

Print all digits from end

Problem

Submissions

Leaderboard

Discussions

Take n as an integer input from the user, then you have to print the digits of the number starting from the end to the first digit of the number where each digit should be printed in a separate line.

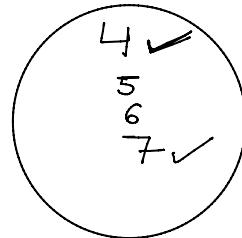
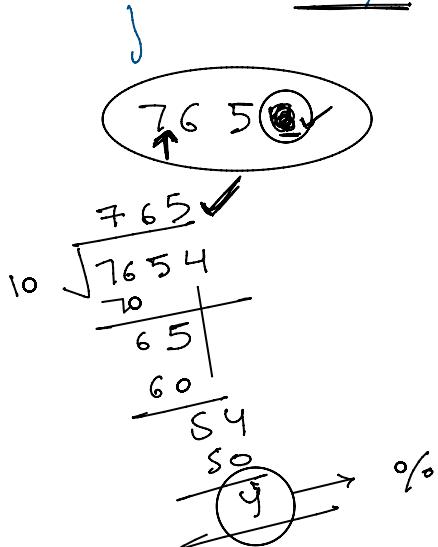
Sample Input 0



Sample Output 0

4
5
6
7

while ($n > 0$) {
 int ans = $n \% 10$ → fetch the last digit
 print (ans); ✓
 $n = n / 10$; → remove
 your last
 digit.
}



Scanner scn = new Scanner(System.in);
int n = scn.nextInt();
while(n > 0){
 int ans = n % 10;
 System.out.println(ans);
 n = n / 10;
}

Output

9 ✓
8 ✓
7 ✓
6 ✓

Memory
n = 6789
ans = 9876

10 $\sqrt{6789}$

60
18
20
89
80

7 ✓
6 ✓

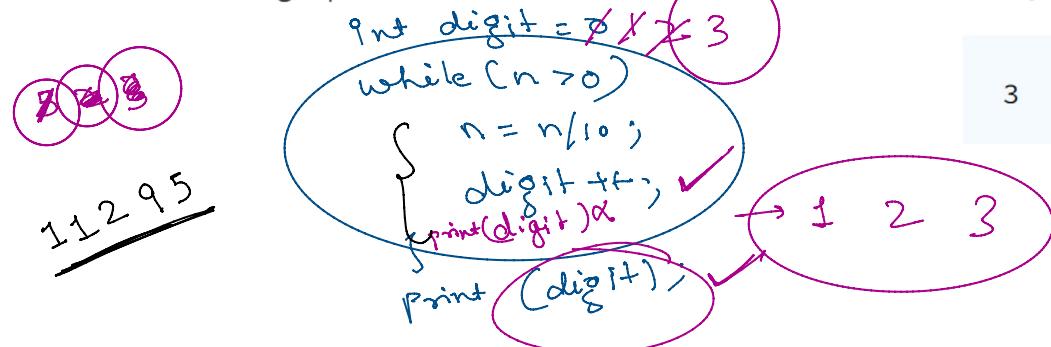
$$\begin{array}{r} 70 \\ 89 \\ -80 \\ \hline 9 \end{array}$$

GKSTR46 Number of Digits

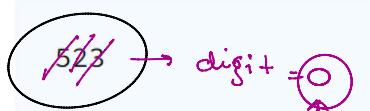
Problem Submissions Leaderboard Discussions

Take an integer N as input.

Print the number of digits present in N.



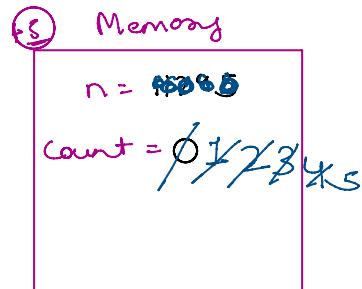
Sample Input 0



Sample Output 0 identify

3

```
Scanner scn = new Scanner(System.in);
int n = scn.nextInt();
int count = 0;
while(n > 0){ → true
    n = n / 10; ←
    count++; ←
}
System.out.println(count); → 5
```



Print total steps when n/2

Problem Submissions Leaderboard Discussions

Take an integer input n and then keep on dividing n by 2, till the time n is greater than equal to 1

Each time you divide n by 2, increment steps by 1.

Print the total number of steps in end.

$n \geq 1$

Print the total number of steps in end.

```
Scanner scn = new Scanner(System.in);
int n = scn.nextInt(); → 20
int steps = 0; ✗ 2 ✗ 3 ✗ 4 ✗ 5
while(n > 0){ 20 > 0
    n = n / 2;   10 > 0
    steps++;
}
System.out.println(steps);
```

Sample Input 1

20 10 5 2 1 0

Sample Output 1

5 ✓

Print steps and update maximum

Problem

Submissions

Leaderboard

Discussions

Take n as input from the user. Then you will be given a list of n positive integers, each time you find a new maximum value, you have to increment the steps by 1.

Take steps as 0 initially and maximum value as -100 in the starting.

