Coding Solutions
Solving problems

- Understand the problem
- Categorize the problem
- Design a solution
- Code the solution
- Debug the solution
Solving problems

- Understand the problem
- Categorize the problem
- Design a solution
- **Code the solution (today's focus)**
- Debug the solution
Coding solutions

- Keep your files organized
  - Subfolder for each problem you solve
- Use the automated test framework
  - Or run interactively for quick tests
- Learn and customize your editor
  - Syntax highlighting, line numbers
  - Automatic indentation, tabs/spaces preferences
  - Key combinations, search/replace
- Practice typing!
  - Goal: speed AND precision
Coding solutions

• Write I/O code first
  – Can help you understand the problem better
  – Gives you a psych boost (got something working!)
  – You can't test anything without getting I/O right

• Programming contest I/O
  – Copy sample input/output from problem description
  – Read from standard input (System.in)
  – Write to standard output (System.out)
  – Use standard error for debugging output (System.err)
  – Output usually must match expected output EXACTLY!!!
I/O patterns

• Common input patterns
  – Number of cases given
  – Stop at signal value
  – Multiple data sets
  – Multiple values per line
  – Integers and floating-point numbers

• Common output patterns
  – Single answer
  – Multiple quantities
  – Floating-point (must be accurate to X digits)
Scanner in = new Scanner(System.in);

int n = in.nextInt();   // data count
in.nextLine();          // discard newline

for (int i = 0; i < n; i++) {

    int x = in.nextInt();   // next number
    in.nextLine();          // discard newline

    ...

    ...
Stop at signal value

Scanner in = new Scanner(System.in);

int x = in.nextInt(); // next number
in.nextLine(); // discard newline

while (x != 0) {
    x = in.nextInt(); // next number
    in.nextLine(); // discard newline

    ...
}
Multiple data sets

Scanner in = new Scanner(System.in);

int n = in.nextInt();   // next data count
in.nextLine();          // discard newline

while (n != 0) {
    for (int i = 0; i < n; i++) {

        int x = in.nextInt();   // next number
        in.nextLine();          // discard newline

        // TODO: do something with x here
    }

    n = in.nextInt();       // next data count
    in.nextLine();          // discard newline

    ...
}
Scanner in = new Scanner(System.in);

// read whitespace-separated line and parse into array
String[] data = in.nextLine().split("\s+");

// convert first item to integer
int x = Integer.parseInt(data[0]);

// convert second item to floating-point
double y = Double.parseDouble(data[1]);
// single integer
System.out.printf("%d", x);

// multiple integers, padded to six characters each
System.out.printf("%6d %6d", x, y);

// float with two decimal digits
System.out.printf("%.2f", x);

// float with two decimal digits, padded to 8 chars
System.out.printf("%8.2f", x);

References:
- https://docs.oracle.com/javase/tutorial/java/data/numberformat.html
Solved problem

Oddities

Some numbers are just, well, odd. For example, the number 3 is odd, because it is not a multiple of two. Numbers that are a multiple of two are not odd, they are even. More precisely, if a number $n$ can be expressed as $n = 2 \times k$ for some integer $k$, then $n$ is even. For example, $6 = 2 \times 3$ is even.

Some people get confused about whether numbers are odd or even. To see a common example, do an internet search for the query “is zero even or odd?” (Don’t search for this now! You have a problem to solve!)

Write a program to help these confused people.

Input

Input begins with an integer $1 \leq n \leq 20$ on a line by itself, indicating the number of test cases that follow. Each of the following $n$ lines contain a test case consisting of a single integer $-10 \leq x \leq 10$.

Output

For each $x$, print either ‘$x$ is odd’ or ‘$x$ is even’ depending on whether $x$ is odd or even.

Sample Input 1

3
10
9
-5

Sample Output 1

10 is even
9 is odd
-5 is odd
Scanner in = new Scanner(System.in);

int n = in.nextInt(); // # of numbers

for (int i = 0; i < n; i++) {

    int x = in.nextInt(); // next number

    System.out.print(x);

    // x is even if the remainder when divided by 2 is zero
    if (x % 2 == 0) {
        System.out.println(" is even");
    } else {
        System.out.println(" is odd");
    }
}
Solved problem

```java
Scanner in = new Scanner(System.in);

int n = in.nextInt();  // # of numbers

for (int i = 0; i < n; i++) {
    int x = in.nextInt();  // next number

    System.out.print(x);
    System.out.println(x % 2 == 0 ? " is even" : " is odd");
}
```