2021. 12. 28 인지연

1. 문제1 -1

```
import java.util.Scanner;

public class Ans1 {
    public static void main(String[] args) {
        /*  아래와 같은 등비 수열이 있다.
            1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, ...
            사용자 입력을 통해 원하는 위치의 값을 뽑아내도록 프로그래밍 해보자!
            (1 ~ 32번째 혹은 31번째 항까지만 올바른 결과가 나올 것임) */

        // 2^8 = 1byte = 256개
        // -128 ~ -1 / 0 ~ 127
        // 0을 포함하기 때문에 2^32승도 맨 끝이 홀수로 되어 있음
        // 그래서 2^n을 표현한다고 할 때 실질적으로 전체 비트 - 1까지만 표현이 가능함
        // 결국 2^31승을 표현하지 못하고 2^31 - 1이 최대값이 되는데
        // 그래서 이 문제에서는 2^30승을 표현하기 위해 31번째까지가 최대가 됨
        final int MAX = 31;
        final int START_IDX = 0;
        final int BASE = 2;
```

필요한 상수들 입력
Base가 2인 이유는 2를 제곱해야하는 값이 등비수열이기 때문,

```
arr[0] 값은 1
arr[1] 값은 2
arr[2] 값은 4
arr[3] 값은 8
arr[4] 값은 16
arr[5] 값은 32
arr[6] 값은 64
```

```
arr[26] 값은 67108864
arr[27] 값은 134217728
arr[28] 값은 268435456
arr[29] 값은 536870912
arr[30] 값은 1073741824
arr[31] 값은 2147483647
```

이 말의 뜻은 출력문을 보면 확인하기 편하다.
Arr[31], 즉 출력값 32번째 값은 홀수가 된다.
2의 제곱이니 무조건 뒷자리는 짝수가 되어야하는데.

이부분이 잘 이해가 안갑니다.
비트 -1 이 무슨 뜻일까요?

1. 문제1 -2

```
System.out.print("찾고자하는 수열의 항을 입력해주세요: ");

Scanner scan = new Scanner(System.in);
int idx = scan.nextInt();

int[] seq = new int[idx];

if (idx > MAX) {
    System.out.println("낵아 표현이 으앙돼 ㅠ 프로그램을 종료합니다.");
} else {
    for (int i = START_IDX; i < idx; i++) {
        // Math.pow()는 n승을 계산함
        // Math.pow(x, y) = x^y로 x의 y승을 계산함
        // 즉 Math.pow(2, i)는 2의 i승을 의미함
        // 2^0 = 1, 2^1 = 2, 2^2 = 4 ...
        seq[i] = (int) Math.pow(BASE, i);
        System.out.printf("seq[%d] = %d\n", i, seq[i]);
    }
}
```

스캐너 입력,
idx값에 사용자의 입력 값int형을 받을예정

배열 선언 / 배열 크기를 사용자 값으로 받는다.

흐름 이해하기

만약 idx가 MAX(31)을 넘어간다면 출력값 입력
(실행이 안된다는 내용)

Idx 즉 사용자가 입력한 값이 31 이하라면 코드 실행

for 문 실행 0~사용자 입력값까지
순서마다 배열[i]에 값 입력.
제곱하는 방법
Math. Pow(~,!) → ~을 !번 곱한다.
(base(2)를 i번 곱하니까 등비수열 가능)

1. 문제1 –내가 적은 답

```
Scanner scan = new Scanner(System.in);
System.out.print("값을 입력하시오(1~31사이)");
int num = scan.nextInt();
int [] arr = new int [num];
final int START = 1;

arr[0]=1;
System.out.printf("arr[%d] 값은 %d\n",0,arr[0]);

for(int i = START ; i<num;i++){
    arr[i] = (int)(Math.pow(2,i));
    System.out.printf("arr[%d] 값은 %d\n",i,arr[i]);
}
```

