Using Spreadsheet Calculations to Demonstrate the Importance of a Correct Problem Specification

Gabriela Lovászová
glovasz@ukf.sk
Univerzita Konštantína Filozofa
Nitra, Slovakia

Jozef Hvorecký
hvorecky@cutn.sk
Vysoká škola manažmentu
Bratislava, Slovakia

Abstract: "Experiment", "discovery" and "self-teaching" are basic bricks of active learning. Despite the fact that learners are investigating notions and relationships known long before, their own discovery of something new and unknown for them results in their fascination with an irreplaceable educational value. Spreadsheets are one of the environments that can successfully be used for building stages in similar experimentation. Activities described in our paper show how spreadsheet-based calculations can lead students to a deeper comprehension of algorithms, their design, development and execution. As only well-specified problems can be solved using algorithms, deep comprehending of the relationship between the formulation of a problem and its solution(s) belongs among programming basics. The goal of our paper is to exemplify how the problem specification can be introduced using our method. Students are given a badly formulated problem and asked to solve it. Students build their spreadsheet solutions, observe many executions of their proposed solutions, formulate hypotheses and to verify them. Due to the ambiguous formulation, they get incompatible results. Using a class discussion we start clarifying the origin of confusions, realize why it happened and what should be done to avoid it in the future. As a result, the way to obtaining this core concept does not depend on a particular programming language and can be practiced *before* the students start using it. We do not want to replace program composition training by our proposed method – we are using it for building high-quality fundamentals before our students go to program design and development.

1. Introduction

This paper is a free continuation of our papers [1] and [2]. In both, spreadsheet calculations are used as tools for demonstrating selected features of computer programs, algorithms, and explain relationships between the outer form (i.e. notation) of a program and its execution. The motivation for a similar teaching and learning method is obvious – students coming to our programming courses usually have passed computer literacy training. They are familiar with at least one algorithm-based method of problem solving – spreadsheet calculations.

In our present paper, we point our attention to "problem-specification" issues. Software developers meet with them often. Different readings of the same text lead to misunderstandings which often result into program failures [3]. Vaguely specified problems have several "correct" (but not equivalent) solutions. The interpretation selected by the programmer can be different from that intended by the customer. If a problem has several interpretations, talking about its correctness is meaningless – all interpretations are equally "good". Only problems specified to the level of exactness that it allows a unique interpretation are solved in a unique way (satisfying both partners). All future programmers should comprehend similar interdependence between a problem's solution and its exact specification. Our aim is to give to our students an opportunity to experience it.

Using Josephus' problem [4], we demonstrate how the ambiguities are born. Its original text is vague and for our aims it has been intentionally "weakened" to allow even more interpretations. Our examples are based on our students' results and demonstrate typical solutions with different levels of difficulty - from a trivial problem to unsolvable one.

2. Josephus' problem

Josephus Flavius, a Roman historian of Jewish origin, describes how he and his friend were captured by Roman soldiers during an uprising. Romans ordered their detainees to make a formation, counted and executed every third person. This continued until two persons remained. These last survivors had been freed. Flavius states that he was capable of calculating these "safe" positions in advance. Due to his calculation, both he and his friend survived.

Notice also that Josephus' original formulation speaks about forty-one detainees; due to limited space, our below illustrations display the case of seventeen people only.

3. Solutions

What is questionable in Josephus' story? Apparently, it could be made up later as there is no witness except his mysterious surviving friend. Also – and this is the subject of our paper – the elimination method is not described properly. The different interpretations result in calculation methods with different "winning" positions. How could Flavius know which of them was going to be applied?

When the problem is presented to our students, the danger of different interpretations is not mentioned. They are told the problem and asked to program an elimination method and implement it using a spreadsheet program. The program must be completed in such a way that displays "winning positions" for any number of detainees. Then we ask them to find regularities in their solutions and express the formula Josephus applied for his survival. To students' surprise, several solutions are proposed. A discussion on the cause of discrepancies is then commenced. It reveals two critical places in the problem specification:

a) **Formation**: To make counting possible, the formation must have a simple regular shape in which detainees form a sequence. For that reason, two formations look natural and are preferably used by students – a line and a circle.
b) **Starting position**: The elimination results are strongly determined by the position of the first counted person. Thus, it is very important to specify "who is the first" and "what happens to the person" as the differences lead to different surviving positions.

