Title: The 3-digit Digit Reversal Problem
(A Problem on Algorithms and Operators)

Activity Type: Exercise, Homework (Written, Pseudo-coding/Flowcharting, Algorithm/Logic)

Pre-requisites: Before undergoing this activity, students must have knowledge in the following topics:

1.) Working knowledge of procedural and event-driven programming in the C# language
2.) Working knowledge on how to use variables and related topics in the C# language
3.) Working knowledge on how to use the basic operators in the C# language as follows:
    a. Mathematical operator (i.e. Addition, Subtraction, Multiplication, Division, Modulo)
    b. String operators (i.e. Concatenation, Substring, Use of character indices)

Objective(s):

1.) Provide a comprehensive understanding of the problem at hand.
2.) Formulate algorithm/s to reverse the digit orders of a 3-digit number.
3.) Make use of tools such as pseudo-coding/flowcharting to break the algorithm down into steps that the student will find easy to follow.
4.) Facilitate to improve the student's critical thinking skills by immersing the student with the algorithm problem at hand.
5.) Reinforce critical thinking through the presentation of alternative algorithm/s to the student whenever possible.
6.) Create a simple program that demonstrates the algorithm at hand. (Optional)
7.) Reinforce the understanding between the difference of programming codes and algorithms by implementing the same algorithm in multiple programming platforms. (Optional)

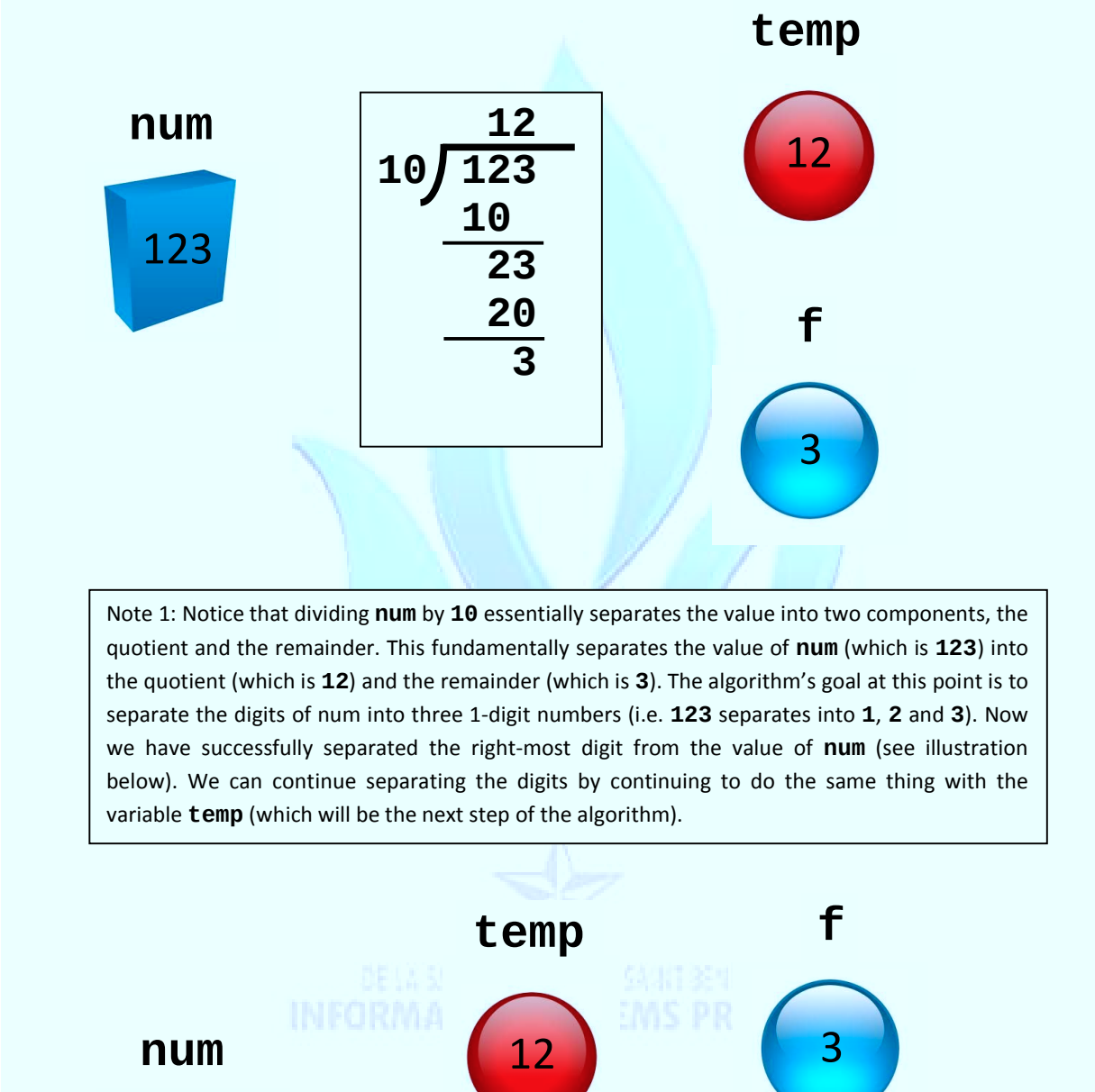
The Problem:

Given an arbitrary 3-digit integer number, provide an algorithm that reverses the order of its digits. Sample input values and their corresponding output values are shown below.

Input Values	Output Values
951	**159**
357	**753**
456	**654**
852	**258**

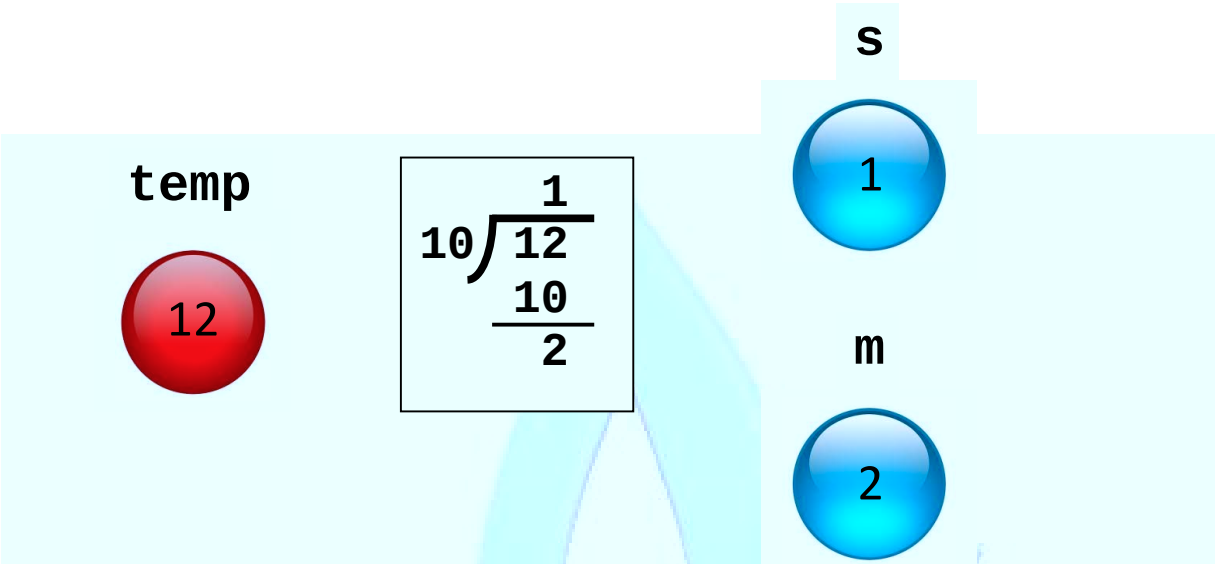
Algorithm 1: *To clearly illustrate algorithm 1, let's assume that a 3-digit integer is stored in a variable* **num** *which is arbitrarily set to the value* **123.**

1.) Divide **num** by **10**, store the quotient in a variable **temp** and store the remainder in another variable **f**.