스캐너로 사용자 입력값 받기
그 값 그대로 배열의 크기로 입력

굳이 이렇게 안해도 2의 0승은 1이여서 문제없음

1. 문제2

```java
import java.math.BigInteger;
import java.util.Scanner;

public class Ans2 {
    public static void main(String[] args) {
        /*  1번 문제에서 32번째 항이 21억 정도가 나올 것이다.
            BigInteger를 통해서 50번째 항을 구해보자! */

        final int START_IDX = 0;
        final BigInteger BASE = new BigInteger("2");

        System.out.print("찾고자하는 수열의 항을 입력해주세요: ");

        Scanner scan = new Scanner(System.in);
        int idx = scan.nextInt();

        BigInteger[] seq = new BigInteger[idx];
        seq[START_IDX] = new BigInteger("1");

        for (int i = START_IDX + 1; i < idx; i++) {
            seq[i] = seq[i - 1].multiply(BASE);
            System.out.println("seq[" + i + "] = " + seq[i]);
        }
    }
}
```

BigInteger 이용한 코드

BigInteger로 상수 사용한다.
숫자 입력할때는 "2"로 입력한다.
문자열로 출력하는구나

BigInteger형으로 배열을 만들었다.
그리고 start_idx, 0번 배열은 1로 지정,
bigInteger에서는 제곱을 구할 수 있는게 없나보다.
그래서 미리 빼놓았다.

Seq[1] = seq[1-1] * base(2) = 1 * 2
Seq[2] = seq[2-1] * base(2) = 2 * 2
Seq[3] = seq[3-1] * base(2) = 4 * 2
... 으로도 2의 제곱을 구할 수 있다!

1. 문제2 – 내가 적은답

```java
final int END = 50;
BigInteger[] sequence = new BigInteger[END];


final int START = 1;

sequence[0]= new BigInteger( val: "1");
System.out.printf("arr[%d] 값은 %d\n",0,sequence[0]);

  for(int i = START ; i<END;i++){
      sequence[i] = sequence. // 빅인티져의 제곱하는 법을 모르겠습니다.
        System.out.printf("arr[%d] 값은 %d\n",i,sequence[i]);
```

BigInteger의 제곱 값을 못구해서 결국 답을 찾지 못했다.
꼭 굳이 제곱 매소드가 아니여도 구할 수 있는 답이 있다는 걸 체크하기

1. 문제3 -1

```
public class Ans3 {
    public static void main(String[] args) {
        /*  배열로 로또 문제를 만들어보기!
            실제 로또 확률은 0.00000023으로 1억명중 23명이 당첨됨
            실제값을 사용하기엔 검토 작업이 너무 고통스러우므로 100명 중 5명을 뽑아보도록 하자!
            배열값에 당첨되는 자리를 배치해놓고 사용자가 돌려서 당첨되는지 안되는지를 판정하도록 한다. */

        final int TOTAL = 100;
        final int SELECT = 5;

        boolean[] lottoBox = new boolean[TOTAL];
        int[] selectIdx = new int[SELECT];

        System.out.println("당첨되는 자리를 배치합니다.");

        // 구현 전략
        // 1. 전체 100개 배열을 만듬
        // 2. 당첨 자리 5개 랜덤하게 할당
        // 3. 할당된 자리 중 중복이 존재할 가능성도 있으므로 검사해야함
        //    선택된 인덱스는 0 ~ 99 사이의 랜덤값임
        //    그렇다면 어떻게 이 랜덤 인덱스의 중복 여부를 판정할 것인가 ?
        //    실제 SELECT는 5개이므로
        //    이 SELECT를 활용한 5개 배열에 할당된 랜덤 인덱스를 배치하면 어떨까 ?
        //    그럼 검사를 최악의 경우라고 가정하더라도 최대 4개만 하면 된다.
```

총 100명의 상수값 입력
총 5명 당첨의 상수값 입력

블리언 형 배열도 있다. 블리언 배열은 100명의 크기를 갖는다.
Int 형 배열은 뽑기 5명 당첨의 배열 크기를 갖는다.
두개를 각각 만들어주는 데는 다 이유가 있겠지? 뒤에서 찾아보자

제일 어려운 부분, 확인하러가자

1. 문제3 -2

```
boolean isRealloc = true;
```
(반복적으로 중복검사 하기위해 블린 설정)

```
int lottoIdx = 0;
int allocCnt = 0;
```
로또 값 받기위해 변수 설정
allooCnt는?

```
for (int i = 0; i < SELECT; i++) {      // 총 5개 배치
    while (isRealloc) {
        lottoIdx = (int) (Math.random() * TOTAL);
                        0~99 중 랜덤 값 만들기
        isRealloc = false;

        for (int j = 0; j < allocCnt; j++) {
            if (selectIdx[j] == lottoIdx) {
                System.out.println("중복 발생!");
                isRealloc = true;
                break;
            }
        }
    }

    lottoBox[lottoIdx] = true;
    selectIdx[allocCnt++] = lottoIdx;

    // 이대로 가면 무엇을 놓치게 될까 ? 중복을 놓치게됨
    // 그러므로 중복 발생 여부를 체크하는 루틴이 추가로 필요해짐!

    System.out.println("lottoBox[" + lottoIdx + "] = " + lottoBox[lottoIdx]);

    isRealloc = true;
}
```

