iOS 개발 스쿨

4차 테스트

Giftbot

안내

> 진행 순서는 다음과 같습니다. 주어진 문제를 확인하고,

(1) A4 용지에 손코딩으로 풀어서 클래스 매니저에게 모두 제출한 후에

(2) 플레이그라운드를 통해 손코딩으로 풀었던 내용이 잘 동작하는 지 점검하고

(3) 문제가 있을 경우 요구사항에 맞게 최종 구현 완료하시면 됩니다

> 플레이그라운드 내에는 총 6개의 서브페이지가 있으며,

손코딩 내용이라는 페이지에는 손 코딩 결과를 그대로 옮겨적고 수정하지 마시고

최종 결과라고 적혀 있는 페이지에 최종 구현 내용을 작성하시면 됩니다,

> 진행 시간은 오후 1시 ~ 4시까지입니다.

1번 문제

문제 1

정아와 창근 둘이서 번갈아가며 탁자 위의 바둑알을 가져가는 게임을 하고 있다.

탁자 위에는 바둑알 N개가 놓여져 있고, 자신의 턴마다 (1개, 3개, 4개) 중 하나를 택해 그만큼 가져갈 수 있다. 자신의 차례에 마지막에 남겨진 바둑알을 가져가는 사람이 승자가 된다.

바둑알의 개수 N이 입력으로 주어졌을 때, 두 부부가 이상적으로 게임을 진행하면 최종적으로 이기게 되는 사람을 출력하는 프로그램을 구현. (게임은 정아가 먼저 시작한다)

입력: N (= 바둑알의 개수) (1 ≤ N ≤ 10000)

출력: 승자의 이름 (정아 또는 창근)

e.g.

N = 5 ==> 정아 / N = 19 ==> 정아 / N = 92 ==> 정아

N = 37 ==> 창근 / N = 350 ==> 창근

2번 문제

문제 2

숫자를 파라미터로 입력받았을 때, 그 숫자가 3의 배수면 Fizz, 5의 배수면 Buzz,

3과 5의 공배수인 경우에는 FizzBuzz 라고 출력하는 함수 구현

(다음 페이지에 실제 문제 안내문 참고)

func fizzBuzz(N: Int) -> String { } (입력값 : 1 <= N <= 100)

아래 1부터 100까지의 결과 이미지 참고

```
["1", "2", "Fizz", "4", "Buzz", "Fizz", "7", "8", "Fizz", "Buzz", "11", "Fizz", "13", "14", "FizzBuzz", "16",
"17", "Fizz", "19", "Buzz", "Fizz", "22", "23", "Fizz", "Buzz", "26", "Fizz", "28", "29", "FizzBuzz", "31",
"32", "Fizz", "34", "Buzz", "Fizz", "37", "38", "Fizz", "Buzz", "41", "Fizz", "43", "44", "FizzBuzz", "46",
"47", "Fizz", "49", "Buzz", "Fizz", "52", "53", "Fizz", "Buzz", "56", "Fizz", "58", "59", "FizzBuzz", "61",
"62", "Fizz", "64", "Buzz", "Fizz", "67", "68", "Fizz", "Buzz", "71", "Fizz", "73", "74", "FizzBuzz", "76",
"77", "Fizz", "79", "Buzz", "Fizz", "82", "83", "Fizz", "Buzz", "86", "Fizz", "88", "89", "FizzBuzz", "91",
"92", "Fizz", "94", "Buzz", "Fizz", "97", "98", "Fizz", "Buzz"]
```

문제 2

1) FizzBuzz 문제를 if문을 이용해 구현

2) FizzBuzz 문제를 switch 문을 이용해 구현

3) 1번 또는 2번에서 구현한 함수의 내용을 클로저로 변환

4) 1부터 100까지의 모든 숫자를 FizzBuzz에 차례대로 전달해 이전 페이지에 주어진 결과 예시처럼 차례대로 출력하되 고차 함수 중 하나인 map을 이용할 것

3번 문제

문제 3

삽입 정렬 (Insertion Sort) 을 이용해 다음의 숫자들을 오름차순 정렬되도록 구현

> 배열을 순회하며 현재 위치와 그보다 작은 인덱스의 값들을 비교해

적절한 위치에 삽입해 나가는 방식

```
var inputCases = [
  [],
  [1],
  [1, 1, 2, 2, 3, 3, 1, 2, 3],
  [14, 10, 9, 7, 6, 5, 3, 2, 1],
  [1, 2, 3, 5, 6, 7, 9, 10, 14],
  [5, 6, 1, 3, 10, 2, 7, 14, 9],
]
```