Generating an Odd Random Number between a given Range

Asked 9 years, 6 months ago Modified 5 years ago Viewed 12k times



How to generate an odd Random number between a given range...

7 For Eg: For range between 1 to 6 .. Random No is 3 or 1 or 5



Method for Generating Random No:



```
Random_No = Min + (int)(Math.Random()*((Max-Min)+1))
```



Refer How do I generate random integers within a specific range in Java?

Method For Generating Odd Random No:

```
Random_No = Min + (int)(Math.Random()*((Max-Min)+1))
if(Random_No%2 ==0)
{
    if((Max%2)==0)&&Random_No==Max)
    {
        Random_No = Random_No - 1;
    }
    else{
        Random_No = Random_No +1;
    }
}
```

This Function will always convert 2 into 3 and not 1 Can we make this a more random function which can convert 2 sometimes into 3 and sometimes into 1??

```
iava math random
```

Share Improve this question Follow

```
edited May 23, 2017 at 12:25

Community Bot

1 1
```

asked Sep 26, 2012 at 5:18

Sanket
383 1 3 12

Sorted by:





¹ Another way would be to generate a number from 0 to 2 ((6−1)/2) and double then increment the result. It's easy to generalize this to a range starting with any number. – BlueRaja – Danny Pflughoeft Sep 26, 2012 at 5:46 ✓



Assuming max is inclusive, I'd suggest the following:



```
if (Max % 2 == 0) --Max;
if (Min % 2 == 0) ++Min;
Random_No = Min + 2*(int)(Math.random()*((Max-Min)/2+1));
```



It results in even distribution among all the odd numbers.



Share Improve this answer Follow



answered Sep 26, 2012 at 5:22

CrazyCasta

24.4k 4 41 67

2 Actually, <u>romedius</u> did the clever stuff; I just cleaned up the edit a bit, accidentally but unavoidably getting undue credit. You can see the details of who did what in the revision history (click on the time by the 'edited' label). – Jonathan Leffler Sep 26, 2012 at 5:51

@CrazyCasta, you do not need the first line where you decrement Max. - rouble Mar 31, 2017 at 5:12

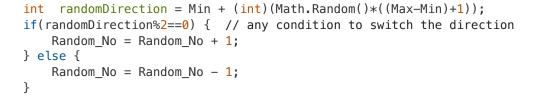
@CrazyCasta. Great solution. But why we need "if (Min % 2 == 0) ++Min". ? I think we do not need to change the value of Min? – user3369592 Apr 19, 2017 at 4:27



If you want to include randomness in the direction as well use random number for the same.

2





Share Improve this answer Follow

answered Sep 26, 2012 at 5:21



Bharat Sinha **13.3k** 6 37 63



1

Instead of generating a random number between 0 and 6, generate one between 0 and 5 and round up to the nearest odd number, that way you'll have a perfect distribution (33% for each possibility (1, 3, 5))



Share Improve this answer Follow



answered Sep 26, 2012 at 5:21

Jonathon Ashworth

1,174 10 19







```
Random_No = Min + (int)(Math.Random()*((Max-Min)+1))
repartitionNumber = (int) (Math.Random()*((2)) // between 0 and 1
if(Random_No%2 ==0)
      if(Random No+1<=Max && Random No-1>=Min)
          if(repartitionNumber==0)
              Random_No = Random_No + 1;
          else
              Random_No = Random_No - 1;
      else if(Random No+1<=Max)</pre>
          Random_No = Random_No + 1;
      else if (Random No-1>=Min)
          Random_No = Random_No - 1;
}
```

Share Improve this answer Follow

answered Sep 26, 2012 at 5:29





I wonder why other answers all use the int cast to generate the random number. Why not generate random integer directly, which is more accurate than real number way?

1



```
Random rn = new Random();
if (maximum % 2 == 1) maximum = maximum + 1; // turn right bound to even
if(minimum % 2 == 0) minimum = minimum - 1; // turn left bound to odd
int range = (maximum - minimum + 1) / 2;
int randomNum = rn.nextInt(range) * 2 + minimum;
```

Share Improve this answer Follow

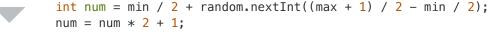
answered Sep 26, 2012 at 5:34





To generate an odd number from a integer you can use n * 2 + 1 Really you are generating random numbers and applying a transformation afterwards

1





This will work even if the range is [1,5] [2,5] [2,6] [1,6]

Share Improve this answer Follow

answered Sep 26, 2012 at 7:17



Peter Lawrey

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up





Mathematically the numbers will not gain anything by rounding up or down in the last step.

O Instead the first and the last number have a 50% lower chance to get picked over all the other numbers.



Stick with CrazyCasta's or J.A's solution.

1

Share Improve this answer Follow







How about checking the return from Math.random() as a floating number. If its int part is an even number, then convert up/down based on its floating part. Like:



assume Math.random() returned x.y; if x is even, return (y>=0.5)?(x+1):(x-1)



Will this randomized a little?

Share Improve this answer Follow

answered Sep 26, 2012 at 5:31



I guess thats what my question is... I want equal probability for all the numbers ranging from x to y – Sanket Sep 26, 2012 at 6:00

Let the rouding above or below depend on a random epsilon.

```
0
```

```
Random_No = Min + (int)(Math.Random()*((Max-Min)+1))
if(Random_No%2 ==0)
{
```



```
if((Max%2)==0)&&Random_No==Max)
{
    Random_No = Random_No - 1;
}
else{
    epsilon = Math.Random();
    if(epsilon > 0.5)
        Random_No = Random_No + 1;
    else
        Random_No = Random_No - 1;
}
```

Share Improve this answer Follow

}

answered May 13, 2013 at 19:37







In Java 1.7 or later, I would use <u>ThreadLocalRandom</u>:



```
import java.util.concurrent.ThreadLocalRandom;
```



```
// Get odd random number within range [min, max]
// Start with an odd minimum and add random even number from the remaining
range
public static int randOddInt(int min, int max) {
    if (min % 2 == 0) ++min;
    return min + 2*ThreadLocalRandom.current().nextInt((max-min)/2+1);
}

// Get even random number within range [min, max]
// Start with an even minimum and add random even number from the remaining
range
public static int randEvenInt(int min, int max) {
    if (min % 2 != 0) ++min;
    return min + 2*ThreadLocalRandom.current().nextInt((max-min)/2+1);
}
```

The reason to use ThreadLocalRandom is explained <u>here</u>. Also note that the reason we +1 to the input to ThreadLocalRandom.nextInt() is to make sure the max is included in the range.

Share Improve this answer Follow

edited May 23, 2017 at 12:17

Community Bot

1 1

answered Mar 31, 2017 at 4:59
rouble
13.7k 16 96 97

