5) (10 pts) ALG (Base Conversion)

Write a function that takes a string *str* and an integer *b* (where $2 \leq b \leq 10$), and returns 1 if *str* represents an integer in base *b* that is a perfect power of *b*. For example:

```
isPower("323", 4);      // Return 0. 323₄ = 59₁₀, which is not a power of 4
isPower("27", 3);       // Return 0. 27 is not a valid base 3 integer.
isPower("plum", 8);     // Return 0. plum is not a valid base 8 integer.
isPower("1000", 10);    // Return 1. 1000₁₀ is a power of 10 (10³)
isPower("000001", 2);   // Return 1. 1₂ = 1₁₀, which is a power of 2 (2⁰)
```

Notes: You may assume *b* is always within the range specified above. Your function must return 0 if *str* is NULL or the empty string. Strings may be padded on the left with any number of zeros.

```
// You must use this function signature. You may write helper functions as
// needed:
int isPower(char *str, int b);

int charToInt(char c) { return c - '0'; }

int isPower(char *str, int b)
{
   int i, length, c, flag = 0;

   if (str == NULL || str[0] == '\0')
      return 0;

   for (i = 0; i < strlen(str); i++)
   {
      // Convert this character to an integer.
      c = charToInt(str[i]);

      // If this is not even a valid digit in base b, return 0.
      if (c > b - 1 || c < 0)
         return 0;

      // Key insight: For this to be a perfect power of b, we must
      // have '1' followed by '0''s only.
      if (c == 1)
      {
         if (flag > 0)  // If we have seen more than one '1', it's over.
            return 0;
         flag = 1;
      }
      // Can't have anything but 0's and 1's.
      else if (c != 0)
      {
         return 0;
      }
   }
   return flag;
}
```

Grading: 4 pts - for taking care of invalid cases, 1 pt - returns 1 or 0 for all cases, 2 pts - rejects strings that have anything but 0 or 1, 2 pts - rejects any string with > 1 non-zero char, 1 pt - accepts correct strings