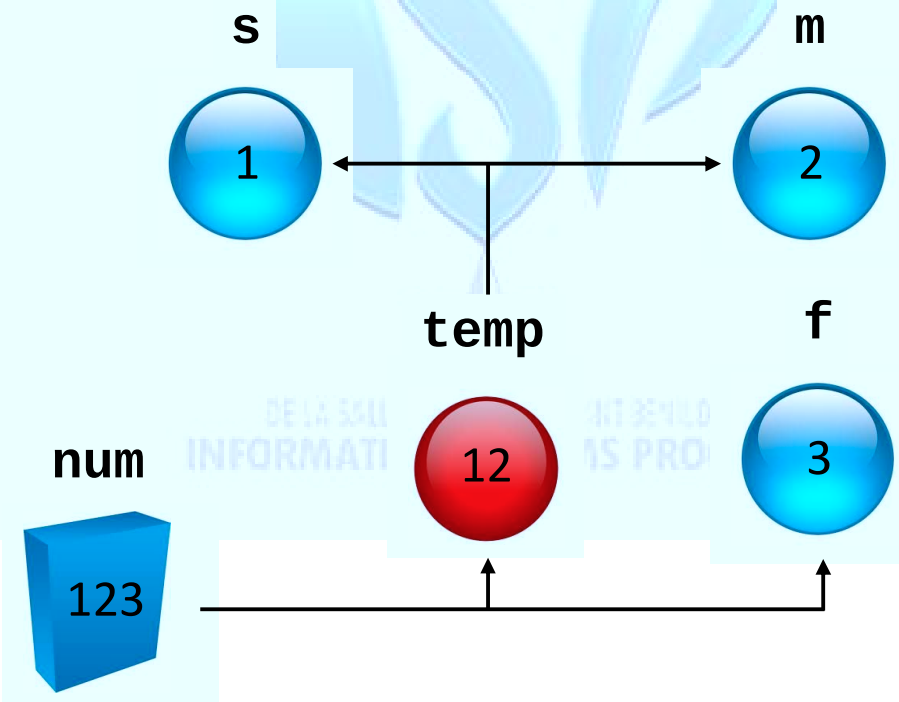


Note 1: Notice that dividing **num** by **10** essentially separates the value into two components, the quotient and the remainder. This fundamentally separates the value of **num** (which is **123**) into the quotient (which is **12**) and the remainder (which is **3**). The algorithm's goal at this point is to separate the digits of num into three 1-digit numbers (i.e. **123** separates into **1**, **2** and **3**). Now we have successfully separated the right-most digit from the value of **num** (see illustration below). We can continue separating the digits by continuing to do the same thing with the variable **temp** (which will be the next step of the algorithm).

2.) Divide **temp** by **10**, store the quotient in a variable **s** and store the remainder in another variable **m**.