In our examples below we show typical interpretations of Josephus' problem.

3.1 The Detainees Standing In a Line, the Third Detainee Is Executed

When detainees form a line the position of the "first" is obvious. Still, there is a problem: Is the first person "simply counted as first" or "executed as first"? Figure 1 shows the situation of 17 detainees when the first one is counted as first, i.e. the third person in the line is executed first. Its rows represent the rounds of elimination – during each round counting begins from the leftmost person and every third person is executed.

For N detainees, the first row of the spreadsheet contains the numbers 1 to N. The cell A2 contains the formula

```
=IF(MOD(COUNTIF($A1:A1;">0");3)=0;0;A1)
```

Its meaning is: Starting from the beginning of the previous row to the given column, we count the number of non-zero values using the COUNTIF function. The starting column is fixed using the dollar sign in front of the first A. Then, the remainder after division by three of the intermediate result is calculated. If the remainder equals zero, then the content of the cell is changed to zero (the detainee has been executed); otherwise it remains unchanged (the detainee survived the current round). To simplify the perception of results, the zeroed elements are hidden using conditional formatting.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
2	1	2		4	5		7	8		10	11		13	14		16	17
3	1	2			5		7			10	11			14		16	
4	1	2					7			10				14		16	
5	1	2								10				14			
6	1	2												14			
7	**1**	**2**															

Figure 1 First two detainees always survive

All other cells in all rows (except the first row) contain "spreadsheet-like" copies of the above formula. For example, the content of the cell E7 is

```
=IF(MOD(COUNTIF($A6:E6;">0");3)=0;0;E6)
```

As one can see, the first and second detainees are not executed. The first and second positions are "winning" ones regardless of the number of detainees. In each round, their nearest neighbour to the right is the first eliminated. Sooner or later, anyone else, including people in very distant positions, becomes "the third". Therefore, the problem is trivial: "1" and "2" are surviving positions for any N; no formula is needed.

3.2 The Detainees Standing In a Row, the First Detainee Is Executed

A small modification in the above formula for A2 causes the first person in the row to become the first victim:

```
=IF(MOD(COUNTIF($A1:A1;">0");3)=1;0;A1)
```

	A	B	S	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
2		**2**	3		5	6		8	9		11	12		14	15		17
3			**3**		5			8	9			12		14			17
4					**5**			8				12		14			
5								**8**				12					
6												**12**					

Figure 2 The last two numbers in the sequence 2, 3, 5, 8, 12, 18, … specify survivors

The formula is almost identical, just the tested value of the remainder is 1 instead of zero. The cells in which the MOD function produces this value are now cleaned. Guessing the surviving position becomes more complex as shown in Figure 2. The survivors are the detainees 8 and 12. With the growing number of detainees, the "safe" positions are 2, 3, 5, 8, 12, 18, 27, 41, 62, 93, etc.

For a given number of detainees N, the solution of the problem (the surviving positions) are the two greatest values of this sequence not exceeding N. For example, for the original Josephus' problem with 41 persons, the detainees with numbers 27 and 41 become survivors. The same values are also solution for any higher number up to 61 (including it). For the number of detainees between 62 and 92, the safe positions are 41 and 63, and so on.

3.3 The Detainees Form a Circle, the Position of the First One Is Specified

Organizing detainees in a circle makes the simulation much more complex. In the above simulations, each row of the spreadsheet corresponds to one round of elimination and all calculations therefore start from the row's left end. The value of the remainder is also the same - 0 in the first elimination method, 1 in the second.

Such regularities do not hold in circles. Here, the remainder changes from round to round depending on the factual number of survivors, because the sequence "never ends". The value of the remainder can only be stated before the first round. As above, it is zero when "the first counted person is used as the beginning of counting", and one when "the first person is executed as the first". In all next rounds, the remainder value depends on the number of people between the last eliminated person and the first survivor from the previous round.

The simplest simulation of the circular method uses pairs of rows. The first row in each pair (shaded in Figure 3) contains the numbers of surviving detainees in similar manner as above. For second row, only one cell is filled in – the value of the remainder applied in the given round. In Figure 3, the first remainder is 1 – the detainee counted as the first is also executed as the first.