In the end print the number of steps performed.

steps = 0 ✗ 2 ✗ 3 ✗ 4 ✗ 5 ✗ 6
max = -100 ✗ 2 ✗ 3 ✗ 4 ✗ 5 ✗ 6

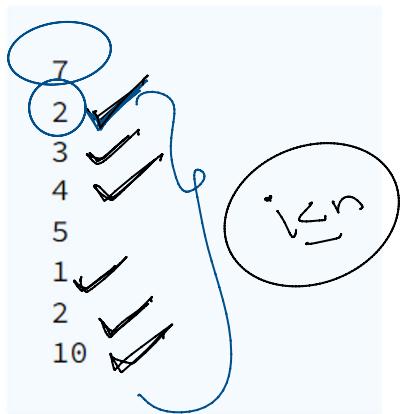
6
1
2
3
4
5
6 ✓

Sample Output 0

6

Sample Input 1

Sample Input 1



Sample Output 1

5

Step = ~~8~~~~7~~~~2~~~~3~~~~4~~~~5~~
max = ~~-100~~ ~~7~~~~3~~~~4~~~~5~~~~10~~

int n → input

max = -100

step = 0

int i = 0;

for (i → n)

while

int num → scn.nextInt()

~~if (num > max)~~

max = num

step++ ✓

} print(steps);

```
Scanner scn = new Scanner(System.in);
int n = scn.nextInt();
int max = -100;
int step = 0;
int i = 1 ;
while( i<=n ){ i<=5
    int num = scn.nextInt(); → q
    if(num > max){ ↗
        max = num;
        step++;
    }
}
System.out.println(step);
```

