

2022년 1학기 자바프로그래밍 중간고사 월요일반

- 기능 구현과 상관없이 프로그램 동작 중 예외가 발생하는 경우, 각 항목별 -2점 감점함
- 각 기능별 부분 점수는 없음 (기능에 대한 코드만 있고 동작이 안되면 점수 없음)
- 그 외 과제의 감점 요인과 동일: 컴파일 에러, 주석, UTF-8 등

1. 배열 기능을 확장한 ArrayUtil 프로그램 (6점)

제출파일: midexam1.zip

기존 정수형 배열에 fill, concat, compare, print 기능을 가지는 ArrayUtil 프로그램을 구현하고자 한다. 아래 제시된 설명에 맞게 기능을 구현하시오.

■ 제출파일: ArrayUtil.java, ArrayUtilApp.java

■ ArrayUtil 클래스 (4점, 각 1점)

✓ void fill(int[] a) 메소드 구현

- 랜덤값(1~99 사이)을 생성하고, 크기가 10인 배열을 생성된 랜덤값으로 저장

✓ int[] concat(int[] a, int[] b) 메소드 구현

- 두 배열(a, b)를 합쳐서 새로운 배열을 생성하고 새롭게 생성된 배열을 리턴
- 리턴값: [a₁, a₂, ..., a₁₀, b₁, b₂, ..., b₁₀]

✓ int compare(int[] a, int[] b) 메소드 구현

- 두 배열의 최대값을 계산하고, 최대값을 비교하여 아래 결과를 리턴
- 1: a의 최대값 > b의 최대값
- 0: a의 최대값 == b의 최대값
- -1: a의 최대값 < b의 최대값

✓ void print(int[] a) 메소드 구현

- 배열의 인덱스 및 원소를 포맷에 맞게 화면에 출력

0	1	2	3	4	5	6	7	8	9
[34,	79,	27,	7,	46,	81,	51,	45,	31,	11]

■ ArrayUtilApp 클래스 (2점)

✓ main()함수가 있는 클래스

- 크기가 10인 두 개의 배열 생성
- 아래 실행 결과의 순서에 맞게 ArrayUtil 클래스의 메소드 호출

실행결과:

```
fill(a) 수행
< 배열 a 인덱스 및 내용 출력 >
  0  1  2  3  4  5  6  7  8  9
[38, 45, 22, 43, 1, 2, 22, 74, 73, 67]

fill(b) 수행
< 배열 b 인덱스 및 내용 출력 >
  0  1  2  3  4  5  6  7  8  9
[64, 94, 99, 76, 2, 67, 76, 17, 65, 80]

concat(a, b) 수행 및 배열 c 생성
< 배열 c 인덱스 및 내용 출력 >
```

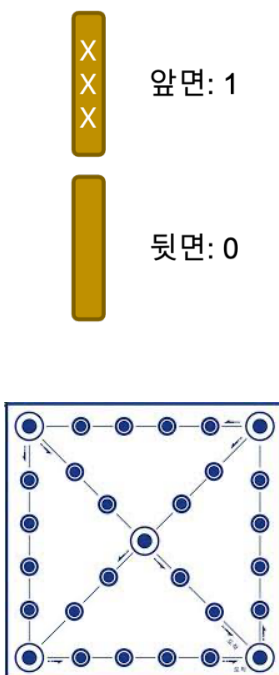


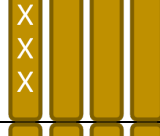


0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
[38,	45,	22,	43,	1,	2,	22,	74,	73,	67,	64,	94,	99,	76,	2,	67,	76,	17,	65,	80]

compare(a, b) 수행
a의 최대값: 74, b의 최대값: 99
a < b

2. 자바 인터페이스 구현을 이용한 윷놀이 게임 프로그램 (19점)

제출파일: midexam2.zip

홍부와 놀부가 윷놀이를 하는데, 각자 4개의 윷을 1번씩 번갈아 던져서 20점 이상의 점수가 먼저 나오는 사람이 승리를 한다. 윷가락을 던져 나온 점수가 "윷(4점)"이나 "모(5점)"가 나오는 경우, 동일한 사람이 한 번 더 던질 수 있다. 주어진 인터페이스를 사용하고 필요한 기능들을 추가로 구현하여 프로그램을 작성하시오.

