Short answer: $y = (\frac{3126}{3125})^{3126}$, where the last 10 digits are 2137165824, and the first 10 digits are 2.718716741 which differs from e from the 5th digit on. The corresponding $x$ value where $x < y$, is $x = (\frac{3126}{3125})^{3125}$, which has the first 10 digits 2.717847030 and the last 10 digits 6359186432.

Long answer:

$x^y = y^x$

Let $y = kx$, where $k \in \mathbb{Q}$.

$x^{kx} = (kx)^x$

$(x^k)^x = (kx)^x$

$x^k = kx$

divide each side by k

$x^{k-1} = k$

raise each side by $\frac{1}{k-1}$

$x = k^{\frac{1}{k-1}}$.

Since k is rational, so is the exponent $\frac{1}{k-1}$, so we can express that exponent in lowest terms (i.e. $p, q \in \mathbb{N}$). Let that exponent, $\frac{1}{k-1} = \frac{p}{q}$. Which makes $k = \frac{p+q}{p}$.

$x = (\frac{p+q}{p})^{\frac{p}{q}}$

But since x is rational, that means that both the numerator and denominator must stay integers.

Both$(p+q)^{\frac{1}{q}}$ and $p^{\frac{1}{q}}$ must be integers. That means that $p+q$and $p$ are powers of q: $p + q = m^q$,and $p = n^q$.

But if $p = n^q$then the next power of q is $(n + 1)^q$, which by the binomial theorem is$n^q + qn^{q-1} + ... + 1$.

Which means that the powers of q are separated by at least$qn^{q-1}$, but we are looking to find two that are separated by only $q$, namely $p + q$ and $p$.

And$qn^{q-1} > q$ when $q > 1$, so we're never going to find a$p$ and $q$that meets our criteria of being powers of $q$ when $q > 1$. So therefore $q = 1$.

This simplifies our equation for x:

$x = (1 + \frac{1}{p})^p$

What does it mean for x to be a non-repeating decimal. It means that x is

$$x = \frac{n}{10^k} \tag{1}$$

where $\exists n, k \in \mathbb{Z}$.

If x is expressed as an irreducible fraction

$x = \frac{a}{b}$

a can be prime factor as $a = 2^{a_2} 3^{a_3} 5^{a_5} \ldots$.

b can be prime factor as $b = 2^{b_2} 3^{b_3} 5^{b_5} \ldots$,

where if $b_i > 0$, then $a_i = 0$.

then,

$\frac{a}{b} 10^k = n \in \mathbb{Z}$

$a10^k = bn$

$2^{a_2} 3^{a_3} 5^{a_5} \ldots 10^k = bn$

$2^{a_2} 3^{a_3} 5^{a_5} \ldots 2^k 5^k = bn$

$2^{a_2+k}3^{a_3}5^{a_5+k}\ldots = bn$
$2^{a_2+k}3^{a_3}5^{a_5+k}\ldots = 2^{b_2}3^{b_3}5^{b_5}\ldots 2^{n_2}3^{n_3}5^{n_5}$
$2^{a_2+k}3^{a_3}5^{a_5+k}\ldots = 2^{b_2+n_2}3^{b_3+n_3}5^{b_5+n_5}\ldots$
$2^{a_2+k-b_2-n_2}3^{a_3-b_3-n_3}5^{a_5+k-b_5-n_5}\ldots = 1.$

Which means that
$a_2 + k = b_2 + n_2$
$a_3 = b_3 + n_3$
$a_5 + k = b_5 + n_5$
$a_7 = b_7 + n_7$
...

Which implies that $b_3, b_7, ...$ must be 0, since $a$and $b$are irreducible i.e. if $b_i > 0$,then $a_i = 0$.

So $b = 2^{b_2}5^{b_5}$.
$x = (1 + \frac{1}{p})^p$
$x = (\frac{p+1}{p})^p$
$x = \frac{(p+1)^p}{p^p}$

Since $p$ and $p+1$ are consecutive, they contain no common factors and $x$ is irreducible.