Memory

n = 5

max = -100

step = 1

i = 12345

num = 72345

Sample Input 4

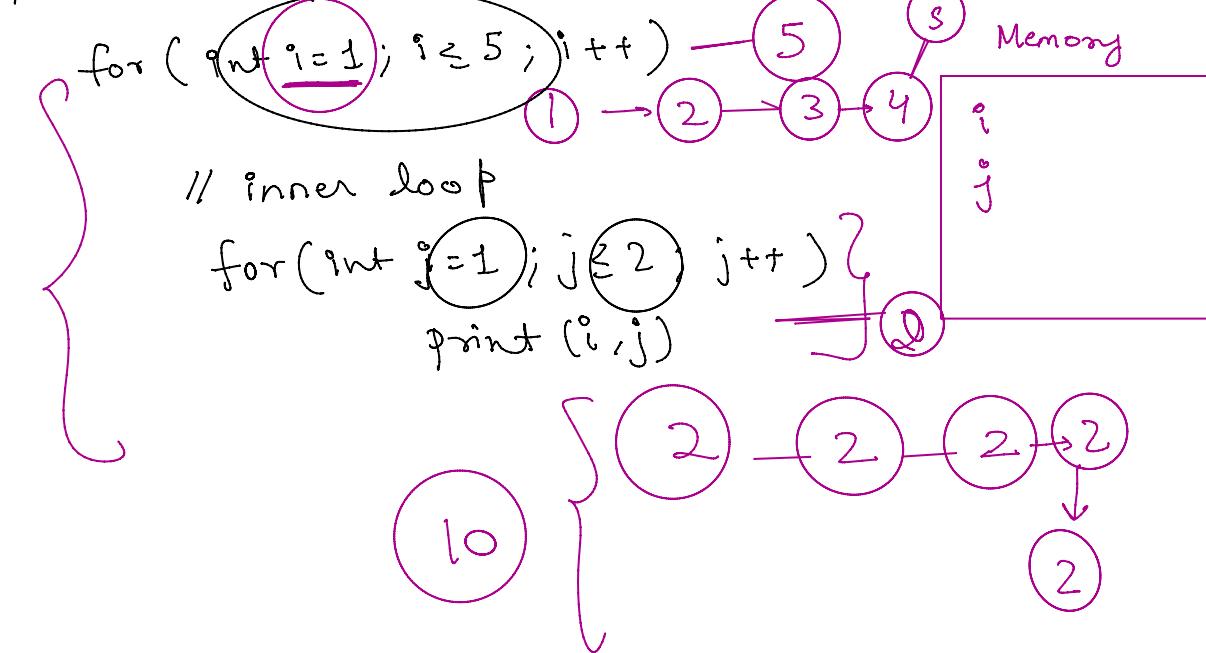


Sample Output 4

✓ 1

Nested for-loops

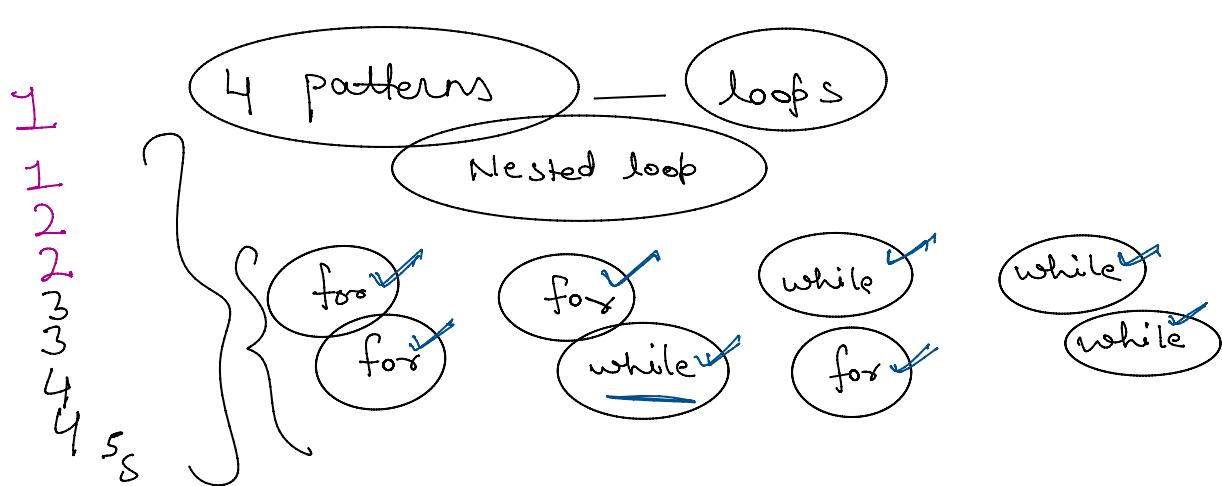
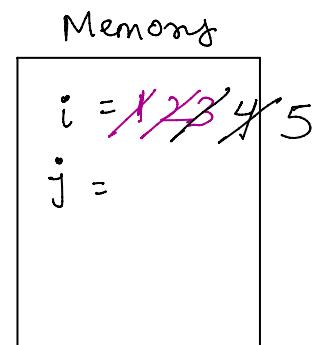
// outer loop



```

for(int i = 1 ; i<=5 ; i++){
    for(int j = 1 ; j<=2 ; j++){
        System.out.println(i);
    }
}

```



```

public class Main {
    public static void main(String[] args) {
        int week = 3;
        int days = 7;
        for(int i = 1; i <= week ; i++){
            System.out.println("week: " + i);
            for(int j = 1 ; j <= days; j++){
                System.out.println(" Day : " + j);
            }
        }
    }
}

```

The handwritten annotations explain the execution flow of the nested loops:

- Outer Loop (week):** Circled with a pink oval. The value 3 is circled at the end of the loop body. A pink arrow points from the value 3 to the circled value 3.
- Inner Loop (days):** Circled with a pink oval. The value 7 is circled at the start of the loop body. A pink arrow points from the circled value 7 to the circled value 7 at the end of the inner loop body.
- Print Statements:** The output of each loop iteration is annotated with green numbers representing the current values of *i* and *j*. For example, one annotation shows "week: 1" followed by "Day : 1" with a green checkmark.

Pattern 1 - Print Stars in same line

Problem Submissions Leaderboard Discussions

Sample Input 0

Take an integer input n and print n stars in the same straight line.

5

for (i = n)
print ("*");

Sample Output 0

```

Scanner scn = new Scanner(System.in);
int n = scn.nextInt();
for(int i = 1 ; i <=n ; i++){
    System.out.print("*");
}

```

Pattern 2 - Print n x 12 star rectangle

Sample Input 0

6

Pattern 2 - Print $n \times 12$ star rectangle

Problem

Submissions

fixed
input

Leaderboard

Discussions

6

Sample Output 0

Take n as an integer input and then print a star rectangle such that each line has n stars.

Also, there are 12 such lines.

int $n \rightarrow$ input

for ($i = 1 \rightarrow 12$) // rows

 for ($i = 1 \rightarrow n$) // columns

 for ($i \rightarrow 12$)

 for ($i \rightarrow n$)

 System.out.print("*");

 } println();



/* Enter your code here. Read input from

Scanner scn = new Scanner(System.in);

j=5

int n = scn.nextInt();

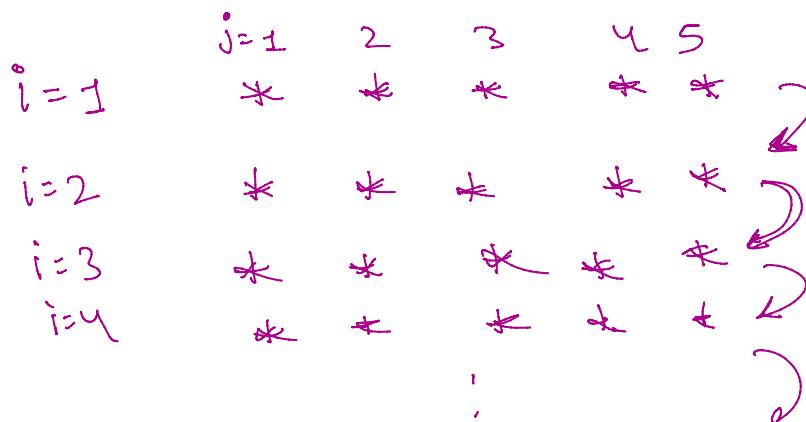
for(int i = 1; i<=12 ; i++){ // row

 for(int j = 1; j<=n ; j++){ // col

 System.out.print("*");

 } System.out.println();

}



ok

$$i = 12$$