커피의 양은 1개입니다.
돈을 넣어주세요: 500
커피가 나옵니다.
남은 커피의 양은 0개입니다.
커피가 다 떨어졌습니다. 판매를 중지 합니다.



커피 자동판매기

```
public class CoffeeMachine {
    public static void main(String[] args) {
        // 커피 자동판매기
        Scanner sc = new Scanner(System.in);

        int coffee = 5;    //커피 총 수량
        while(true) {
            System.out.print("돈을 넣어주세요: ");
            int money = sc.nextInt();
            if(money == 500) {
                System.out.println("커피가 나옵니다.");
                coffee -= 1;
            }
            else if(money > 500) {
                System.out.println("거스름돈 " + (money - 500) + "원을 돌려주고 커피가 나옵니다.");
                coffee -= 1;
            }
            else {
                System.out.println("돈을 돌려주고 커피는 나오지 않습니다.");
            }
            System.out.println("남은 커피의 양은 " + coffee + "개입니다.");

            if(coffee == 0) {
                System.out.println("커피가 다 떨어졌습니다. 판매를 중지 합니다.");
                break;
            }
        }
        sc.close();
    }
}
```



은행 업무 프로그램

■ 은행 업무 프로그램

- 메뉴 화면 만들기 : 1.예금 | 2.출금 | 3.잔액 조회 | 4.종료
- 예금 : 예금액을 입금하면 "정상 처리, 현재 잔액" 출력
- 출금 : 예금액을 입금하면 "정상 처리, 현재 잔액" 출력
- 종료 : 프로그램 종료
- 메뉴에 없는 번호 선택 : 메뉴를 잘못 누름, 다시 입력

```
=====
1. 예금 | 2. 출금 | 3. 잔액 조회 | 4. 종료
=====
선택> 1
예금액>
20000
정상 입금되었습니다. 현재 잔액: 20000
=====
1. 예금 | 2. 출금 | 3. 잔액 조회 | 4. 종료
=====
선택> 2
출금액>
10000
정상 출금되었습니다. 현재 잔액: 10000
=====
1. 예금 | 2. 출금 | 3. 잔액 조회 | 4. 종료
=====
선택> 3
잔액> 10000
=====
1. 예금 | 2. 출금 | 3. 잔액 조회 | 4. 종료
=====
선택> 4
프로그램을 종료합니다.
```



은행 업무 프로그램

■ 은행 업무 프로그램

```
import java.util.Scanner;
public class Banking1 {
    public static void main(String[] args) {
        // 은행 업무 프로그램 - 예금, 출금, 잔액조회
        boolean run = true; //스위치(토글) 변수
        int balance = 0; //잔액
        Scanner sc = new Scanner(System.in);

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

            int selectNum = sc.nextInt(); //메뉴 선택
            int amount = 0; //입금, 출금액

            switch(selectNum) {
                case 1:
                    System.out.println("예금액>");
                    amount = sc.nextInt(); //예금액 입력
                    balance += amount;
                    System.out.println("정상 입금되었습니다. 현재 잔액: " + balance );
                    break;
```



은행 업무 프로그램

■ 은행 업무 프로그램

```
        case 2:
            System.out.println("출금액>");
            amount = sc.nextInt(); //출금액 입력
            balance -= amount;
            System.out.println("정상 출금되었습니다. 현재 잔액: " + balance );
            break;
        case 3:
            System.out.println("잔액> " + balance);
            break;
        case 4:
            System.out.println("프로그램을 종료합니다.");
            run = false; //스위치(바꿈)
            break;
        default:
            System.out.println("메뉴를 잘못 눌렀습니다. 다시 선택해 주세요.");
            break;
    }
}
sc.close();
}
```



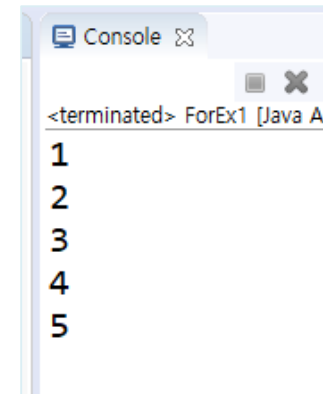
반복문(for문)

