About Integration in Teaching College Mathematics and Computer Programming

Yuri Yatsenko

This article deals with some challenges in teaching university mathematics and computer science courses. Colleges, textbooks and teaching techniques separate these areas, but students often take these courses simultaneously. Modern learning theories recommend that the teaching should be integrated in such cases. It does not mean that the basic subjects should be changed, but teachers should be aware of potential problems and how to avoid them.

Engineering colleges offer a degree in computer science (CS), and business colleges offer a degree in computer information systems (CIS). These majors require advanced mathematical skills and knowledge. Both CIS and CS majors first take an introductory course in computer science, which may be named differently (computer concepts, foundations of programming, introduction into programming logic and so on) but has the same basic curriculum: programming logic (PL) without using a computer and a specific programming language. Although the students' background (and prerequisites) for the PL course includes College Algebra and sometimes Calculus I or Finite Mathematics, they experience many difficulties in understanding the logic concepts. The author has extensive experience in helping students to overcome such problems.

As a general field, the programming logic is closely related to mathematics (see more in the last section below). However, this subject is taught in the context of engineering (CS) or business colleges (CIS); therefore, the practical skills are of primary importance. During the PL course, students are expected to understand and be able to develop several basic patterns of PL (like decision and iteration structures). More precisely, they learn how to read and write computer algorithms in the simplest formalized language (so-called pseudo-code). The PL course is a prerequisite for all further and more advanced CS courses, which may include specific programming languages (Visual Basic, C/C++, Java, Ada and others), software design and development techniques, database development and others. If a student has trouble passing this first course, he or she would not succeed in more sophisticated subjects that follow.

There are some essential differences between mathematical reasoning and PL that students often experience for the first time in a CS course. Students must understand these differences and learn how to use them correctly.

1. Equalities

Even the common equality formula,

Expression 1 = Expression 2 **(1)**

has different basic meanings in mathematics and PL. In mathematics, (1) sets a relationship between variables (that is, A = B) or represents an equation with respect to some unknown variables, for example $3x + 5 = 0$.

Expression (1) is an assignment statement in PL, which means that the right side of the expression is assigned to its left side at this step of the algorithm. A common form of the assignment statement (1),

$$X = X + 1 \quad (2)$$

at first glance seems senseless for some PL students, especially for those with a strong mathematical background. Indeed, there is no solution to the equation (2). The alternative two-step form of the same statement,

$$Y = X + 1, X = Y \quad (3)$$

is useful in the explanation of how the assignment statement works. However, it's worth to mention that the formula (3) is less efficient than (2) from the viewpoint of PL.

The Order of Operations and Parentheses

Another important issue students must understand to succeed in PL is the order of operations and the use of parentheses. Students usually know the basic rules for the arithmetic operations, but they often lack practice. The exercises are important because the

students soon learn about additional operations (logical, unary and binary).

As a CIS professor, I would expect mathematics teachers to practise more with the use of parentheses. Simple and useful examples with parentheses follow:

Problem 1

Evaluate the following expressions (that differ only by the number of parentheses):

No parentheses	With parentheses	Multiple parentheses
3 + 5 × 2 – 8 / 4 =?	(3 + 5) × 2 – 8 / 4	((3 + 5) × 2 – 8)/ 4
3 + 10 – 8 / 4	8 × 2 – 8 / 4	(8 × 2 – 8) / 4
3 + 10 – 2	16 – 8 / 4	(16 – 8) / 4
13 – 2	16 – 2	8 / 4
11	14	2

Students usually have more fun with the following problem:

Problem 2

Obtain all numbers from 0 to 10 using exactly five 2s and four arithmetic operations +, –, ×, ÷ (fill in the question marks with the operation signs). Do this (a) without parentheses, (b) with parentheses:

2 – 2 / 2 – 2 / 2 = 0
2 ? 2 ? 2 ? 2? 2 = 1
.....
2 + 2 + 2 + 2 + 2 = 10
.....

A challenging (bonus) version of the problem is to obtain as many integers as you can. We shall notice that the solution to problem 2 is not unique even without using parentheses; that is, the same results can be achieved in different ways. Generally, the existence of several solutions is much more common in PL than in traditional mathematics. Sometimes the best solution (the shortest, the most efficient and so on) can be found among several acceptable solutions, but it requires changes in the statement of a problem under study and some advanced techniques.

Vectors and Arrays

Arrays are a common technique in PL. From a mathematical viewpoint, the arrays correspond to vectors and matrices. However, even after completing a College Algebra course, students often have problems understanding and using even one-dimensional arrays. The main difficulties are in differentiating between an array and its index (subscript), more generally, between independent and dependent variables. A possible reason lies in the shifted focus; namely, the PL students need to define, fill in and handle arrays, while in mathematical courses these tasks are supposed to be solved before a problem starts.

As compared with college mathematics, a new concept is storing data (information) in arrays. In PL, a key feature of arrays is that they can keep (store) data values instead of immediately processing them one-by-one and forgetting them. To clarify this point, the following problem is useful. It also illustrates a version of the pseudo-code for mathematics teachers.

