How to approximate $\sqrt{x}$ using bit shifts

Consider the binary representation of x:

$$x = \sum_{i=0}^{n} 2^i x_i ,$$

where $x_n = 1$. Then $2^n \le x < 2^{n+1}$ and $2^{n/2} \le \sqrt{x} < 2^{(n+1)/2}$.
Let $m = \lfloor \frac{n}{2} \rfloor$ and consider the number

$$y = \left\lfloor \frac{x}{2^m} \right\rfloor .$$

Then

$$\begin{aligned} y < 2^{n+1-m} &= 2^{n+1-\lfloor n/2 \rfloor} \le 2^{n+1-(n-1)/2} \\ &= 2^{(n+1)/2} \\ &= \sqrt{2} \cdot 2^{n/2} \\ &\le \sqrt{2} \cdot \sqrt{x} \end{aligned}$$

Conversely, $2^m \le 2^{n/2}$, so $\frac{x}{2^m} \ge \frac{x}{2^{n/2}} \ge \frac{x}{\sqrt{x}} = \sqrt{x}$.

Thus, $\left\lfloor \frac{x}{2^m} \right\rfloor > \sqrt{x} - 1$, that is, it is no smaller than the largest integer no greater than $\sqrt{x}$. Since any non-prime x must be divisible by an integer $\le \sqrt{x}$, it suffices to test divisibility of x by integers between 2 and y.

Now, $\left\lfloor \frac{x}{2^m} \right\rfloor = x >> m$ (right-shift by m positions)

One way to compute it is:

```
int y=x;
while (x > 1) {
    x >>=2;
    y >>=1;
}
```