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 * 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

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:

C

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>

// A function that returns random numbers from 1 to 6 with equal proba
int getRandomNumber() {
    return (rand() % 6) + 1;
}

// Generate random numbers between 1 and 12 with equal probability usi
// function that generates random numbers from 1 to 6 with equal proba
int generate()
{
    int x = getRandomNumber();
    int y = getRandomNumber();

    return 2*x - (y & 1);
}

int main(void)
{
    // initialize srand with a distinctive value
    srand(time(NULL));

    int freq[13];
    memset(freq, 0, sizeof(freq));

    for (int i = 0; i < 1000000; i++)
    {
        int val = generate();
        freq[val]++;
    }

    for (int i = 1; i <= 12; i++) {
        printf("%2d ~ %0.2f%%\n", i, freq[i]/10000.0);
    }

    return 0;
}
```

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.

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:

C

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

// A function that returns random numbers from 1 to 6 with equal proba
int getRandomNumber() {
    return (rand() % 6) + 1;
}

// Generate random numbers between 1 and 12 with equal probability usi
// function that generates random numbers from 1 to 6 with equal proba
int generate()
{
    int x = getRandomNumber();
    int y = getRandomNumber();

    return x + (y & 1) * 6;
}

int main(void)
{
    // initialize srand with a distinctive value
    srand(time(NULL));

    int freq[13];
    memset(freq, 0, sizeof(freq));

    for (int i = 0; i < 1000000; i++)
    {
        int val = generate();
        freq[val]++;
    }

    for (int i = 1; i <= 12; i++) {
        printf("%2d ~ %0.2f%%\n", i, freq[i]/10000.0);
    }

    return 0;
}
```

Java

Python

Output (will vary):

```
1 ~ 8.29%
2 ~ 8.35%
3 ~ 8.34%
4 ~ 8.39%
5 ~ 8.29%
6 ~ 8.32%
7 ~ 8.31%
8 ~ 8.32%
9 ~ 8.36%
10 ~ 8.34%
11 ~ 8.32%
12 ~ 8.37%
```

Author: Aditya Goel