**Perfect Numbers: -**

Easy Accuracy: 17.21% Submissions: 127K+ Points: 2

Given a number **N,** check if a number is perfect or not. A number is said to be perfect if sum of all its factors excluding the number itself is equal to the number. Return 1 if the number is Perfect otherwise return 0.

**Example 1:**

**Input:**

**N =** 6

**Output:**

1

**Explanation:**

Factors of 6 are 1, 2, 3 and 6.

Excluding 6 their sum is 6 which

is equal to N itself. So, it's a

Perfect Number.

**Example 2:**

**Input:**

**N =** 10

**Output:**

0

**Explanation:**

Factors of 10 are 1, 2, 5 and 10.

Excluding 10 their sum is 8 which

is not equal to N itself. So, it's

not a Perfect Number.

**Your Task:**  
You don't need to read input or print anything. Your task is to complete the function **isPerfectNumber()** which takes an Integer N as input and returns 1 if N is a Perfect number else returns 0.

**Expected Time Complexity:** O(sqrt(N))  
**Expected Auxiliary Space:** O(1)

**Constraints:**  
1 <= N <= 1012

**Code: -**

//{ Driver Code Starts

#include <bits/stdc++.h>

using namespace std;

// } Driver Code Ends

class Solution {

public:

int isPerfectNumber(long long N) {

long long sum = 1;

for(int i=2; i<=sqrt(N); ++i){

if(N % i == 0){

sum += i;

sum += (N / i);

}

}

if(sum == N and N != 1) return 1;

else return 0;

}

};

//{ Driver Code Starts.

int main() {

int t;

cin >> t;

while (t--) {

long long N;

cin>>N;

Solution ob;

cout << ob.isPerfectNumber(N) << endl;

}

return 0;

}

// } Driver Code Ends

**T.C: - O(√n)**

**S.C: - O(1)**