TECHIE DELIGHT </>

Q

FAANG Interview Prep Practice HOT

Data Structures and Algorithms 🗡

Generate desired random numbers with equal probability

Write an algorithm to generate random numbers from 1 to 12 with equal probability, using a given function that generates random numbers from 1 to 6, with equal probability.

Practice this problem

Approach 1

The idea is to make two separate calls to the specified function and store the result in two variables, x and y, which would be random numbers between 1 and 6. Then we can quickly establish that:

- 1. The expression $x \times 2$ returns an even random number between 2 and 12 (i.e., 2, 4, 6, 8, 10, and 12) with equal probability.
- 2. The expression y & 1 returns either 0 or 1 depending upon whether y is even or odd.

The idea is to use the expression (x * 2) - (y & 1), which returns random numbers from 1 to 12 with equal probability. This expression works since

This website uses cookies. By using this site you agree to the use of cookies, our policies, copyright terms and other conditions. Read our Privacy Policy.

2. If y & 1 is 1, the expression returns the random odd numbers 1, 3, 5, 7, 9, and 11 with equal probability.

Following is the C, Java, and Python program that demonstrates it:

```
#include <stdio.h>
1
2
    #include <string.h>
    #include <stdlib.h>
3
    #include <time.h>
4
5
6
    // A function that returns random numbers from 1 to 6 with equal proba
7
    int getRandomNumber() {
8
         return (rand() % 6) + 1;
9
    }
10
11
    // Generate random numbers between 1 and 12 with equal probability usi
12
    // function that generates random numbers from 1 to 6 with equal proba
13
    int generate()
14
    {
         int x = getRandomNumber();
15
         int y = getRandomNumber();
16
17
18
         return 2*x - (y \& 1);
19
    }
20
    int main(void)
21
22
23
         // initialize srand with a distinctive value
24
         srand(time(NULL));
25
26
         int freq[13];
27
         memset(freq, 0, sizeof(freq));
28
29
         for (int i = 0; i < 1000000; i++)
30
         {
31
             int val = generate();
             freq[val]++;
32
33
         }
34
35
         for (int i = 1; i \le 12; i++) {
             printf("%2d \sim %0.2f%\n", i, freq[i]/10000.0);
36
37
38
39
         return 0;
40
    }
```

This website uses cookies. By using this site you agree to the use of cookies, our policies, copyright terms and other conditions. Read our Privacy Policy.

Java	•
Python	~
Output (will vary):	
1 ~ 8.33%	
2 ~ 8.35%	
3 ~ 8.35%	
4 ~ 8.31%	
5 ~ 8.32%	
6 ~ 8.33%	
7 ~ 8.29%	
8 ~ 8.38%	
9 ~ 8.35%	
10 ~ 8.34%	
11 ~ 8.35%	
12 ~ 8.31%	

Approach 2

Another way to generate the desired random numbers is to use the expression x + (y & 1) * 6 or x + !(y & 1) * 6, where x and y represent the output of two distinct calls made to the random() function.

How this works?

Let's consider the expression x + (y & 1) * 6:

- 1. |x| returns random numbers from 1 to 6 with equal probability.
- 2. y & 1 returns 0 or 1 depending upon whether y is even or odd.

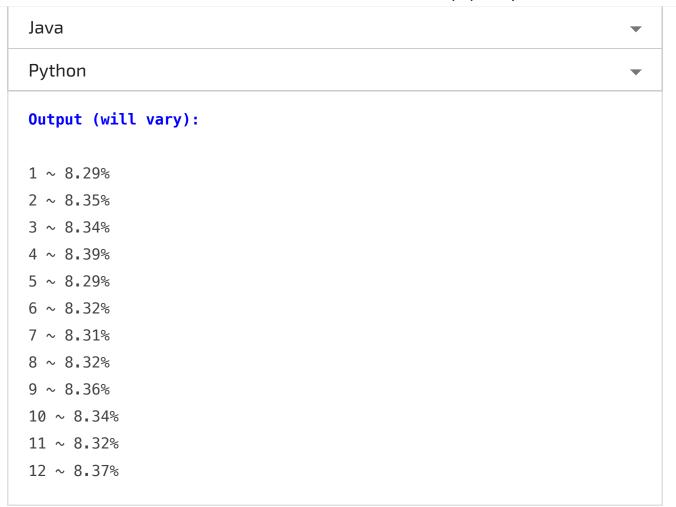
This website uses cookies. By using this site you agree to the use of cookies, our policies, copyright terms and other conditions. Read our Privacy Policy.

If y is even, the expression reduces to x, which gives random numbers from 1 to 6, and if y is odd, the expression is reduced to x + 6, which gives random numbers from 7 to 12.

Following is the C, Java, and Python program that demonstrates it:

```
1
    #include <stdio.h>
2
    #include <stdlib.h>
3
    #include <string.h>
    #include <time.h>
4
5
6
    // A function that returns random numbers from 1 to 6 with equal proba
7
    int getRandomNumber() {
         return (rand() % 6) + 1;
8
9
    }
10
11
    // Generate random numbers between 1 and 12 with equal probability usi
    // function that generates random numbers from 1 to 6 with equal proba
12
13
    int generate()
14
    {
15
         int x = getRandomNumber();
16
         int y = getRandomNumber();
17
18
         return x + (y & 1) * 6;
19
    }
20
21
    int main(void)
22
    {
         // initialize srand with a distinctive value
23
24
         srand(time(NULL));
25
26
         int freq[13];
27
         memset(freq, 0, sizeof(freq));
28
29
         for (int i = 0; i < 1000000; i++)
30
         {
31
             int val = generate();
             freq[val]++;
32
33
         }
34
35
         for (int i = 1; i \le 12; i++) {
36
             printf("%2d \sim %0.2f%\n", i, freq[i]/10000.0);
37
         }
38
39
         return 0;
40
    }
```

This website uses cookies. By using this site you agree to the use of cookies, our policies, copyright terms and other conditions. Read our Privacy Policy.



Author: Aditya Goel

Techie Delight © 2022 All Rights Reserved. | Privacy Policy | Send feedback

This website uses cookies. By using this site you agree to the use of cookies, our policies, copyright terms and other conditions. Read our Privacy Policy.