Valid Perfect Square

Given a positive integer *num*, write a function which returns True if *num* is a perfect square else False.

Note: Do not use any built-in library function such as `sqrt`.

Example 1:

```
Input: 16
Returns: True
```

Example 2:

```
Input: 14
Returns: False
```

Credits:
Special thanks to @elmirap for adding this problem and creating all test cases.

Solution 1

```
public boolean isPerfectSquare(int num) {
    int i = 1;
    while (num > 0) {
        num -= i;
        i += 2;
    }
    return num == 0;
}
```

written by fhqplzj original link here

Solution 2

Just slightly modified my sqrt solutions. You can find some explanation there.

(Note I renamed the parameter to x because that's the name in the sqrt problem and I like it better.)

Java, C++, C, C#

```
long r = x;
while (r*r > x)
    r = (r + x/r) / 2;
return r*r == x;
```

Python

```
r = x
while r*r > x:
    r = (r + x/r) / 2
return r*r == x
```

Ruby

```
r  = x
r = (r + x/r) / 2 while r*r > x
r*r == x
```

JavaScript

```
r = x;
while (r*r > x)
    r = ((r + x/r) / 2) | 0;
return r*r == x;
```

written by StefanPochmann original link here

Solution 3

```
class Solution {
public:
    bool isPerfectSquare(int num) {
        if (num < 0) return false;
        int root = floorSqrt(num);
        return root * root == num;
    }

    int32_t floorSqrt(int32_t x) {
        double y=x; int64_t i=0x5fe6eb50c7b537a9;
        y = *(double*)&(i = i-(*(int64_t*)&y)/2);
        y = y * (3 - x * y * y) * 0.5;
        y = y * (3 - x * y * y) * 0.5;
        i = x * y + 1; return i - (i * i > x);
    }
};
```

written by dploop original link here