$p = 2^{b_2}5^{b_5}$.

Let's treat these three cases seperately $b_2 < b_5$ , $b_2 = b_5$and $b_2 > b_5$.

If$b_2 < b_5$,multiply the top and bottom by $2^{b_5-b_2}$to let the 2s catch up with the 5s.

$x = (\frac{2^{b_5}5^{b_5}+2^{b_5-b_2}}{2^{b_5}5^{b_5}})^p = (\frac{10^{b_5}+2^{b_5-b_2}}{10^{b_5}})^p$, and notice that $2^{(b_5-b_2)} \mod 10^m \neq 0$, so that term will be responsible for the trailing digits.

If $b_2 = b_5$, x will end with ...0001 since $p = 10^b$.

For $b_2 > b_5$, we multiply the top and bottom of x by $5^{b_2-b_5}$.

$x = (\frac{2^{b_2}5^{b_5}+1}{2^{b_2}5^{b_5}})^p = (\frac{2^{b_2}5^{b_2}+5^{b_2-b_5}}{2^{b_2}5^{b_2}})^p = (\frac{10^{b_2}+5^{b_5-b_2}}{10^{b_2}})^p$, likewise notice that $5^{b_5-b_2} \mod 10^m \neq 0$.

Since we are only interested in the first 8 digits and the last 8 digits of x, we can use the binomial theorem on$x = (1 + \frac{1}{p})^p$.

Binomial theorem is

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}$$

Expanding out the numerator separately
$x = (\frac{10^{b_5}+2^{b_5-b_2}}{10^{b_5}})^p$
where $a = 10^{b_5}$, $b = 2^{b_5-b_2}$, $n = p = 2^{b_2}5^{b_5}$,
and the denominator is simply $10^{b_5 p}$ and the numerator is nonzero $\mod 10^m$. The last 8 digits of x will therefore be
$(10^{b_5} + 2^{b_5-b_2})^p \mod 10^8$.

$$(10^{b_5} + 2^{b_5-b_2})^p = \sum_{k=0}^{p} \binom{p}{k} 10^{b_5 k} 2^{(b_5-b_2)(p-k)}$$

Simpler isn't it? Let's look at this equation in $\mod 10^8$, all the terms where $b_5 k \geq 8$ will be 0. So the $k = 0$ term is always necessary, but even the $k = 1$term can be neglegted when $b_5 \geq 8$.

The $k = 0$term is
$2^{(b_5-b_2)p}$ when $b_5 > b_2$and $5^{(b_2-b_5)p}$ when $b_2 > b_5$.

k=1 term

$$\binom{p}{1} 10^{b_5 1} 2^{(b_5-b_2)(p-1)} = p10^{b_5} 2^{(b_5-b_2)(p-1)} = 2^{b_2} 5^{b_5} 10^{b_5} 2^{(b_5-b_2)(p-1)}$$

$$= 2^{b_2} 5^{b_2} 5^{b_5-b_2} 10^{b_5} 2^{(b_5-b_2)(p-1)} = 10^{b_2} 5^{b_5-b_2} 10^{b_5} 2^{(b_5-b_2)(p-1)} = 10^{b_2+b_5} 5^{b_5-b_2} 2^{(b_5-b_2)(p-1)}$$

$= 10^{b_2+b_5} 5^{b_5-b_2} 2^{(b_5-b_2)} * 2^{(b_5-b_2-1)(p-1)} = 10^{b_2+b_5} 10^{b_5-b_2} 2^{(b_5-b_2-1)(p-1)} = 10^{(b_2+b_5)(b_5-b_2)} 2^{(b_5-b_2-1)(p-1)}$

This is zero $\mod 10^8$when $(b_2 + b_5)(b_5 - b_2) \geq 8$. Actually $(b_2, b_5)$where this is nonzero is only $\{(0, 1), (0, 2), (1, 2), (2, 3), (3, 4)\}$ which correspond to $p = \{5, 25, 50, 500, 5000\}$. After rulling out these values of $p$by calculation x directly (without the binomial theorem), we can just go ahead and use the $k = 0$term only.

