Book Code

a small Nifty Assignment

John Motil
Book Number

a short, simple, surprizing "Nifty" Assignment

John M. Motil, Professor Emeritus
California State University, Northridge

for

CS0: with ints, strings, etc
CS1: with methods, objects, etc
CS2: with stacks, queues, trees, etc
Book Number

is ONE problem which can be solved in MANY ways, with:

- multiply and divide operations
- divide and mod operations
- loops & nests of loops
- strings in many ways
- arrays in more ways
- methods & objects
- recursion
- stacks
- queues
- trees
- more!

A Universal Problem
Book Number:

**ISBN = International Standard Book Number**

Examples:

0205080057 or 020508005–7

123456789X or 1–234–56789–X

The last character is a check symbol related to the other 9 for error detection
### Many

<table>
<thead>
<tr>
<th>Algorithms:</th>
<th>Data Structures</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>1.</td>
</tr>
<tr>
<td>2.</td>
<td>2.</td>
</tr>
<tr>
<td>3.</td>
<td>3.</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>M</td>
<td>N</td>
</tr>
</tbody>
</table>
**Algorithm** for the check symbol

**Sum** the first digit and  
2 times the second digit, plus  
3 times the third digit,  
to  
9 times the ninth digit.

**Divide** this sum by 11, and the remainder is the checkSum.

**Fix** the check symbol to an ‘X’  
if the checkSum is 10.
Data Flow Diagram of the bookNumber Algorithm

For example: 020508005-7

\[
\text{sum} = 2*2 + 4*5 + 6*8 + 9*5 \\
= 4 + 20 + 48 + 45 \\
= 117
\]

\[
\text{check} = 117 \mod 11 \\
= 7
\]
**General Algorithm**

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Get digits $d_1, d_2, \ldots, d_9$</td>
</tr>
<tr>
<td>2</td>
<td>$\text{sum} = d_1 + 2d_2 + 3d_3 + 4d_4 + 5d_5 + 6d_6 + 7d_7 + 8d_8 + 9d_9$</td>
</tr>
<tr>
<td>3</td>
<td>$\text{check} = (\text{sum mod 11})$</td>
</tr>
</tbody>
</table>
| 4    | If ($\text{check} == 10$)  
   Output "X"  
Else // usually  
   Output check  
EndIf |
Many algorithms for the check symbol

$$\text{sum} = 1\times d_1 + 2\times d_2 + 3\times d_3 + 4\times d_4 + 5\times d_5 + 6\times d_6 + 7\times d_7 + 8\times d_8 + 9\times d_9$$

$$\text{check} = \left( \text{sum} \mod 11 \right)$$

Another: 10 down to 2

$$\text{total} = 10\times d_1 + 9\times d_2 + 8\times d_3 + 7\times d_4 + 6\times d_5 + 5\times d_6 + 4\times d_7 + 3\times d_8 + 2\times d_9$$

$$\text{check} = \left( 11 - \left( \text{total} \mod 11 \right) \right) \mod 11$$
Representation of BookNumber data

1. Many digits
   \[ \text{d[]} = \ \begin{array}{ccccccc} 0 & 2 & 0 & 5 & 0 & 8 & 0 & 0 & 5 \end{array} \]

2. Integer
   \[ \begin{array}{c} 0 2 0 5 0 8 0 0 5 \end{array} \]

3. Real number?
   \[ \begin{array}{c} 0 \ 2 0 5 0 8 0 0 5 \end{array} \]

4. String of Characters
   \[ \begin{array}{ccccccccc} 0 & - & 2 & 0 & 5 & - & 0 & 8 & 0 & 0 & 5 \end{array} \]

5. Array of digits or characters
   \[ \text{d[ ]} = \ \begin{array}{ccccccc} 0 & 2 & 0 & 5 & 0 & 8 & 0 & 0 & 5 \end{array} \]
More representations

Stack

Tree

Queue
One Integer form of Book Number

First Solution: Series of divs & mods

```c
int d1, d2, d3, d4, d5, d6, d7, d8, d9;

int bn = 123456789; // book number

d1 = bn / 100000000;
bn = bn % 100000000;

bn = bn / 10000000;

bn = bn % 10000000;

bn = bn / 1000000;

bn = bn % 1000000;

... 

bn = bn / 10;

bn = bn % 10;
```