랜덤문 뽑아서 배열 For문, select(5)개의 값을 돌린다.
시작이 true여서 lottoinx 값 랜덤설정이 되고,
False로 바꿔서 일단 랜덤값 더 생성하지 않게 한다.

중복을 확인하기 위해 j의 For안으로 들어가
Index[0] = 랜덤값 => 중복발생, 값을 true로 만들고 for문 나가기
True로 바꿔서 i값 for문 다시 돌아가게 한다?
Break를 걸어서 j의 for문은 끝나고 i의 for문으로 가서 중복끝냄?

1. 문제3 -3

```
boolean isRealloc = true;

int lottoIdx = 0;
int allocCnt = 0;

for (int i = 0; i < SELECT; i++) {      // 총 5개 배치
    while (isRealloc) {
        lottoIdx = (int) (Math.random() * TOTAL);

        isRealloc = false;

        for (int j = 0; j < allocCnt; j++) {
            if (selectIdx[j] == lottoIdx) {
                System.out.println("중복 발생!");
                isRealloc = true;
                break;
            }
        }
    }

    lottoBox[lottoIdx] = true;
    selectIdx[allocCnt++] = lottoIdx;

    // 이대로 가면 무엇을 놓치게 될까 ? 중복을 놓치게됨
    // 그러므로 중복 발생 여부를 체크하는 루틴이 추가로 필요해짐!

    System.out.println("lottoBox[" + lottoIdx + "] = " + lottoBox[lottoIdx]);

    isRealloc = true;
}
```

만약 for I = 3이였고, if문이 true로 break가 되어서 for문이 종료되면,
For I 값은 증감되지 않고 다시 I = 3으로 시작되는건가요?

Lottebox[랜덤번호] = true → 로또박스배열에 숫자입력
블린형 배열은 어떤걸 하나요?
배열 안에 값을 입력하는것도 아닌것같은데 왜 쓰는건가요?

lottoBox[lottoIdx]=true는 왜 있어야하는지 모르겠습니다.
그냥 selectIdx[allocCnt++]값만 있어도 값이 나오는게 아닌가요?

1. 문제3 -다른풀이

```java
public class Ans3_2 {
    // 3번 꼼수 버전
    // 배열을 이용해서 처리할 경우
    // 추가적인 작업이 존재할 때
    // 예) 번호 범위 어디는 배당률 얼마 같은 설정이 가능해짐
    // 현재 케이스는 배열이 아니기 때문에 이와 같은 복합 설정은 어려움
    // 어쨌든 문제 풀이는 맞다고 볼 수 있음
    public static void main(String[] args) {
        final int MAX_NUM = 5;

        int[] selectedLotto = new int[MAX_NUM];

        final int MAX = 100;
        final int MIN = 1;

        int range = MAX - MIN + 1;

        for (int i = 0; i < MAX_NUM; i++) {
            selectedLotto[i] = (int) (Math.random() * range + MIN);
            System.out.printf("당첨 번호: %d\n", selectedLotto[i]);
        }
    }
}
```

1~100중에 랜덤수를 고르고
5번 반복해서 당첨번호를 입력한다.

1. 문제3 –내가 푼 코드

```java
Scanner sc = new Scanner(System.in);
System.out.print("번호를 입력해주세요(1~100)");
int num = sc.nextInt();

final int MAX = 100;
final int MIN = 1;
int range = MAX - MIN +1;
int rand = 0;


final int END = 100;
int [] arr =  new int[END];

for(int i=0;i<END;i++){ // 1~100 순서대로 일단 넣기
    arr[i] = i+1;
}

for(int i = 0;i<5;i++){ // 랜덤문 5개만 뽑기, 근데 뽑으면.. 값이 겹칠수도있잖아?? 교환해야할것같은데
    rand = (int)(Math.random()*range+MIN);
    arr[i] = rand;
    System.out.printf("당첨번호는 %d입니다.\n",rand);
}
if(rand == num){
    System.out.println("당첨입니다!");
}else {
    System.out.println("땡");
}
```

For문 사용.. 하지만 뭔가 이상하다
For문으로 랜덤값 5번만 반복하는게 제일 나은것같다.