Figure 4 shows of the first remainder equal to 0 – the first detainee is simply the beginning point of counting and the third counted person is executed as first[1].

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Starting formation
2	1																	
3		2	3		5	6		8	9		11	12		14	15		17	1st round
4	2																	
5		2			5	6			9		11			14	15			2nd round
6	0																	
7		2			5				9		11				15			3rd round
8	2																	
9		2							9		11							4th round
10	0																	
11		2							9									5th round

Figure 3 "The first is executed first" in a circle

[1] There exists a third option – the second person in the row is executed as first. In this case, the initial value of remainder equals 2. The solution does not appear in students' solution probably it is do not refers to "common sense".

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Starting formation
2	0																	
3	1	2		4	5		7	8		10	11		13	14		16	17	1st round
4	1																	
5		2		4			7	8			11		13			16	17	2nd round
6	1																	
7				4			7				11		13				17	3rd round
8	2																	
9				4							11		13					4th round
10	0																	
11				4							11							5th round

Figure 4 "The third is executed first" in a circle

The two calculations only differ by this constant, everything else is the same. The first row contains numerical constants identifying detainees. The first cell in the third row contains the formula

```
=IF(A1=0;0;IF(MOD(COUNTIF($A1:A1;">0");3)=$A2;0;A1))
```

All cells in odd rows contain its "spreadsheet copy". Using conditional formatting, zeroed elements in these rows are colored in white (and become invisible).

The number of people in the circle is decreasing after each round. The counting continues with a possibly different the value of the remainder. After the first round, its new value is calculated using the formula (typed into A4):

```
=MOD(3-MOD(COUNTIF(3:3;">0");3)+$A2;3)
```

Again, its copies are used for calculating the following remainders and placed into the first cells of even rows.

3.4 The Detainees Form a Circle, the Position of the First One Is Not Specified

There is one principal problem with people standing in a circle – there is in fact no "first person" because circles have no beginning or end. In reality, every detainee in the formation can be randomly chosen as "the first".

Both calculations in Section 3.3 are based on an assumption that the position of the first has been specified in advance and is fixed. Under this condition, they work properly and the positions of two survivors can be calculated. Such an assuption is very unrealistic. This implies that there is no appropriate initial value, and consequently no algorithmic solution. The process still can be simulated by placing another constant into the second row. Its value is between 1 and the total detainees and indicates the position of the "first". Figure 5 shows the case when it equals 7. Again, two selection methods ("counted as first" or "executed as first") can be alternatively used.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	17	Number of people															
2	7	First person															
3	7	8	9	10	11	12	13	14	15	16	17	1	2	3	4	5	6
4	0	Counted as first: 0; executed as first: 1															
5	7	8		10	11		13	14		16	17		2	3		5	6
6	1																
7		8		10			13	14			17		2			5	6
8	1																
9				10			13				17		2				6
10	2																
11				10							17		2				
12	0																
13				10							17						

Figure 5 Simulating a circle with a random starting position

There are three constants determining the result:

- the cell A1 contains the number of detainees,
- A2 indicates the randomly selected “first” person,
- A4 specifies the remainder i.e. the elimination method.

The rotating pattern of the sequence in the third row is generated as follows. The value in its first cell (i.e. A3) is copied from A2 (using the formula =A2). The second cells contain the formula

```
=IF(COLUMN()>$A$1;0;MOD(A3;$A$1)+1)
```

All other cells contain its relevant copies. For the cells in columns less or equal to the number of detainees, it calculates the detainee’s number. The cell farther to the right gets the value “zero”; and due to conditional formatting seems to be empty. Naturally, depending on the initial choice of the parameters, any detainee now can survive.

4. Conclusion

What is the educational value of similar examples?

The simplicity of our problem is its first advantage. At the beginning, students feel certain that they fully understand the problem and can successfully solve it. They are later very surprised that other students’ understanding of the same text is completely different. It is likely that in the future they will not be so confident when reading a problem specification. They will hopefully be aware of a chance of its different interpretations and become more cautious and prudent.