(First is worst!)
Another Solution

series of divide and multiply

```c
int bn = 020508005;

int d1, d2, d3, d4, d5, d6, d7, d8, d9;

d9 = bn - 10*(bn/10);
d8 = bn/10 - 10*(bn/100);
d7 = bn/100 - 10*(bn/1000);
d6 = bn/1000 - 10*(bn/10000);
d5 = bn/10000 - 10*(bn/100000);
d4 = bn/100000 - 10*(bn/1000000);
d3 = bn/1000000 - 10*(bn/10000000);
d2 = bn/10000000 - 10*(bn/100000000);
d1 = bn/100000000 - 10*(bn/1000000000);
```
Yet Another Series Solution:

div & mod again!

// Book Number as one single integer

int d1,d2,d3,d4,d5,d6,d7,d8,d9;

int bn = 123456789; // book number

\[
\begin{align*}
\text{d9} &= \text{bn} \% 10; \\
\text{bn} &= \text{bn} / 10;
\end{align*}
\]

\[
\begin{align*}
\text{d8} &= \text{bn} \% 10; \\
\text{bn} &= \text{bn} / 10;
\end{align*}
\]

\[
\begin{align*}
\text{d7} &= \text{bn} \% 10; \\
\text{bn} &= \text{bn} / 10;
\end{align*}
\]

etc .. To d1
Loop form of BookCode

\[ bn = 123456789 \]  // book number

\[ \text{sum} = 0 \]

\[ \text{Loop } i = 9 \text{ to } 1 \]

\[
\begin{align*}
\text{di} & = bn \mod 10 \\
bn & = bn / 10 \\
\text{Inc sum} & = i \times \text{di}
\end{align*}
\]

EndLoop
Nested Loop of BookCode as single int

bn = 123456789; // book number
sum = 0;

Loop i = 9 to 1

\[
\text{di} = \text{bn} \mod 10;
\text{bn} = \text{bn} / 10;
\]

// Inc sum by i*di;

Loop j = 1 to i
Inc sum = di;
EndLoop

EndLoop
What about leading zeros:

```java
int bookNumber = 020508005;
```

(books in English begin with 0)

In Java a leading zero is interpreted as octal!

Nasty
Array representation of BookNumber

d = \begin{bmatrix}
0 & 2 & 0 & 5 & 0 & 8 & 0 & 0 & 5 \\
1 & 2 & 3 & \ldots & i & \ldots & 9
\end{bmatrix}

digit[] = \{0, 2, 0, 5, 0, 8, 0, 0, 5\}
sum = 0
Loop i = 1 to 9
Inc sum by i * digit[i]
EndLoop i
Product of two Arrays

sum = 0

Loop i = 1 to 9

Inc sum = weight[i] * digit[i]

EndLoop i

d =
0 2 0 5 0 8 0 0 5

w =
1 2 3 4 5 6 7 8 9
**BookNumber as an Array: Code another way**

\[ d = \begin{array}{ccccccc}
0 & 2 & 0 & 5 & 0 & 8 & 0 & 0 & 5 \\
1 & 2 & 3 & \ldots & i & \ldots & 9 
\end{array} \]

**Pass 1: Increment each position**

Loop \( i = 9 \) to 1

Inc \( \text{digit}[i-1] \) by \( \text{digit}[i] \)

EndLoop \( i \)

\[ d = \begin{array}{ccccccc}
20 & 20 & 18 & 18 & 13 & 13 & 5 & 5 & 5 \\
1 & 2 & 3 & \ldots & i & \ldots & 9 
\end{array} \]

**Pass 2: Accumulate all values; 117**

Avoids all multiplication
String representation of BookNumber

No hyphens

![Number representation without hyphens]

Normal hyphenation

![Number representation with normal hyphenation]

