5.

import java.util.ArrayList;  
import java.util.Scanner;  
public class powCycle {  
 public static int powNum (int target){  
 int powResult = 0;  
 while(target>0){ // 모든 자리를 구할 때 까지  
 powResult += (int) Math.*pow*(target%10,2); //타겟의 일의자리를 받아온다  
 target /= 10; // 매 반복마다 10을 나눠준다  
 }  
 return powResult;  
 }  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 ArrayList<Integer> cycle = new ArrayList<>();  
 int target = sc.nextInt();  
 int answer = 0; // 반복 횟수 cnt 와 답을 저장할 answer를 0으로 초기화한다  
 cycle.add(target); // 사이클에 타겟값을 넣어준다  
 while(true){  
 target = *powNum*(target); // 각자리 제곱값을 더한것을 타겟으로 재선언한다  
// System.out.println(cycle.size()+" 번쨰 "+target);  
 if(cycle.contains(target)){  
 // 만약 사이클에 타겟이 이미 있는 상태라면, 사이클에 저장된 값이 다시 등장했을경우 그 이전까지의 길이만큼 반복된다  
 answer = cycle.size()-cycle.indexOf(target); // 현재까지 들어가있는 사이클 안의 수의 개수가 반복 주기가 된다  
 break;  
 }  
 if(cycle.size()>=100) break; // 100개 이상의 수가 저장되었음에도 탈출하지 못했다면 탈출  
 cycle.add(target); // 사이클에 타겟값을 넣어준다  
 }  
 if(answer==0) System.*out*.println("100개 이상");  
 else System.*out*.println(answer);  
 for (Integer integer : cycle) {  
 System.*out*.println(integer);  
 }  
 }  
}

리뷰: 처음에 너무 어렵게 생각하여 스택에 하나씩 쌓고 길이를 1~스택길이만큼 반복하며 연속하는 수열인지 확인하려 했으나 하면서도 너무 비효율적이라 느껴져 고민을 조금 하였던것같다. 알고리즘이 수학적 지식을 기반으로 하다 보니 접근 방식을 달리하여 패턴을 찾는것으로 해답을 찾을 수 있었다. 중복되는 수가 하나라도 존재한다면 그 수열은 사이클이 있는 수열이라고 판단할 수 있었다. 그래서 차례대로 리스트에 요구사항에 만족하는 수를 넣고 contains 를 활용하여 스택안에 포함된 수 가 재등장할시 재등장한 수의 인덱스에 스택의 길이의 차이를 통해

연속하는 배열의 길이를 반환하도록 하였다. 그리고 리스트의 크기가 100을 초과할 시 만족하지 않기에 100개 이상이라고 출력을 주었다. 이 문제를 통해 알고리즘에 다가가는 방식을 달리해야겠다 느꼈고 사고하는 방식을 전환할 필요성을 느꼈다.

6.

public class ArrayMinMaxMid {  
 public static void Solve(int[] ar){  
 for (int i = 1; i < ar.length; i++) {  
 for (int j = i; j >= 0 ; j--) {  
 if (ar[i]>ar[j]){  
 int temp;  
 temp = ar[i];  
 ar[i] = ar[j];  
 ar[j] = temp;  
 }  
 else break;  
 }  
 }  
 System.*out*.println("제일 큰 값은 "+ar[ar.length-1]+" 입니다.");  
 System.*out*.println("제일 작은 값은 "+ar[0]+" 입니다.");  
 System.*out*.println("중간값은 "+ar[ar.length/2]+" 입니다.");  
 }  
  
 public static void main(String[] args) {  
 int[] ar = {1,2,3,4,5};  
 *Solve*(ar);  
 }  
}

리뷰 : for 문을 한번 만 써서 구하는 방법을 오랫동안 고민해보았지만 마땅히 떠오르는것이 없어 for, if, else 를 모두 쓰는 선택 정렬을 사용하여 배열을 정렬해주었다. 배열의 길이가 짝수인 경우도 생각해보았으나 애초에 짝수인 경우 중간값을 구하기 모호하다 판단되어 길이가 홀수인 배열을 입력받는것을 전제로 구현하였다. 복잡도 면에서 비효율적이라 판단되어 추후에 고민을 더 해봐야겠다고 느꼈다.