k repetitions.

At most k bit operations for the addition of up and low, and at most k bit operations for the division, as by point (b), the maximum bit operations is kl, where in this case, 2 is a single bit number, and thus $l = 1$. Magma's inbuilt floor function is polynomial time, and adds no bit operations.

At most $3k^2$ bit operations for the cubing of s, and at most $3k$ bit operations for the subtraction $s^3 - n$. At most $3k^2$ bit operations for the cubing of $s + 1$, and at most $3k$ bit operations for the subtraction $(s + 1)^3 - n$.

At most $2k$ bit operations for the final comparisons.

From the bit operations calculated at each stage, we have the following function, $f_c(k)$, to model the computational complexity of the algorithm.

$$\begin{aligned} f_c(k) &= k^2 + k^2 + (k + k + 3k^2 + 3k + 2k) \cdot k + k + k + 3k^2 + 3k + 3k^2 + 3k + 2k \\ &= (3k^2 + 7k) \cdot k + 8k^2 + 10k \\ &= 3k^3 + 7k^2 + 8k^2 + 10k \\ &= 3k^3 + 15k^2 + 10k \end{aligned}$$

$$\therefore f_c(k) = 3k^3 + 15k^2 + 10k$$

In order to prove the algorithm is indeed polynomial time, we must show that $f_c(k)$ is $O(k^\alpha)$, for some $\alpha \in \mathbb{Z}$. We first need to define O notation. A function, $f(k)$ is $O(k)$ if

$$f(k) \leq C \cdot g(k) \quad \forall\, k \geq N$$

where $C, N \in \mathbb{Z}^+$. Thus, we need to find such an N, C, and $g(k)$. We thus look for an N, through the following inequalities.

$$\begin{aligned}
3k^3 &\geq 15k^2 \\
3k &\geq 15 \\
\therefore k &\geq 5 \dots\dots\dots (A) \\
3k^3 &\geq 10k \\
3k^2 &\geq 8 \\
k^2 &\geq \frac{8}{3} \\
\therefore k &\geq \frac{2\sqrt{2}}{\sqrt{3}} \\
\therefore k &\geq 2 \dots\dots\dots (B)
\end{aligned}$$

From these two results, (A) and (B), it is clear that $k \geq 5$, and thus we have the existence of an N, where $N = 5$ in this case. Thus we have the following results.

$$\begin{aligned}
f_c(k) &= 3k^3 + 15k^2 + 10k \\
\therefore f_c(k) &\leq 3k^3 + 3k^3 + 3k^3 \quad \forall\, k \geq 5 \\
&= 9k^3 \\
\therefore f_c(k) &\leq 9k^3
\end{aligned}$$

As a result, we have the existence of a C, where $C = 9$, and a $g(k)$, where $g(k) = k^3$. As a result, we have that $f_c(k)$ is $O(k^3)$. Thus the algorithm is polynomial time. Upon inputting large values of n, the algorithm behaves like a polynomial time algorithm is expected to, thus backing up the proof. In the following section, the magma code for the algorithm is provided.

Magma Code

```
cube:=procedure(n)
    log:=1;
    k:=0;
    while log lt n do
        log:=log*2;
        k:=k + 1;
    end while;
    up:=n;
    low:=0;
    for i:=1 to k do
        m:=(up+low) div 2;
        val:=m^3 - n;
        if val gt 0 then
            up:=m;
        end if;
        if val lt 0 then
            low:=m;
        end if;
    end for;
    s:=Floor((up+low) div 2);
    under:=AbsoluteValue(s^3 - n);
    over:=AbsoluteValue((s+1)^3 - n);
    if over - under gt 0 then
        print s;
    end if;
    if over - under lt 0 then
        print(s+1);
    end if;
end procedure;
```