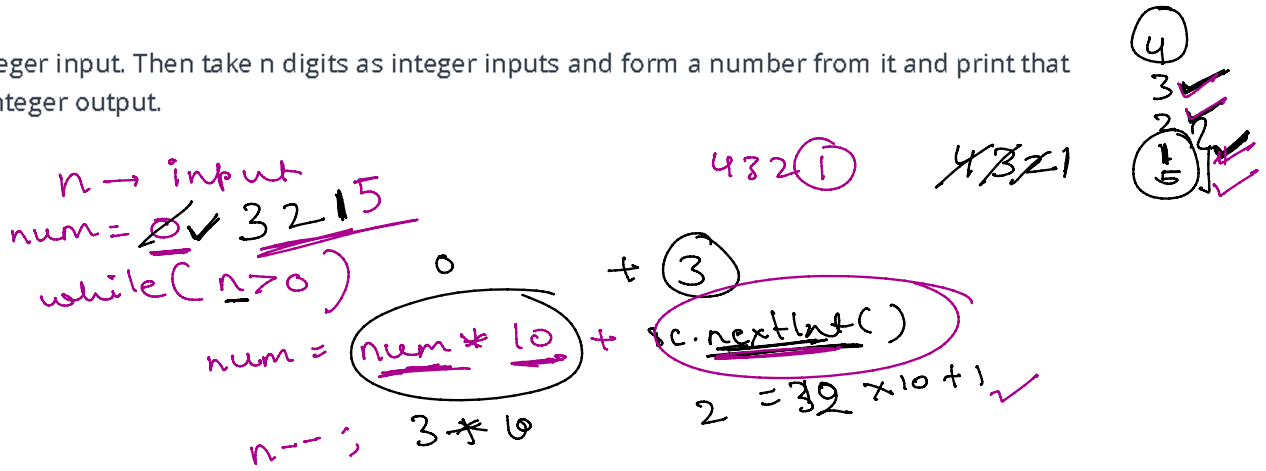
Java DSA Lecture 19

Friday, December 9, 2022 8:03 PM

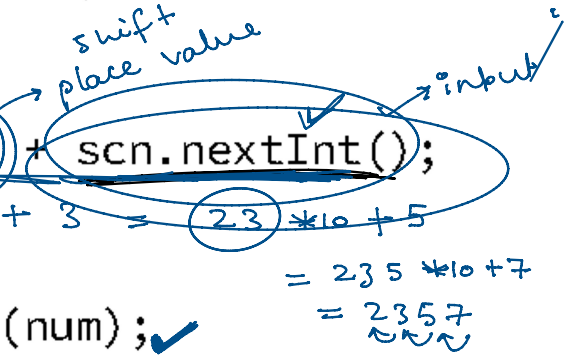
Print the final number xyzw...

Problem | Submissions | Leaderboard | Discussions

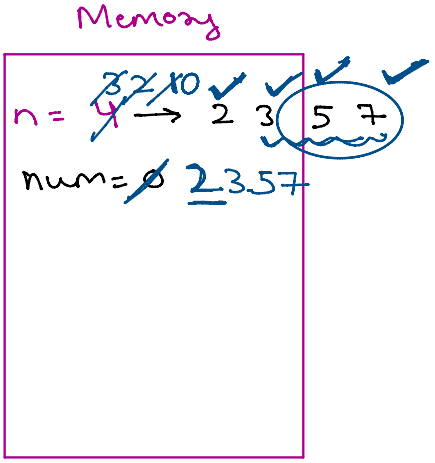
Take n as an integer input. Then take n digits as integer inputs and form a number from it and print that number as an integer output.