Due to the use of spreadsheets, the solutions are very transparent. Consequently, our students have no problems reading and understanding their peers’ solutions. They start soon comprehending that their peers’ solutions are also “correct” even if they are incompatible and not-interchangeable. They realize that their peers simply solved different problems. Comprehending the fact that the same text has been “correctly” interpreted in many ways is the most important goal of this exercise.

As we stressed in our paper [2], every spreadsheet calculation can be seen as a complete “program trace”. Thus, one cannot only determine two survivors, but also see the stepwise way in which the detainees are eliminated in each individual round. Such observations play another important and supportive role – they help to students understand the method their peers designed and developed. The degree of visualisation is evidently better compared to the same solutions written in a programming language. In the later case, the output formats are very individual and – if very few or no intermediate results are displayed - may fog the principle of the method.

Then we start asking students how to avoid similar misinterpretations. The answer is obvious – it is necessary to get "insider's information" – a complete set of all problem-related data that are known to those who formulate the problem. The customers are unknowingly and unwillingly "hide" this information as they are not aware that their partners are not familiar with it. If the communication channels between customers and programmers fail because each partner has a different set of pieces of knowledge on the problem, the term "correctness of solution" loses its meaning. Better saying, all interpretations that not contradict to the original formulation, lead to "correct" solutions. Naturally, this is not the way in which "correctness" is interpreted by common people.

The originators have usually a good picture what their requested solution should look like. They simply fail to supply all information to their partners. To solve the problem in their presumed way, the solver must pull together its complete and unambiguous specification. In other words, the only way to move from imperfect specification to a perfect one is a stepwise clarification of the problem by collecting all relevant facts. Searching for missing ones and requesting more exact and complete specification of all details is therefore a crucial part of problem solving.

For these reasons, both customers and developers should train and develop their abilities to specify problems as perfectly as possible. By meeting with similar vaguely defined problems as the Josephus problem and by solving them, our students become understand the risks related to wrong specifications and accustom to a systematic hunt for perfectly designed ones.

5. Historical Remarks

Let us repeat two sentences from the previous chapter: "Then we start asking students how to avoid similar misinterpretations. The answer is obvious – it is necessary to get "insider's information" – a complete set of all problem-related data that are known to those who formulate the problem." The last conclusion raises a question concerning Josephus Flavius' personal integrity. The problem ("what of many elimination methods to apply") was formulated by Roman soldiers. To solve it correctly (i.e. to calculate "safe positions"), he had to have "insider's information". This, in fact, accuses him of being a traitor because as he could only get it from his enemies. As Flavius reached a high position in the Roman Empire, this version can not be excluded.

On the other hand, there exists another logical explanation in which he does not look so negative. Flavius speaks about 41 detainees. This number plays a role in only one of the above interpretations: <u>a row in which "the first is executed first"</u>. Flavius was the last in the row (possibly hiding himself till the last moment) and miracuously survived. Then he might interpret it – even for himself – as a purposeful calculation. (The calculation of surviving positions in this case is so complex that it is unlikely that he really did it.)

One detail in his formulation of the story makes this version even more probable. The middle of circle is the only reasonable position for the person performing the counting. Nevertheless, it is the most dangerous position, too, as the detainees – with their minimum chances for survival – may attack him. Keeping them in a row substantially simplifies their control.

On the other hand, when "the third is executed the first" elimination method is used, first two detainees survive regardless of their number. This method lacks the element of uncertainty –

the reason for choosing a complex elimination method. So, it is unlikely that soldiers did use it. Thus, if Josephus was not a traitor, the “single-row in which the first person is executed first” was used. He was very likely the last in the row (and simply lucky).

References

[1] 1. Hvorecký, J. (2001): Spreadsheets to Programming. Wei-Chi Yang, Sung-Chi Chu, Zaven Karian, Gary Fitz-Gerald (editors): Proceedings of the Sixth Asian Technology Conference in Mathematics, ATCM, Melbourne.

[2] Lovászová, G., Hvorecký, J. (2003): On Programming and Spreadsheet Calculations. Spreadsheets in Education, Vol. 1, No.1, pp. 44-51

[3] D. Budgen (1994): Software Design. Addison-Wesley

[4] Graham, R.L., Knuth, D.E., Patashnik, O. (1994): Concrete Mathematics. Addison-Wesley