■ for 문

- 주로 조건이 횟수인 경우에 사용
- 초기화식, 조건식, 증감식을 한꺼번에 작성

```
for(초기화식; 조건식; 증감식){  
    수행문;  
}
```

```
int n;  
    ① → ② ← ④  
for(n = 1; n <= 5; n++) {  
    ③ ↗  
    System.out.println(n);  
}
```



The screenshot shows a console window titled "Console" with a close button. It displays the output of a Java application named "ForEx1". The output consists of the numbers 1 through 5, each on a new line, indicating that the loop executed five times.

```
<terminated> ForEx1 [Java Ap  
1  
2  
3  
4  
5
```



반복문(for문)

▪ for문

"Hello"를 10번 반복하기

```
int n = 1; //초기값  
  
for(n=1; n <= 10; n++) {  
    System.out.println("hello");  
}
```

1부터 10까지 합계

```
int n;  
int sum;  
for(n=1, sum = 0; n <=10; n++) {  
    sum += n;  
}  
System.out.println(sum);
```



문자 세트 반복

- 문자 세트(유니코드)

- ## - 알파벳과 한글 자음, 모음 출력하기

```
char ch;
for(ch=65; ch<123; ch++) {
    System.out.print(ch + " ");
}

System.out.println();

for(ch=12593; ch<12686; ch++) {
    System.out.print(ch + " ");
}
```

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\] ^ _ ` a b c d e f g h i j k l m n o p q r s t u
ㄱ ㅋ ㆁ ㄴ ㄷ ㄹ ㅁ ㅂ ㅅ ㅇ ㅈ ㅊ ㅋ ㆁ ㄴ ㄷ ㄹ ㅁ ㅂ ㅅ ㅇ ㅈ ㅊ ㅋ ㆁ ㄴ ㄷ ㄹ ㅁ ㅂ ㅅ



구구단

■ 구구단 프로그램

6x1=6
6x2=12
6x3=18
6x4=24
6x5=30
6x6=36
6x7=42
6x8=48
6x9=54

```
//구구단  
int dan = 6;  
int i;  
for(i=1; i<10; i++) {  
    System.out.println(dan + "x" + i + "=" + (dan*i));  
}
```



반복문(중첩 for문)

■ 중첩된 반복문(Nested Loop)

반복문 내부에 또 반복문이 사용됨

	열1	열2	열3	열4	열5
행1					
행2					
행3					
행4					
행5					

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

```
int i, j;  
for(i=1; i<=5; i++) {  
    for(j=1; j<=5; j++) {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```



반복문(중첩 for문)

■ 삼각형 모양의 별 찍기1

hint) 열(column)이 변하는 것에 주목한다.

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

```
int i, j;  
for(i=1; i<=5; i++) {  
    for(j=1; j<i+1; j++) {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```

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

```
for(i = 1; i <= 5; i++) {  
    for(j = 5; j >= i; j--) {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```



반복문(중첩 for문)

■ 삼각형 모양의 별찍기2

hint) 공백과 별로 나눠서 생각

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

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

```
for(i = 1; i <= 5; i++) {
    for(j = 1; j <= 5-i; j++) {
        System.out.print(" "); //공백문자
    }
    for(j = 1; j <= i; j++) {
        System.out.print("*");
    }
    System.out.println(); //행 바꿈
}
```

```
for(i = 1; i <= 5; i++) {
    for(j = 1; j < i; j++) {
        System.out.print(" "); //공백문자
    }
    for(j = 5; j >= i; j--) {
        System.out.print("*");
    }
    System.out.println(); //행 바꿈
}
```



반복문(for문)

■ 중첩된 반복문(Nested Loop)

■ 구구단의 예

```
for(dan = 2; dan <= 9; dan++) {  
    for(times = 1; times <=9; times++) {  
        System.out.println(dan + "x" + times + "=" + dan * times);  
    }  
    System.out.println();  
}
```

각 단에서 1~9를 곱하는 내부 반복문



continue문

▪ continue 문

- 반복문과 함께 쓰이며, 반복문 내부 continue 문을 만나면 이후 반복되는 부분을 수행하지 않고 조건식이나 증감식을 수행함
- 예제 – 1부터 10까지의 자연수 중 4나 8을 제외한 수를 출력하기

