1288A - Deadline

Idea: **adedalic**

Tutorial

1288A - Deadline

At first, let's note that if x is integer and x and y are non-negative then $x + \lceil y \rceil = \lceil x + y \rceil$. So, instead of looking at $x + \lceil \frac{d}{x+1} \rceil$ we can consider $\lceil x + \frac{d}{x+1} \rceil$.

It's easier since the function $x + \frac{d}{x+1} = (x+1) + \frac{d}{(x+1)} - 1$ is more common function and it can be proven that it's concave upward. It means that this function has a unique minimum and, moreover, we can calculate it: $f(x) = x + \frac{d}{x+1}$ has minimum value in $x_0 = \sqrt{d} - 1$ and $f(x_0) = 2\sqrt{d} - 1$.

Since the ceiling function is monotonically increasing so we can assume that $\lceil f(x) \rceil \le \lceil f(x+1) \rceil$ for all $x \ge \sqrt{d}$.

So we can just iterate x from 0 to $\lfloor \sqrt{d} \rfloor$ and check the unequation $\lceil f(x) \rceil \le n$. The total complexity is equal to $O(T\sqrt{d})$.

There is a simple optimization: because of the monotone ceiling we can prove that we need to check only $\lfloor \sqrt{d} - 1 \rfloor$ and $\lceil \sqrt{d} - 1 \rceil$.

Solution (adedalic)

```
#include<bits/stdc++.h>

using namespace std;

int main() {
#ifdef _DEBUG
	freopen("input.txt", "r", stdin);
//	int tt = clock();
#endif
```

```cpp
	int T; cin >> T;
	while(T--) {
		int n, d;
		cin >> n >> d;

		int x, MAG = (int)sqrt(d) + 10;
		for(x = 0; x < MAG; x++) {
			if(x + (d + x) / (x + 1) <= n)
				break;
		}
		cout << (x < MAG ? "YES" : "NO") << endl;
	}

	return 0;
}
```