Solutions of Homework 6: *Programming Project*

- The following is the complete *C* program that implements the given function with the required modifications.

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

int count;

double third(double n){
  count++;
  if (n<=3)
    return (1);
  else
    return (3+third(pow(n,1.0/3.0)));
}

int main(){
int i;
 printf("n\t\tCount\t\tReturned Value\n");
 printf("=====\t\t=====\t\t==============\n");

 for (i=1; i<5; i++){
   count = 0;
   printf("3^%3.0f\t\t%d\t\t%5.2f\n",pow(3.0,i),
              count,third(pow(3.0,pow(3.0,i))));
 }
}
```

- The output of the above program is:

```
 n               Count            Returned Value
 =====           =====            ==============
 3^  3           2                 4.00
 3^  9           3                 7.00
 3^ 27           4                10.00
 3^ 81           5                13.00
```

- To analytically find the *Count*, we establish the recurrence relation that describes the number of times, $T(n)$, the function *third()* gets called and then solve it as follows:

$$
\begin{aligned}
T(n) &= 1 + T(\sqrt[3]{n}) = 1 + T(n^{\frac{1}{3}}), \quad T(2) = 1 \\
&= 1 + \left(1 + T((n^{\frac{1}{3}})^{\frac{1}{3}})\right) = 2 + T(n^{\frac{1}{3^2}}) \\
&= 2 + \left(1 + T((n^{\frac{1}{3^2}})^{\frac{1}{3}})\right) = 3 + T(n^{\frac{1}{3^3}}) \\
&= 3 + \left(1 + T((n^{\frac{1}{3^3}})^{\frac{1}{3}})\right) = 4 + T(n^{\frac{1}{3^4}}) \\
&\vdots \\
&= k + T(n^{\frac{1}{3^k}})
\end{aligned}
$$

The process stops when we reach $T(3)$ at which case

$$
\begin{aligned}
& n^{\frac{1}{3^k}} = 3 \\
\Rightarrow \quad & \frac{1}{3^k} \log_3 n = 1 \\
\Rightarrow \quad & 3^k = \log_3 n \\
\Rightarrow \quad & k = \log_3 \log_3 n \\
\Rightarrow \quad & T(n) = k + T(3) = \log_3 \log_3 n + 1.
\end{aligned}
$$

Note that the above function gives the same values for *Count* obtained from the program at different values of n. For example when $n = 3^{27}$, $T(n) = \log_3 \log_3 3^{27} + 1 = \log_3 27 + 1 = 3 + 1 = 4$.

- To analytically find the *Returned Value*, we establish the recurrence relation that describes the final value, $F(n)$, returned by the function *third()* and then solve it as follows:

$$\begin{aligned}
F(n) &= 3 + F(\sqrt[3]{n}) = 3 + F(n^{\frac{1}{3}}), \quad F(3) = 1 \\
&= 3 + \left(3 + F((n^{\frac{1}{3}})^{\frac{1}{3}})\right) = 2 \cdot 3 + F(n^{\frac{1}{3^2}}) \\
&= 2 \cdot 3 + \left(3 + F((n^{\frac{1}{3^2}})^{\frac{1}{3}})\right) = 3 \cdot 3 + F(n^{\frac{1}{3^3}}) \\
&= 3 \cdot 3 + \left(3 + F((n^{\frac{1}{3^3}})^{\frac{1}{3}})\right) = 4 \cdot 3 + F(n^{\frac{1}{3^4}}) \\
&\vdots \\
&= 3k + F(n^{\frac{1}{3^k}})
\end{aligned}$$

As shown before, the process stops when we reach $F(3)$ at which case

$$\begin{aligned}
& n^{\frac{1}{3^k}} = 3 \\
\Rightarrow\ & \frac{1}{3^k} \log_3 n = 1 \\
\Rightarrow\ & 3^k = \log_3 n \\
\Rightarrow\ & k = \log_3 \log_3 n \\
\Rightarrow\ & F(n) = 3k + F(3) = 3 \log_3 \log_3 n + 1.
\end{aligned}$$

Note that the above function gives the same values for *Returned Value* obtained from the program at different values of n. For example when $n = 3^{81}$, $F(n) = 3 \log_3 \log_3 3^{81} + 1 = 3 \log_3 81 + 1 = 3 \cdot 4 + 1 = 13$.