**Siena College’s 35th Annual High School Programming Contest**

| **Sponsored by** |  |
| --- | --- |

##### March 31, 2023

###### Green Problem #4: Perfection!

Background Information:

A positive integer N is a ***perfect*** number if the sum of its proper factors (i.e. the factors that are strictly less than N) is equal to N. For example, since 6 = 1 + 2 + 3 it’s a perfect number.

If the sum of the proper factors of N is greater than N, N is an ***abundant*** number. If the sum is less than N, N is a ***deficient*** number. For example, the proper factors of 12 are 1, 2, 3, 4, and 6 and the sum of these factors is 16 which is greater than 12 so 12 is an abundant number. The positive integer 32 is an example of a deficient number. The proper factors of 32 are 1, 2, 4, 8, and 16 and the sum of these factors is 31. The number 1 has no proper factors so the sum of its proper factors is 0 and hence 1 is a deficient number.

A relatively easy theorem to prove in number theory is that all prime numbers are deficient.

Write a program that identifies input values as perfect, abundant, or deficient. For both Java and Python the mod operator is %. So if N and K are integers and N = 25 and K = 7 then N % K is 4. This operator will probably be useful.

###### Programming Problem:

Input:  A positive integer less than 1 billion.

Output: The number followed by the adjective Perfect, Abundant, or Deficient

###### Example 1: Input: 28

Output: 28 Perfect

###### Example 2: Input: 31

Output: 31 Deficient

###### Example 3: Input: 1

Output: 1 Deficient

###### Example 4: Input: 24

Output: 24 Abundant