1. 문제4

```java
public class Ans4 {
    public static void main(String[] args) {
        /*  반 학생이 30명이 있다.
            이들은 모두 시험을 치렀고 모든 학생들은 60점 미만이 없다고 한다.
            이 상태에서 학생들의 점수를 임의로 배치하고
            학급의 평균값을 구해보도록 한다. */

        // 1. 고정값 30
        // 2. 최소값 60
        // 3. 난수 생성
        // 4. 배열 필요

        final int STUDENT_NUM = 30;
        final int MAX = 100;
        final int MIN = 60;

        int[] score = new int[STUDENT_NUM];

        int range = MAX - MIN + 1;
        int sum = 0;

        for (int i = 0; i < STUDENT_NUM; i++) {
            score[i] = (int) (Math.random() * range + MIN);
            sum += score[i];

            System.out.printf("score[%d] = %d\n", i, score[i]);
        }

        System.out.println("반 평균 = " + (float)(sum) / STUDENT_NUM);
    }
}
```

```java
        final int STUDENT = 30;
        int []arr = new int[STUDENT];

        final int MAX = 100;
        final int MIN = 60;
        int range = MAX - MIN +1;
        int rand = 0;
        int sum = 0;

        for(int i=0;i<STUDENT;i++){
            arr[i] = (int)(Math.random()*range+MIN);
            sum += arr[i];
            System.out.printf("%d 학생의 점수는 %d\n",(i+1),arr[i]);
        }
        int avg = sum/STUDENT;

        System.out.printf("학생들의 평균은 ? %d",avg);
    }
}
```

1. 문제5

```java
public class Ans5 {
    public static void main(String[] args) {
        /*  4번 문제에서 평균을 구했으므로 표준편차와 분산을 구하도록 한다.
            힌트: Math.sqrt() - 루트 계산 */

        /* 분산, 표준편차 공식: https://math100.tistory.com/11 참고 */

        final int STUDENT_NUM = 30;
        final int MAX = 100;
        final int MIN = 60;

        int[] score = new int[STUDENT_NUM];

        int range = MAX - MIN + 1;
        float sum = 0;
        float average;

        // 평균을 구하기 위해
        // 랜덤 샘플(각각 1명이 가지는 값) 생성
        for (int i = 0; i < STUDENT_NUM; i++) {
            score[i] = (int) (Math.random() * range + MIN);
            sum += score[i];

            System.out.printf("score[%d] = %d\n", i, score[i]);
        }

        average = sum / STUDENT_NUM;
        System.out.println("평균 = " + average);
```

여기까지는 4번과 똑같다.

Sum초기화
(학생들시험점수)

```java
        sum = 0;

        for (int i = 0; i < STUDENT_NUM; i++) {
            sum += Math.pow(score[i] - average, 2);
        }

        System.out.println("분산 = " + (sum / STUDENT_NUM));
        System.out.println("표준편차 = 루트 분산 = " + Math.sqrt(sum / STUDENT_NUM));
    }
}
```

표준편차
1. 평균 구하기(완)
2. 제곱한 값을 구하기, 각 값에서 평균을 빼고 2제곱
3. 위의 값 모두 더하기
4. 자료개수로 나눈다.(학생수 30명 나누기) ->분산 구한것
5. 제곱근 구하기 (Math.sqrt사용하기)

1. 문제5 - 내가쓴답

```
//표준편차
// 1단계 평균 구하기(완) 2단계 제곱한 값을 구하기, 각 값에서 평균을 빼고 2제곱 /
// 3. 위의 값 모두 더하기/4.자료개수로 나눈다./5. 제곱근 구하기
for(int i=0;i<STUDENT;i++){
    sum2 = (int)(Math.pow((arr[i]-avg),2));
}
int avg2 = sum2 / STUDENT;
int std = (int)(Math.sqrt(avg2));
System.out.printf("분산값은%d\n",avg2);
System.out.printf("표준편차는 %d",std);
```

깜박하고 +=를 안했다.