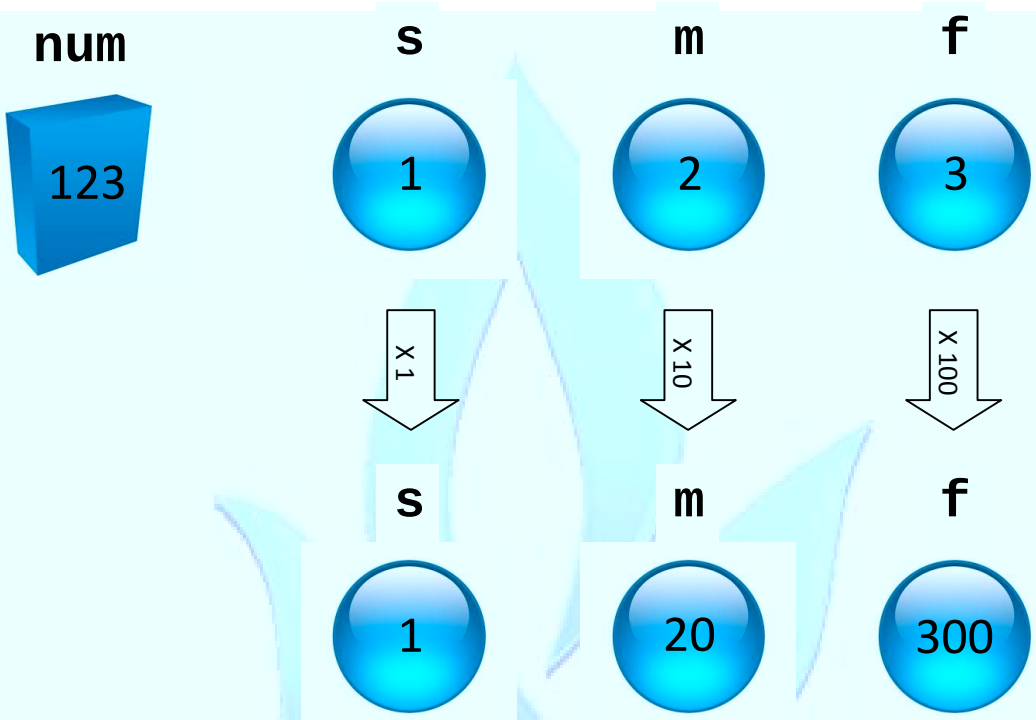


Note 2: This step essentially separates the value of **temp** (which is **12**) into its quotient (which is **1**) and its remainder (which is **2**). So now we have access to the three separate digits of the value of **num**.



The next few steps will attempt to re-assemble the three 1-digit numbers so that the digit orders are reverse.

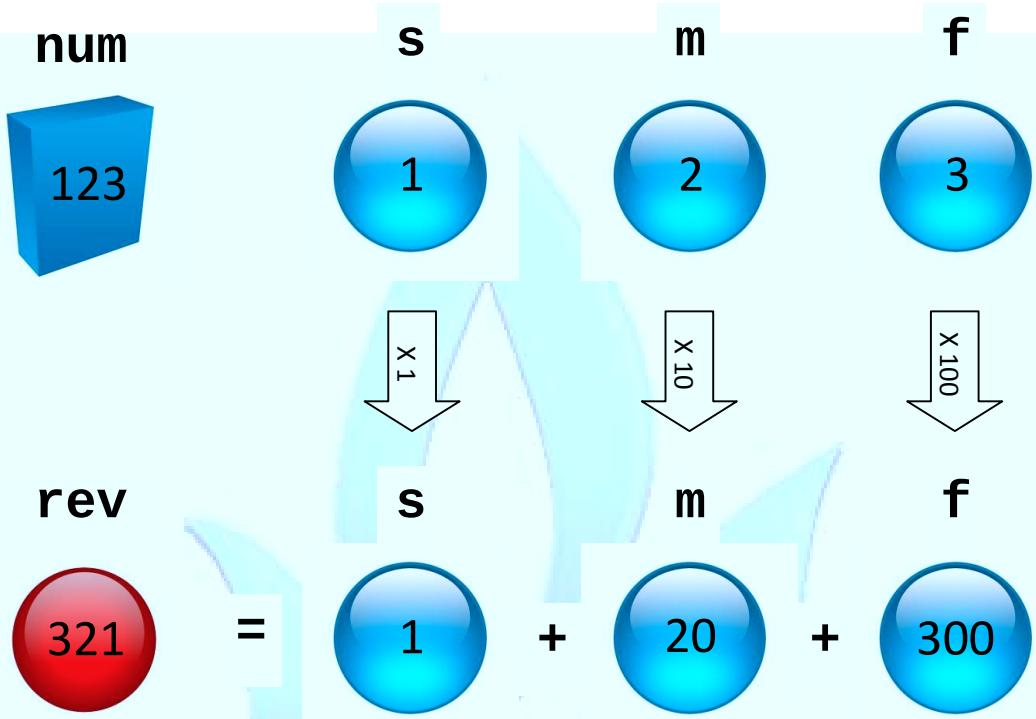
3.) Multiply the value of the variable **f** by **100**, the value of the variable **m** by **10** and the value of the variable **s** by **1**.



Note 3: Notice that multiplying a 1-digit number by **100** essentially pushes that number's value into the hundreds digit making it the left-most digit in a 3-digit number. Similarly, multiplying a 1-digit number by **10** essentially pushes that number's value into the tens digit making it the middle digit in a 3-digit number. And multiplying a number by **1** leaves the number unchanged thereby making it the right-most digit in a 3-digit number.

The final step of the process will combine the new values of the variables **s**, **m** and **f** into the reversed digit number of the original 3-digit number stored in **num**.

4.) Add the values of the variables **s**, **m** and **f** and store it in a new variable **rev**.



Note 4: Adding the values of the variables **s**, **m** and **f** will combine their values into a new number whose digits are the reverse order of the variable **num**. The resulting value is stored in the variable **rev**.