So we only need to calculate $2^{(p-2)(b_5-b_2)} 10^{2b_5} \mod 10^8$ since we are multiplying all terms by$10^{pb_5}$.

To calculate the starting digits of x, we use the k=...,p-2, p-1, p terms. Recall the denominator is $10^{b_5 p}$

k=p-1

$$\binom{p}{p-1} 10^{b_5(p-1)} (2^{b_5-b_2})^1 = p10^{b_5 p - b_5} (2^{b_5-b_2})$$

Dividing by the denominator gives $p10^{-b_5}(2^{b_5-b_2}) = 2^{b_2} 5^{b_2} 5^{b_5-b_2} 10^{-b_5}(2^{b_5-b_2}) = 5^{b_5-b_2} 10^{b_2-b_5} 2^{b_5-b_2} = 1$.

k=p

$$\binom{2^{b_2} 5^{b_5}}{2^{b_2} 5^{b_5}} (2^{-b_2} 5^{-b_5})^{2^{b_2} 5^{b_5} - 2^{b_2} 5^{b_5}} = 1$$

Clearly we need at least the $k = p - 2$ term, and dividing by $10^{b_5 p}$
$\binom{p}{p-2} 10^{-2b_5} (2^{b_5-b_2})^2 = \frac{p(p-1)}{2} 10^{-2b_5} (2^{b_5-b_2})^2$

The $k - i$term is
$\frac{p(p-1)(p-2)\ldots(p-i+1)}{i!} 10^{-ib_5} (2^{b_5-b_2})^i$

We are interested when this becomes $< 10^{-8}$.

The $k - i$term is smaller than
$\frac{p^i}{i!} 10^{-ib_5} (2^{b_5-b_2})^i = \frac{1}{i!} 2^{ib_2} 5^{ib_2} 5^{ib_5-ib_2} 10^{-ib_5} 2^{ib_5-ib_2} = \frac{1}{i!} 10^{ib_2} 10^{-ib_5} 10^{ib_5-ib_2} = \frac{1}{i!} 10^0 = \frac{1}{i!}$ Wow, that's helpful!

So when does $\frac{1}{i!} < 10^{-10}$?

$i! > 10^{10} \Rightarrow i > 13$. Only the last 13 terms are needed, possibly less.

It turns out that these terms roughly correspond to the Taylor exansion of $e = 1 + \frac{1}{1!} + \frac{1}{2!} + ...$

$2^n \mod 10^m$ cycles over $n$ with a period of$4 * 5^{m-1}$.

So we have $n = (b_5 - b_2)2^{b_2}5^{b_5} (\mod 4 * 5^7)$

When $2^{b_2} \geq 4$and $5^{b_5} \geq 5^7$we get $2^{b_2-2}5^{b_5-7} * 4 * 5^7 \mod 4 * 5^7 = 0$. That means that when $b_2 \geq 2$ and $b_5 \geq 7$ (we already assumed $b_5 \geq 8$), we'll always see the same value for the last 8 digits of x. Solutions, if they exist, must be $b_2 = 0$ or 1. If $b_2 = 0$, powers of 5 do not meet the criteria of last 8 digits distinct and no 0s or 9s, which are checked with the Scala code below.

```
 val nbonus5 = (for { b5 <- 0 to scala.math.pow(2,6).toInt}
 yield{
 (BigInt(5).modPow(b5,BigInt(10).pow(8)),b5)})
.filter{x=>x._1.toString().reverse.dropWhile(_=='0')
.reverse.takeRight(8).filter{x => x != '9' && x != '0'}
.distinct.length() == 8 }
// Yields the empty set, even without the no '9' or '0' criteria,
// there's only 1 solution: 5^48 = 37890625.

 val nbonus2 = (for {b <- 0 to 4*scala.math.pow(5,7).toInt}
yield{
(BigInt(2).modPow(b,BigInt(10).pow(8)),b)})
.filter{x=>x._1.toString().reverse.dropWhile(_=='0')
.reverse.takeRight(8).filter{x => x != '9' && x != '0'}
.distinct.length() == 8 }
//> nbonus2  : scala.collection.immutable.IndexedSeq[(scala.math.BigInt, Int)
//| = Vector((78345216,1324), (15284736,1836), (75186432,3125), ....
```