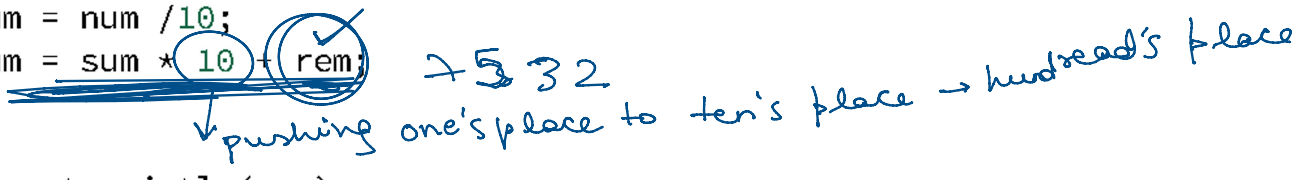

```
Scanner scn = new Scanner(System.in);
int n = scn.nextInt();
int num =0;
while(n > 0){
    num = num * 10 + scn.nextInt();
    n--;
}
System.out.println(num);
```



2357



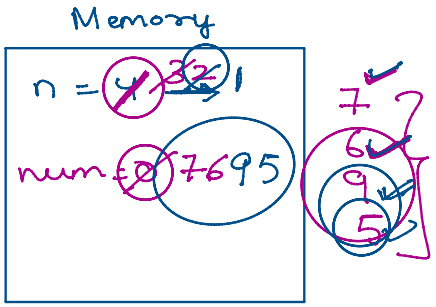

```
Scanner scn = new Scanner(System.in);
int num =scn.nextInt(); — 2357
int sum = 0;
while(num > 0){
    int rem = num % 10;
    num = num /10;
    sum = sum * 10 + rem;
}
System.out.println(sum);
```



```
System.out.println(sum);
```

Bhageswar

```
Scanner scn = new Scanner(System.in);
int n = scn.nextInt();
int num =0;
while(n > 0){
    num = num * 10 + scn.nextInt();
    n--;
}
System.out.println(num);
```



0 * 10 + 7 → *10 + 6 = 76 → *10 + 9

769 x10 + 5 = 7695

Reverse n-digit number

Problem | Submissions | Leaderboard | Discussions

Take a number n greater than or equal to zero as an integer input.

Then you will be given n digits as integer inputs and you have to form a number from it. Print the number formed.

Then you have to reverse the digits of this number. And then print the final reversed number in the next line.

Sample Input 0

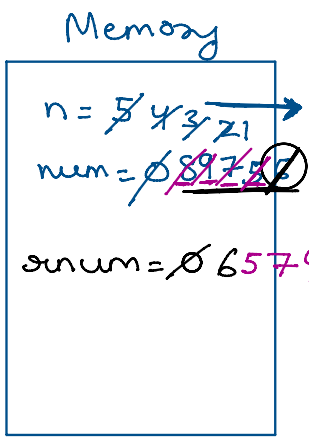
```
3
2
5
6
```

— scn.nextInt()

Sample Output 0

backup
256
652
} while

```
Scanner scn = new Scanner(System.in);
int n =scn.nextInt();
int num = 0;
while(n > 0){
    num = num * 10 + scn.nextInt();
    n--;
}
System.out.println(num); → 89756
int rnum = 0;
while(num > 0){ →
    int rem = num % 10;
    rnum = rnum * 10 + rem;
    num = num/10;
}
System.out.println(rnum); — 65798 → reverse
```



8 9 7 5 6

System.out.println(rnum); — 65 798 → reverse

Armstrong number

$\{ abcd = a^n + b^n + c^n + d^n \}$

n → count of digit

153 = $1^3 + 5^3 + 3^3$ ⇒ 1 + 125 + 27

n = 3

= 153

1634 = $1^4 + 6^4 + 3^4 + 4^4$

n = 4

= 1 + 1216 + 81 + 256

= 1634

489 = $4^3 + 8^3 + 9^3$

n = 3

64 512 729

n → input — 729

temp = n

int sum = 0 ~~729~~ 737

while (n > 0)

int rem = n % 10;

sum = sum + (rem * rem * rem) ← cube

n = n / 10

if (digit == 3) Math.pow(rem, 3)

737 + 7^3

= sum = 0 + 729

729 + 2^3

8

n=n/10 ✓ Math.pow(rem, x) count ✓ 729+ $\frac{2^3}{8}$

digit == y

sum = sum + (rem * rem * rem*rem)

```
Scanner scn = new Scanner(System.in);
int n = scn.nextInt();→ 153
int temp = n;
int sum = 0; ✓
while(n > 0){  153
    int rem = n % 10;
    sum = sum + (rem * rem * rem);
    n = n / 10; ✓ 1x 1x1
}

if(sum == temp){
    System.out.println(true);
}else{
    System.out.println(false);
}
```

Memory

n = ~~153~~ 15 ✓

temp = 153 ✓

Sum = ~~0~~ 27 + 125 = 152 + 1 = 153 ✓

rem = ~~3~~ 5 1

Print Armstrong in a range

Problem | Submissions | Leaderboard | Discussions

Take x and y as integer inputs.