Problem 3

The sum or the average (mean) value of an arbitrary number of input data values may be calculated without saving the values. A possible algorithm includes three variables *X*, SUM and COUNTER and one iteration loop to accumulate the sum:

```
COUNTER = 0
SUM = 0
LOOP WHILE COUNTER < MAX
  INPUT X
  SUM = SUM + X
  COUNTER = COUNTER + 1
END LOOP
AVERAGE = SUM / COUNTER
```

However, to calculate and output the individual deviations of the data values from the mean value, all input data values should be stored; that is, the above algorithm needs to use an array *X*() of the dimension MAX instead of the simple variable *X* and at least two loops (to accumulate the sum at first and to calculate the deviations next):

```
COUNTER = 0
SUM = 0
LOOP WHILE COUNTER < MAX
  INPUT X (COUNTER)
  SUM = SUM + X (COUNTER)
  COUNTER = COUNTER + 1
END LOOP
AVERAGE = SUM / COUNTER
COUNTER2 = 0
LOOP WHILE COUNTER < MAX
  COUNTER = COUNTER + 1
  DEV = X (COUNTER) – AVERAGE
  OUTPUT DEV
END LOOP
```

The first code is more efficient; it consumes less computer memory and does not depend on the length of the input data file. However, this type of processing is possible for very restricted and simple tasks only. The code with an array is much more general.

We shall notice that the second code calculates the individual deviations rather than the standard deviation. Using basic statistical rules, the latter may be calculated without an array.

Similar calculations are very common in college mathematics and statistics, so it would be worth to stress the above algorithmic problems while studying statistics.

Decimal and Binary Numbers

During the PL course, students often gain knowledge of binary numbers for the first time. Computers use binary rather than decimal numbers to store data in memory. Binary numbers are the numbers "in base 2" and use only two digits (0 and 1). Examples follow:

- Decimal number 9 is binary number 1001: $(9)_{10} = 8 \times 1 + 4 \times 0 + 2 \times 0 + 1 = (1001)_2$
- Decimal number 15 is binary number 1111: $(15)_{10} = 8 \times 1 + 4 \times 1 + 2 \times 1 + 1 = (1111)_2$
- $(10001)_2 = (23)_{10}$, $(100011)_2 = (35)_{10}$ and so on.

The exercises like the problem below are used in the PL course but they can be also mentioned in the college mathematics.

Problem 4

Do simple conversions from decimal numbers to binary numbers and vice versa:

$(7)_{10} = (111)_2$

$(23)_{10} = (1101)_2$

$(101)_2 = (5)_{10}$

$(1101)_2 = (13)_{10}$ and so on.

Doing simple arithmetic with binary numbers helps students better understand the common base-10 numeration system as well. For example, the sum of two binary units

$(1)_2 + (1)_2 = (10)_2$

makes the rule of moving the unit to the decimal location of the next higher order more clear. It also demonstrates the case when "one plus one is not two" in mathematics.

Generally speaking, the programming logic stands out as the first consumer of abstract mathematical knowledge that students have learned in a business or engineering college. First, the students begin to feel and understand the applied character and practical importance of mathematical theories. On the other hand, as the above examples show, the programming logic can enrich the students' understanding of mathematics itself.

The section that follows is more about advanced relations between mathematical theory and computer algorithms practice, and they are worth dealing with in both mathematics and CS courses.

Advanced Relations Between Mathematics and Programming Logic

As a scientific discipline, the programming logic is closely linked to several fields of advanced university mathematics, such as computational mathematics, numeric methods, optimization and complexity of algorithms. Generally speaking, these mathematical fields investigate what operations are necessary to solve a specific mathematical problem and the programming logic shows how to implement them in a form suitable for computer implementation.

After a new applied problem arises in science or engineering practice, it usually passes through the following stages in its computer investigation.

Stage 1. Developing an algorithm for solving the problem (any algorithm)

This stage uses a combination of numeric methods and PL. The first developed algorithm is used for a while to investigate the problem for different initial data. It normally reveals some flaws and needs an improvement. The next stage is also the subject of numeric methods and PL.

Stage 2. Developing various algorithms for different situations

At this stage, a set of different algorithms is obtained and some experience in solving the problem numerically is gained. Then one can formulate a problem of finding the best possible algorithm under certain assumptions. Mathematically correct definitions of "the best possible algorithm" include the most accurate algorithm under given restrictions on initial data errors, the fastest algorithm under given restrictions on the accuracy of results, the most robust algorithm and so on. Such problems are investigated by the theory of algorithms optimization and complexity. They may be used for the next step.

Stage 3. Finding the best (optimal) algorithm under the given specific assumptions

Even after finding an optimal algorithm, the analysis may not be finished. Different optimal algorithms can be obtained under various assumptions about initial data and tested on real data and real problems. This leads to more experience in numerically solving the problem, which results in the next step.

Stage 4. Obtaining a set of efficient algorithms optimal in different situations

Starting with the first step, an applied outcome of the above mathematical theories is standard software programs for solving specific problems. The exposed scheme has been used in the development of many efficient software packages for statistics, sorting and filtering data, solving differential equations, optimization and so on.

Of course, not all real-life problems are undergoing such theoretical treatment. It depends on their importance and specifics. In some cases, the first algorithms appear to be good enough for practical purposes. Other problems never become standard and are occasionally solved using custom algorithms and software.

Isaac Newton (1642–1727)

This British mathematician wrote in his book *Arithmetica Universalis*: "In my studies I discovered that the actual problems are often of more value than the rules." He posed the following problem:

Three pastures have an area of 3 1/3 ha, 10 ha and 24 ha. The growth conditions are identical in all three pastures. The grass density and yield per unit area are the same. On the first pasture, 12 oxen graze for the duration of 4 weeks and on the second pasture 21 oxen graze for the duration of 9 weeks, at which time these pastures have been completely grazed. How many oxen could graze on the third pasture for the duration of 18 weeks?