Check Digits

From about 1970 International Standard Book Numbers have been used by book sellers and libraries to insure that most commercially available books have a unique identifying number. Since these numbers were often entered by hand, it was useful to have an error checking system built–in. The most common human error in transcribing numbers is the transposition of digits. The ISBN-10 checks for this, as well as incorrect individual digits. In 2007 ISBN-13 was introduced, which we will discuss later. Each ISBN-10 consists of 10 digits, $d_1, \ldots, d_{10}$, the final being the check digit. This digit is chosen so that

$$\sum_{k=1}^{10}(k)(d_k) = \sum_{k=1}^{10}(11-k)(d_k) = 0 \mod 11. \tag{0.1}$$

The prime 11 is used in place of 10 for this calculation, so in some cases the value of the check digit is 10; then the symbol X is used. Today such checks are carried out by computer, but there are relatively simple ways to carry out the check by hand. By using the properties of arithmetic modulo 11, it is not hard to show that if $d_1, \ldots, d_{10}$ is a valid ISBN, then for any $j < k$ with $d_j \neq d_k$, if the j, k digits are transposed, then the result is no longer a valid ISBN. This is also the case if any single digit is changed.

Here is a perl subroutine which computes the ISBN check from a string of 10 decimal digits. The subroutine expects input in the form of a string with 8 decimal digits.

```
sub isbn {
    $_[0] =~ m/(\d+)/;
    my  @a=split //, $_[0];
    $n=scalar(@a) ;
    if ($n != 10 ) { return -1; } # Invalid input!
    my $i=1;
    my $acc=$a[0];
    while ( $i<$n ) {
        $acc+= $a[$i]*(++$i); # array indices 0..9
        }
    return $acc % 11;
    }
```

A certain collection of 8-digit student numbers seems to use something like the above check, if two zeros are placed before the number to yield something like 0011106069, then the ISBN check yields 0 in many cases, but sometimes 10 when the final digit of the input number is 0 or 1. It seems that, for $d_{10} = 0$, zero is used in place of X. However, for $d_{10} = 1$ case, the ISBN algorithm would yield 0 if d_{10} were set to 0. The precise relation to the ISBN algorithm is unclear.

Other methods of choosing a check digit are now more common. The Luhn Algorithm is used for credit card and imei numbers. The digits are numbered from right to left, the rightmost (i.e., first counting from the right) is the check digit. One computes the sum of the odd digits. For each even digit one multiplies by 2. If the result is greater than 9, the two resulting digits are added. This can also be expressed as doubling the digit and subtracting 9 if the result is greater than 9. Note that this maps the numbers $0 \ldots 4$ to the even digits in order, while it maps $5 \ldots 9$ to the odd digits in order. Each even position digit is transformed in this way and the sum of the transformed values is added

to the sum of the odd position digits. The Luhn digit is remainder of this total when divided by 10. The check digit is chosen to make the Luhn digit 0. Given $d_n, \ldots, d_1$, one computes the Luhn total $L(d_n, \ldots, d_1)$. If this is 0, then

the Luhn check is ok. If not, by replacing d_1 by $d_1 - L(d_n, \ldots, d_1) \bmod 10$, a number with $L = 0$ is generated.

Here is a perl subroutine for computing the Luhn check digit. The input is the string of decimal digits. The output is the input digit string followed by the rightmost digit of the string followed by the value the rightmost digit should be for the Luhn check to be 0.

```
sub luhn {
    $_[0] =~ m/(\d+)/;
    my $k=$1;
    return $_[0],'?',-1 unless $k;
    my  @a= split //, $k;
    my $n = $#a;
    my $i=0; my $j=1; my $acc=0;
    my $ii=$a[$n];
    while ($n>0) {
        $n--; $i=$a[$n];
        if ($j & 1 ) { $i*=2; $i-=9 if $i>9; }
        $acc+=$i; $acc %=10; $j++;
        }
    $acc=10-$acc if $acc;
    return $k,$ii, $acc;
    }
```

In ISBN-10, arithmetic modulo 11 is used. The Luhn Algorithm uses arithmetic modulo 10. Universal Product Code (UPC) and ISBN-13 also use modulo 10. Instead of the Luhn mapping for alternate digits, they use multiplication by 1 and multiplication by 3 of the sums of alternate digits. For UPC the final digit is the check digit. One computes the sum of the even position digits (check digit excluded) plus 3 times the sum of the odd digits (check digit excluded). The check digit is the negative of this total modulo 10. Note that in checking for consistency of the sequence of digits, the final digit is taken with the even position digits, in that is not multiplied by 3. ISBN-13 is similar to UPC except the even position digits are multiplied by 3: if a zero is placed in front of a thirteen digit ISBN-13, the calculation coincides with that of UPC.

Other similar modulo 10 check algorithms use 7 as a multiplier. 3, 7, 1 and 9 are important because they are the only invertible elements of the multiplicative group modulo 10. Note that the transformation in the Luhn algorithm is also a one-to-one mapping of the 10 digits. A more subtle and powerful check digit was devised by the Dutch mathematician Jacobus Verhoeff, but it does not appear to be as widely used as the above mentioned.

The isbn check seems to work ok for student numbers with first two digits 11 or less. However, numbers beginning with 12 do not correspond to isbn 10. For a collection of 41 student numbers beginning with 12, two families of linear mod b check algorithms were tried. For linear mod 11 algorithms, all permutations of the factors $1, \ldots, 10$ were tested. All failed for at least half of the set of student numbers. For mod 10 linear algorithms, all variants of multiplying by 1, 3, 7, 10 and Luhn transformation were tried. None were satisfied for more than half of the sample set.