Print all the Armstrong numbers in separate line which lie in the range x to y (both x and y inclusive)

Use the function isArmstrong() which checks if a number is an Armstrong number or not and returns true or false accordingly.

Sample Input 0

```
1
200
```

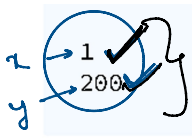
Boolean isArmstrong(int n)

temp = n

ans = 0 ✓

while (n > 0) n%10;

rem = 3;



Sample Output 0

temp =
ans = 0
while (n > 0)
rem = n % 10;
n = n/10;
int pow = (int)Math.pow(rem, 3);
ans += pow
if (ans == temp)
return true
else return false

$1^3 = 1$

Main ( )
{ x - input
y - input
for (i = x → y) {
isArmstrong (i) {
} print

```
static boolean isArmStrong(int n){
    int temp = n ;
    int ans = 0;
    while(temp > 0){
        int rem = temp % 10;
        temp = temp/ 10;
        int pow = (int)Math.pow(rem , 3);
        ans+=pow; // 153 = 1^3 + 5^3 + 3^3
    }

    if(ans == n){
        return true;
    }else{
        return false;
    }
}
public static void main(String[] args) {
    /* Enter your code here. Read input from
    Scanner scn = new Scanner(System.in);
    int x = scn.nextInt();
    int y = scn.nextInt();
    for(int i = x; i<=y ;i++){
        if(isArmStrong(i)){
            System.out.println(i);
        }
    }
}
```

Memory

x = 150

y = 1000

i = 150 151 152 153

temp = 153 → 15 → 1 → 0

n = 153

pow = 27 → 125 → 1

ans = 27 + 125 = 152 + 1 = 153

rem = 5 → 1

$0^3 = 0$

false

true → 153

Rotate 7-digit number to right by three

Problem | Submissions | Leaderboard | Discussions

Take n as an integer input, you have to pick the last 3 digits of the number of and put them in the starting.

eg. 1234567 is given, then this number should transform to 5671234.

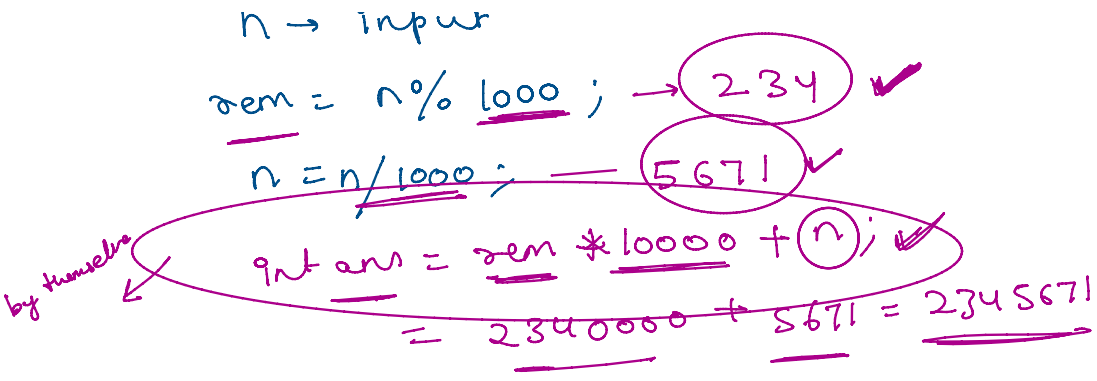
n rem

234 5671

234 0000 + 5671

n → input

rem = n % 1000 ; → 234




```
/* Enter your code here. Read input fr
Scanner scn = new Scanner(System.in);
int n =scn.nextInt();
int rem = n % 1000;
n = n / 1000;
int ans = rem * 10000 + n;
System.out.println(ans);
```

4589 6471

647

→ 4589

647 * 10000 + 4589

= 6470000 + 4589

= 6474589