Many hyphens

![Number representation with many hyphens]
Tree of BookNumber

Sum of internal nodes = 117
Stack of BookNumber

Input

Inc

Sum

no multiplication
Queues with BookNumber

Input

0 2 0 5 0 8 0 0 5

Inc

22 22 18 18 13 13 5 5 5

Sum

117
Queue with BookNumber
**BookNumber as an Object:**

```
BookNumber

0  countryCode

1  3  4  7  8
  publisher  number

X  checkSymbol

true  isValid
```
Class Diagram of BookNumber:

BookNumber (d1,d2,...,d9)
BookNumber (integer)
BookNumber (string)
BookNumber (array)

Attributes: data, state

917930 publisher
65 pubNumber
false isValid
0-91.. ISBN

CONSTRUCT

setPublisher(s)
getCountry()

TRANSFORM

setPubNumber(i)

void methods
boolean methods
typed methods

COMPARE

isValid() ?
isEnglish ?

OBSERVE

checkSymbol
### Algorithms:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1.</strong> Up vs Down (10 to 2)</td>
<td>1. Many 10 digits</td>
</tr>
<tr>
<td><strong>2.</strong> Series (2) of div &amp; mult</td>
<td>2. One integer</td>
</tr>
<tr>
<td><strong>3.</strong> Loop &amp; nest of div &amp; mod</td>
<td>3. Two Arrays</td>
</tr>
<tr>
<td><strong>4.</strong> Two-Pass Incr &amp; Sum</td>
<td>4. A string</td>
</tr>
<tr>
<td><strong>5.</strong> Recursion</td>
<td>5. Stacks</td>
</tr>
<tr>
<td><strong>6.</strong> Weird: Primitive</td>
<td>6. Queue</td>
</tr>
<tr>
<td><strong>7.</strong> Weirder</td>
<td>7. List</td>
</tr>
<tr>
<td><strong>8.</strong> Weirdest</td>
<td>8. Tree</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td><strong>M. More</strong></td>
<td><strong>N. More</strong></td>
</tr>
</tbody>
</table>

### Data Structures

An assignment for all seasons (after all concepts)
**Weirdest solution:**

Search

**Compute all possible valid book numbers (many!)**

Store them (somehow, somewhere)

Search them to check for any given isbn.
Generalizing, Growing

1. from 10 digits to 13
   by the year 2007

2. Nifty Numbers
   “self consistent” integers
   of any length (not fixed)
Many

MANY

many

man

Many

many

Man

many
That’s all

More at

www.csun.edu/~jmotil/NiftyBookNumber
That is all!

Thank You
**Primitive Solution: Weird & Wonderful involving no multiplication:**

\[
\text{sum} = d_1 + 2d_2 + 3d_3 + 4d_4 + 5d_5 + 6d_6 + 7d_7 + 8d_8 + 9d_9
\]

\[
\text{sum} = d_1 + \\
d_2 + d_2 + \\
d_3 + d_3 + d_3 + \\
d_4 + d_4 + d_4 + d_4 + \\
d_5 + d_5 + d_5 + d_5 + d_5 + \\
\ldots \\
d_9 + d_9 + d_9 + d_9 + \ldots + d_9
\]
**Primitive Solution:** Weird & Wonderful involving no multiplication:

\[ \text{sum} = d_1 + 2d_2 + 3d_3 + 4d_4 + 5d_5 + 6d_6 + 7d_7 + 8d_8 + 9d_9 \]

\[ \text{sum} = d_1 + \]
\[ d_2 + d_2 + \]
\[ d_3 + d_3 + d_3 + \]
\[ d_4 + d_4 + d_4 + d_4 + \]
\[ d_5 + d_5 + d_5 + d_5 + d_5 + \]
\[ \ldots \ldots \]
\[ d_9 + d_9 + d_9 + d_9 + \ldots \ldots + d_9 \]

and no division:

```plaintext
while (sum >= 11)
    Dec sum by 11
EndWhile
```
A solution involving no division:

\[
\text{while (sum} > 11) \\
\quad \text{Dec sum by 11} \\
\text{EndWhile}
\]

And yet another solution with no division

\[
\text{If (sum} >= 11) \text{ sum = sum - 11} \\
\text{If (sum} >= 11) \text{ sum = sum = 11} \\
\ldots \\
\text{If (sum} >= 11) \text{ sum = sum - 11}
\]