If $b_2 = 0$ or 1, then $n = (b_5 - b_2)2^{b_2}5^{b_5} (\mod 4 * 5^7)$ will cycle over $b_5$with period 4 due to the $(b_5 - b_2) \mod 4$, and $5^{b_5} \mod 5^7$ not playing a role when $b_5 \geq 7$. Therefore there are only (10+9) cases to consider. Namely $(b_2, b_5)$= (0, $b_5$ from 1 to 10, where $b_5$cycles with period 4), and (1, $b_5$ from 1 to 7 where $b_5$cycles with period 2.)

```
(0,0,2),
(0,1,248832),
(0,2,75482624),
(0,3,57293568),
(0,4,89509376),
(0,5,59186432),*
(0,6,72890624),
(0,7,20813568),
(0,8,87109376),
(0,9,79186432),* so b2=0, b5=1 mod 4, is an infinite family of solutions
(0,10,12890624),
->
(0,11,20813568),
```

```
(0,12,87109376),
(0,13,79186432), *
(0,14,12890624),

(1,0,225),
(1,1,37424601),
(1,2,49402624),
(1,3,63589376),
(1,4,59290624),
(1,5,11109376),
(1,6,12890624),
(1,7,87109376),
->
(1,8,12890624),
(1,9,87109376),
```

As the only entries marked * have 8 distinct digits, there are no solutions for x for the bonus question with no 9s or 0s.

If $b_2 > b_5$then $x = (\frac{2^{b_5}5^{b_5}+2^{b_5-b_2}}{2^{b_5}5^{b_5}})^p = (\frac{10^{b_5}+5^{b_2-b_5}}{10^{b_5}})^p$. The trailing digits are powers of 5. The cycle length of the powers of $5^n \mod 10^m$ is only $2^{m-2}$, which is only 64 in our case. Only one solution ends up matching the non bonus question $(b_2, b_5, x) =$ (4,1,37890625).

How about y?

$y = (1 + \frac{1}{p})^{p+1}$. Exact same rules apply: p is made up of 2s and 5s.

$y = (\frac{p+1}{p})^{p+1}$.

A very small modification to the program yields results to the bonus question!

```
// For y
(Stream.from(0) map (b5 =>
 for {
  b2 <- (0 to 10)
 }
 yield  {
  val pBig = BigInt(2).pow(b2) * BigInt(5).pow(b5)
  if (b5>b2)
  (b2,b5, (BigInt(10).pow(b5) + BigInt(2).pow(b5-b2))
.modPow(pBig+1,BigInt(10).pow(8)),
(BigInt(10).pow(b5) + BigInt(2).pow(b5-b2))
.modPow(pBig+1,BigInt(10).pow(10)))
  else
(b2,b5, (BigInt(10).pow(b2) + BigInt(5).pow(b2-b5))
.modPow(pBig+1,BigInt(10).pow(8)),
(BigInt(10).pow(b2) + BigInt(5).pow(b2-b5))
.modPow(pBig+1,BigInt(10).pow(10)))
  }))
```

```
.take(100).toList.flatten
.filter{x=>x._3.toString()
.reverse.dropWhile(_ == '0').reverse.takeRight(8)
.filter{x=>x != '9' && x != '0'}.distinct.length() == 8 }
.foreach(println)
//> (0,5,37165824,2137165824)
//| (1,52,86314752,186314752)
//| (1,532,81476352,8181476352)
//| (1,978,46153728,3646153728)
//| (0,1324,78345216,7278345216)
//| (1,1325,78345216,7278345216)
//| (2,1326,78345216,7278345216)
//| (3,1327,78345216,7278345216)
//| (4,1328,78345216,7278345216)
//| (5,1329,78345216,7278345216)
//| (6,1330,78345216,7278345216)
//| (7,1331,78345216,7278345216)
//| (8,1332,78345216,7278345216)
//| (9,1333,78345216,7278345216)
//| (10,1334,78345216,7278345216)
//| (0,1836,15284736,5015284736)
```

