Practice Quiz 2 Free Response (60 pts)

Time Limit: **15 mins**

Name: _____ AndrewID: _____

Time limit: 15 mins

Free Response 1: largestLengthKNumber (30 pts)

Write the function `largestLengthKNumber(n, k)` that takes a positive integer `n` and returns the largest number of length `k` that occurs in `n`.

For example `largestLengthKNumber(1075734, 3)` should return 757. The length 3 numbers in 1075734 are: 107, 757, 573, and 734. Of these, 757 is the largest.

You may assume `n` has at least `k` digits, and that k > 0.

Note: You may only use material that was covered in units 1-2. Thus, do not use strings, lists, tuples, sets, dictionaries, OOP, or recursion.

Test cases:

```
def testLargestLengthKNumber():
    assert(largestLengthKNumber(1075734, 3) == 757)
    assert(largestLengthKNumber(27583, 2) == 83)
    assert(largestLengthKNumber(2951, 2) == 95)
    assert(largestLengthKNumber(12345, 1) == 5)
    assert(largestLengthKNumber(9871, 3) == 987)
    assert(largestLengthKNumber(57510, 5) == 57510)
```

Write your answer below.

Free Response 2: nthPairishPrime (30 pts)

Background: A number n is pairish (a coined term) if it has at least 3 digits, and every pair of digits sum to equivalent numbers. For example 924783 is pairish because (9 + 2) == (4 + 7) == (8 + 3) == 11.

If there is an odd number of digits, the leftmost digit is alone in a pair. For example, 73452 is pairish because (7) == (3 + 4) == (5 + 2) == 7.

A pairishPrime is a number that is both pairish, and prime.

With that in mind, write the function `nthPairishPrime(n)` that takes an integer n and returns the nth pairishPrime. Counting starts at 0, so `nthPairishPrime(0) == 101`.

Note: You may only use material that was covered in units 1-2. Thus, do not use strings, lists, tuples, sets, dictionaries, OOP, or recursion.

Test cases:

```
def testIsPairishPrime():
    assert(isPairishPrime(101) == True)
    assert(isPairishPrime(211) == True)
    assert(isPairishPrime(431) == True)
    assert(isPairishPrime(523) == True)
    assert(isPairishPrime(1753) == True)

    assert(isPairishPrime(2) == False)
    assert(isPairishPrime(97) == False)
    assert(isPairishPrime(11) == False)
    assert(isPairishPrime(100) == False)
    assert(isPairishPrime(1001) == False) # Not prime: 1001 = 7 * 11 * 13
    assert(isPairishPrime(3113) == False) # Not prime: 3113 = 11 * 283
    assert(isPairishPrime(113) == False)
    assert(isPairishPrime(347) == False)
    assert(isPairishPrime(10211) == False)
```

Write your answer below.