1. An integer $x$ is said to be a *prime power* if and only if it can be written as $p^k$ where $p$ is a prime number and $k \geq 1$ is a positive integer. Define a function `primepower` which takes an integer $n$ and checks whether $n$ is a prime power.

(You may define auxiliary functions.) For instance,

```plaintext
- primepower;
val it = fn : int -> bool
- primepower(0);
val it = false : bool
- primepower(1);
val it = false : bool
- primepower(823543);
val it = true : bool
- primepower(214358881);
val it = true : bool
- primepower(32);
val it = false : bool
```

2. Define a function `consec_ones` which takes an integer and checks whether the binary representation of its absolute value has three consecutive ones.

(You may define auxiliary functions.) For instance,

```plaintext
- consec_ones;
val it = fn : int -> bool
- consec_ones(5);
val it = false : bool
- consec_ones(7);
val it = true : bool
- consec_ones(20);
val it = false : bool
- consec_ones(23);
val it = true : bool
- consec_ones(28);
val it = true : bool
- consec_ones(40);
val it = false : bool
- consec_ones(224);
val it = true : bool
```