y is a solution when $(b_2, b_5)$ =(0,5),(1,52),(1,532)...

```
 val nbonus2 = (for {
b <- 0 to 4*scala.math.pow(5,7).toInt }
 yield{
 (BigInt(2).modPow(b, BigInt(10).pow(8)),b)    })
.filter {x=>x._1.toString()
.reverse.dropWhile(_ == '0').reverse.takeRight(8)
.filter { x => x != '9' && x != '0'}.distinct.length() == 8 }

scala.collection.immutable.IndexedSeq[(scala.math.BigInt, Int)]
Vector((78345216,1324), (15284736,1836), (75186432,3125),
(28417536,11716), (76451328,11747), (32765184,16434),...
```

We'll still get $n = (b_5 - b_2)(2^{b_2}5^{b_5} + 1)(\mod 4 * 5^7) = (b_5 - b_2)2^{b_2}5^{b_5} + (b_5 - b_2)(\mod 4 * 5^7)$. The first term is 0 when $b_2 \geq 2$and $b_5 \geq 7$.

We run a bit of code to create the set$N = \{1324, 1836, 3125, 11716, 11747, 16434, ...\}$, which has 167 elements. Solutions to the bonus question are therefore solutions to:

$$\boxed{(b_5 - b_2) \mod 4 * 5^7 \in N}$$

For example one family of solutions is of the form:

$b_5 = 1326 + s * 4 * 5^7$, and $b_2 = 2 + t * 4 * 5^7$

where $s$, and $t$ are arbitrary non-negative integers, and $b_5 > b_2$ $(s \geq t)$. Recalling that this is the equation for

$y = (1 + \frac{1}{p})^{p+1}$, where $p = 2^{b_2}5^{b_5}$.

(0,5,37165824) and (1,52,86314752) are solutions, varified in Wolfram Alpha, the rest aren't varified.

Now to get the first 10 digits

```
def binomialCoefficient(n: Int, k: Int) ={
 if (k > n/2.0)
 {
  val k2=n-k
  (BigInt(n - k2 + 1) to n).product / (BigInt(1) to k2).product
 }
 else
  (BigInt(n - k + 1) to n).product / (BigInt(1) to k).product
}

val nbonus = for {
 b2 <- 0 to 0
 b5 <- 5 to 5 }
yield{
(for {
  k <- 0 to 14    }
yield{
 // (p+1 choose k) * 10^-kb5 * 2^kb5
val p = (scala.math.pow(2,b2) * scala.math.pow(5,b5)).toInt
val a = binomialCoefficient(p+1, k)
val b = scala.BigDecimal(10).pow(-k*b5) * scala.BigDecimal(2).pow(b5*k)
   BigDecimal(a) * b  })}
//> nbonus  : scala.collection.immutable.IndexedSeq[scala.collection.immutabl
//| .IndexedSeq[scala.math.BigDecimal]] = Vector(Vector(1, 1.00032, 0.5001600
//| 00, 0.166666649600000, 0.041639995736064000000, 0.0083200042680314757120000
//| , 0.00138489244376139923718144000, 0.000197525230835911571200278528000000
//|  0.0000246432477990883276229467491532800, 0.000002732005213779818863230
//| 9826130739, 2.7250112804325425269402806425758 45E-7, 2.470148407222633458
//| 2447849386569E-8, 2.051869943599600859612433346890443E-9,1.57280566261394
//| 635737430583193029930E-10, 1.11911863490679200240142035538263 35349E-11))
nbonus.flatten.sum
//> res16: scala.math.BigDecimal = 2.718716741954659466357518590853400295826
```