```
for(int i=1; i<=10; i++) {  
    if(i==4 || i==8)  
        continue;  
    System.out.println(i);  
}
```

1
2
3
5
6
7
9
10

```
// 홀수만 출력  
for(int i=1; i<11; i++) {  
    if(i%2==0)  
        continue;  
    System.out.println(i);  
}
```



반복문(for문)

■ 중첩된 반복문(Nested Loop)

```
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
```

```
int i, j;
for(i=0; i<5; i++) {
    for(j=1; j<=5; j++) {
        System.out.print(i*5+j + " ");
        //i*열의 끝수+j
    }
    System.out.println();
}
System.out.println();
```



반복문(for문)

■ 중첩된 반복문(Nested Loop)

1	2	3	4	5
7	8	9	10	11
13	14	15	16	17
19	20	21	22	23
25	26	27		

```
public class NestedFor2 {  
  
    public static void main(String[] args) {  
  
        for(int i=0; i<5; i++) {  
            for(int j=1; j<=5; j++) {  
                int num = 6*i+j;  
                if(num > 27)  
                    break;  
                System.out.print(num + " ");  
            }  
            System.out.println();  
        }  
        System.out.println();  
    }  
}
```



자리배치도 프로그램 만들기

입장객 수에 따라 좌석을 배치하는 프로그램을 작성하세요.

(파일이름: Seats.java)

```
입장객 수 입력: 44
좌석 열의 수: 5
좌석1 좌석2 좌석3 좌석4 좌석5
좌석6 좌석7 좌석8 좌석9 좌석10
좌석11 좌석12 좌석13 좌석14 좌석15
좌석16 좌석17 좌석18 좌석19 좌석20
좌석21 좌석22 좌석23 좌석24 좌석25
좌석26 좌석27 좌석28 좌석29 좌석30
좌석31 좌석32 좌석33 좌석34 좌석35
좌석36 좌석37 좌석38 좌석39 좌석40
좌석41 좌석42 좌석43 좌석44
```



자리배치도 프로그램 만들기

■ 자리 배치(Seat Allocation)

```
import java.util.Scanner;

public class Seats {

    public static void main(String[] args) {
        //좌석 줄 수 계산
        //입장객 수, 좌석 열의 개수
        Scanner sc = new Scanner(System.in);
        System.out.print("입장객 수 : ");
        int customNum = sc.nextInt();
        System.out.print("좌석 열 수 : ");
        int colNum = sc.nextInt();

        int rowNum; //줄(행) 수

        if(customNum % colNum == 0) {
            rowNum = customNum / colNum;
        }else { //나머지 인원이 있는 경우 1줄 추가
            rowNum = customNum / colNum + 1;
        }
    }
}
```



자리배치도 프로그램 만들기

■ 자리 배치(Seat Allocation)

```
//좌석 배치 반복
for(int i = 0; i < rowNum; i++) { //줄
    for(int j = 1; j <= colNum; j++) { //열
        int seatNum = i * colNum + j; //좌석 번호
        if(seatNum > customNum)
            break;
        System.out.print("좌석" + seatNum + " ");
    }
    System.out.println();
}
sc.close();
}
```



실습 문제 1 – 반복 조건문

1부터 더했을때 그 합이 100이 넘는 자연수와 합계를 구하세요.

[파일이름: Sum100.java]

👉 실행 결과

```
n = 14  
sum = 105
```



실습 문제 2 – 은행 업무 프로그램

예제로 만들었던 은행 업무 프로그램에서 아래의 개선사항을 구현하세요
[출금시에 잔고가 출금액보다 작은 경우를 처리하세요.]

👉 실행 결과

```
=====
1. 예금 | 2. 출금 | 3. 잔액 조회 | 4. 종료
=====
```

```
선택> 1
```

```
예금액>
```

```
20000
```

```
정상 입금되었습니다. 현재 잔액: 20000
```

```
=====
1. 예금 | 2. 출금 | 3. 잔액 조회 | 4. 종료
=====
```

```
선택> 2
```

```
출금액>
```

```
30000
```

```
잔액이 부족합니다. 다시 입력해 주세요
```

```
=====
1. 예금 | 2. 출금 | 3. 잔액 조회 | 4. 종료
=====
```

```
선택>
```