Algorithm 1: Narrative Summary

1.) Divide **num** by **10**, store the quotient in a variable **temp** and store the remainder in another variable **f**.
2.) Divide **temp** by **10**, store the quotient in a variable **s** and store the remainder in another variable **m**.
3.) Multiply the value of the variable **f** by **100**, the value of the variable **m** by **10** and the value of the variable **s** by **1**.
4.) Add the values of the variables **s**, **m** and **f** and store it in a new variable **rev**.

Algorithm 1: Narrative with Pseudo-codes

1.) Divide **num** by **10**, store the quotient in a variable **temp** and store the remainder in another variable **f**.

```
temp = num / 10
f = num % 10
```

2.) Divide **temp** by **10**, store the quotient in a variable **s** and store the remainder in another variable **m**.

```
s = temp / 10
m = temp % 10
```

3.) Multiply the value of the variable **f** by **100**, the value of the variable **m** by **10** and the value of the variable **s** by **1**.

```
f = f * 100
m = m * 10
s = s * 1
```

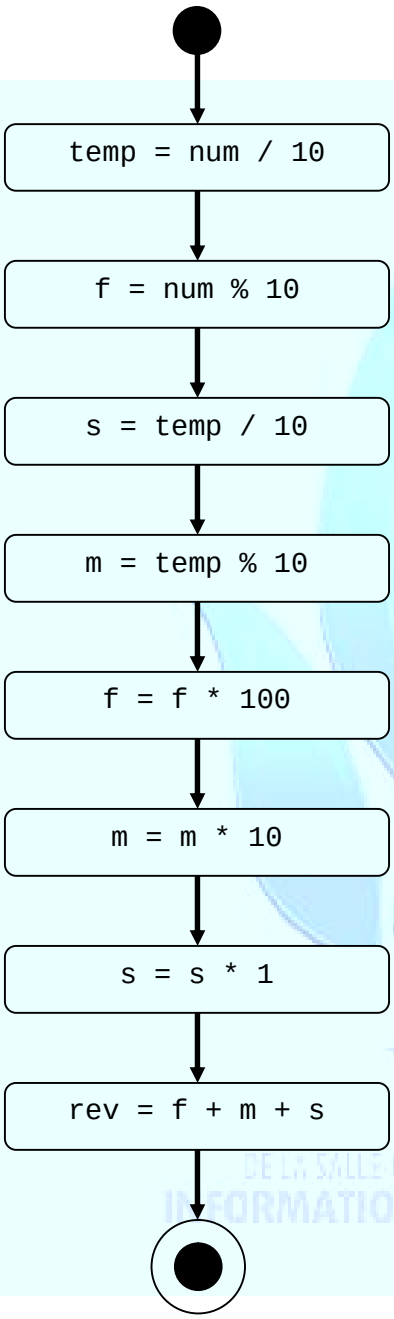
4.) Add the values of the variables **s**, **m** and **f** and store it in a new variable **rev**.

```
rev = f + m + s
```

Algorithm 1: Pseudo-codes

```
1.)     temp = num / 10
2.)     f = num % 10
3.)     s = temp / 10
4.)     m = temp % 10
5.)     f = f * 100
6.)     m = m * 10
7.)     s = s * 1
8.)     rev = f + m + s
```

Algorithm 1: Flowchart



Algorithm 2: *Assume that a 3-digit integer is stored in a variable* **num** *which is arbitrarily set to the value* **123.**

1.) Convert num to a string and store the value in a variable **`strNum`**.
2.) Get the first character (the left-most digit) from **`strNum`** by extracting the character using string indexing and store the character in a string variable **`a`**.
3.) Get the second character (the middle digit) from **`strNum`** by extracting the character using string indexing and store the character in a string variable **`b`**.
4.) Get the third character (the right-most digit) from **`strNum`** by extracting the character using string indexing and store the character in a string variable **`c`**.
5.) Concatenate the string variables **`a`**, **`b`** and **`c`** in the reverse order (that is **`c + b + a`**) and store the value in another variable **`d`**.
6.) Convert **`d`** to an **`int`** data type using **`System.Convert`** or **`Parse`** methods and store the value in an **`int`** variable **`rev`**.