 <p>앞면: 1</p> <p>뒷면: 0</p>		배열의 값: 1111 (모, 점수: 5점)
		배열의 값: 0000 (윷, 점수: 4점)
		배열의 값: 1000 (걸, 점수: 3점)
		배열의 값: 1100 (개, 점수: 2점)
		배열의 값: 1110 (도, 점수: 1점)

- 제공된 인터페이스를 사용하지 않거나 임의로 변경할 경우, 0점 처리함

■ 제출파일: YutPlayer.java, YutPlayApp.java

■ YutPlayer.java (13점)

- ✓ 주어진 YutInterface를 구현함
- ✓ 생성자에 사람 이름을 입력 받음
- ✓ **int castYut()** 메소드 (4점)
 - 4개 랜덤 숫자(0, 1)를 생성해서 크기가 4인 정수형 배열에 저장
 - 생성된 숫자에 따라 점수 계산 및 누적 점수 계산

- 배열을 사용하지 않는 경우, 해당 점수 없음
- ✓ **int getTotalScore()** 메소드 (1점)
 - 현재까지 누적된 점수 리턴
- ✓ **int compareTo(Object obj)** 메소드 (3점)
 - Upcasting/downcasting 사용 (미사용시: -2점)
 - 두 사람의 누적 점수를 비교하여 승패를 계산
- ✓ **void displayResult()** 메소드 (3점)
 - 경기가 종료된 다음, 각 점수(도, 개, 걸, 윷, 모)의 발생 비율을 %로 계산
 - 출력시 자리 수 맞춤
- ✓ **String toString()** 메소드 구현 (2점)
 - 객체 출력을 위한 오버라이딩
 - 출력 예) "홍부 [0 0 0 1] 걸 (3점/총 5점)"
- **YutPlayApp.java** (6점)
 - ✓ main() 함수가 있는 클래스
 - ✓ 두 명의 YutPlayer객체를 생성하고 생성자에 "홍부", "놀부" 입력
 - ✓ 두 명중 한 명의 점수가 20점 이상 될 때까지 반복 (2점)
 - ✓ "모"나 "윷"이 나오면, 다시 한 번 던짐 (2점)
 - ✓ 한 번씩 윷을 던질 때마다, 그 결과값을 화면에 출력 (1점)
 - ✓ 게임이 종료되면 승패 결과를 화면에 출력하고, 각 점수별 발생 비율을 %로 출력 (1점)

```
public interface YutInterface {
    /**
     * 1회 윷을 던져서 나온 값을 리턴
     * 1111: 모 : 5
     * 0000: 윷 : 4
     * 1000: 걸 : 3
     * 1100: 개 : 2
     * 1110: 도 : 1
     */
    public abstract int castYut();

    /**
     * 현재까지 누적 점수를 리턴
     * @return
     */
    public abstract int getTotalScore();

    /**
     * 두 사람의 전체 누적 점수를 비교
     * @param obj (Upcasting, downcasting 사용)
     * @return
     * . player1 > player2, return 1
     * . player1 == player2, return 0
     * . player1 < player2, return -1
     */
}
```

```

    *
    */
    public abstract int compareTo(Object obj);

    /**
     * 윷을 던져서 나온 값들의 통계값을 화면에 출력
     * - 각 점수별 나온 비율(%)
     */
    public abstract void displayResult();
}

```

실행 결과

실행 결과 #1

```

흥부 [0 0 0 0] 윷 (4점/총 4점) --->
흥부 [1 1 1 0] 도 (1점/총 5점) --->
                                     <--- 놀부 [0 0 0 0] 윷 (4점/총 4점)
                                     <--- 놀부 [1 1 0 1] 도 (1점/총 5점)
흥부 [1 0 0 1] 개 (2점/총 7점) --->
                                     <--- 놀부 [0 0 0 0] 윷 (4점/총 9점)
                                     <--- 놀부 [1 1 0 0] 개 (2점/총 11점)
흥부 [1 1 0 1] 도 (1점/총 8점) --->
                                     <--- 놀부 [1 1 1 1] 모 (5점/총 16점)
                                     <--- 놀부 [1 0 1 0] 개 (2점/총 18점)
흥부 [0 0 0 1] 걸 (3점/총 11점) --->
                                     <--- 놀부 [0 1 0 0] 걸 (3점/총 21점)

```

흥부: 11, 놀부: 21 ==> 놀부 승리

	도	개	걸	윷	모	단위(%)
흥부	40.0	20.0	20.0	20.0	0.0	
놀부	14.3	28.6	14.3	28.6	14.3	

실행 결과 #2

```

흥부 [1 0 0 1] 개 (2점/총 2점) --->
                                     <--- 놀부 [0 0 1 1] 개 (2점/총 2점)
흥부 [0 0 1 0] 걸 (3점/총 5점) --->
                                     <--- 놀부 [0 1 1 0] 개 (2점/총 4점)
흥부 [1 0 0 0] 걸 (3점/총 8점) --->
                                     <--- 놀부 [1 0 0 0] 걸 (3점/총 7점)
흥부 [1 1 0 1] 도 (1점/총 9점) --->
                                     <--- 놀부 [1 1 1 0] 도 (1점/총 8점)
흥부 [0 0 0 0] 윷 (4점/총 13점) --->
흥부 [1 1 1 1] 모 (5점/총 18점) --->
흥부 [0 0 1 0] 걸 (3점/총 21점) --->

```

흥부: 21, 놀부: 8 ==> 흥부 승리

도 개 걸 윗 모 단위(%)

흥부 14.3 14.3 42.9 14.3 14.3

도 개 걸 윗 모 단위(%)

놀부 25.0 50.0 25.0 0.0 0.0
