Journal of Mathematics and Science Teacher
2024, 4(1), em053
e-ISSN: 2752-6054
https://www.mathsciteacher.com/

MODESTUM

Research Article **OPEN ACCESS**

Empowering mathematics education through programming

Thomas Lingefjärd [1]*

[1] University of Gothenburg, Gothenburg, SWEDEN
***Corresponding Author:** Thomas.Lingefjard@gu.se

Citation: Lingefjärd, T. (2024). Empowering mathematics education through programming. *Journal of Mathematics and Science Teacher, 4*(1), em053. https://doi.org/10.29333/mathsciteacher/13847

ARTICLE INFO

Received: 11 May 2023

Accepted: 12 Oct. 2023

ABSTRACT

One of my last assignments at the university of Gothenburg was to teach a sequence of three seminars in programming for prospective teachers (*n*=37). The three seminars are given in the introduction of this manuscript. Since this was a course for prospective upper secondary teacher of mathematics, it was decided that it should be a course in programming for learning mathematics. This manuscript is a research article but also a manuscript about programming in GeoGebra, Python, and Wolfram Alpha. The examples I shared with my students led me to write a book about programming. See the reference list.

Keywords: programming, GeoGebra, Python, Wolfram Alpha, numbers

INTRODUCTION

Programming environments are good at calculations, to sum numbers, to solve equations, to find correlations between sets of values, and to find maximal value of a function. In mathematics and physics, we investigate different projectile's motion such as a golf balls flight towards green, head or tails with coin tossing, or to throw dice. To program that kind of situation is also a way to analyze and learn more about the situation. I selected the three programming environments GeoGebra, Python, and Wolfram Alpha for my seminars (Lingefjärd, 2022).

GeoGebra has at present time over 100 million users around the world. Python is considered the fastest growing text-based programming language in the world and Wolfram Alpha gets over 1.5 billion hits on their web site every day. These three programming environments are used by many people every day.

It is possible to run GeoGebra files on applications in smartphones, other mobile electronic devices, and computers over the Internet.

Python has a support for so called *arbitrary-precision integers*. We can add 25 digit numbers in Python and get the correct result, which is impossible to do in Excel or in GeoGebra. When we program in Python we need a compiler. This manuscripts use the web compiler *repl.it* as compiler. Links to *repl.it* are possible to execute over any device with the Internet connection, such as smartphones, other mobile electronic devices, and computers.

When you program in GeoGebra and Wolfram Alpha all the possible coding is there in the program. In Python we can import many different modules to call the desired command in the correct way.

Wolfram Alpha is a powerful system for mathematics but can also answer questions about other subjects. Since code in all three programming environments is available through GeoGebra material, through *repl.it*, and through Wolfram Alpha's web page, it is possible to distribute all the programming in a convenient and controlled way.

Learning With Programming

Pseudo code is a programming description that does not require strict programming syntax or underlying technology considerations. It is used for creating a rough draft of a program code. *Pseudo code* summarizes a program's flow but excludes underlying details.

Pseudo code will offer you, after you tested how to write *pseudo code* over how to solve an equation, to learn what an equation is at a deeper level. In that sense is the process of programming and also the programming environment a sort of amplifier but also a reorganization of what content you are studying or programming.

Tinkering

When learning how to program, you can follow the instructions line by line to write a program, but you can also learn by changing some values in a code that someone else wrote. An example activity is to study a code and trace the code saying exactly what each command will make happen when it runs.

Tinkering is a common way to solve programming exercises. When we are looking at the code for summarizing the first nine natural numbers, we also see the possibility to summarize for instance the first 20 natural numbers and how to alter the code in order to fulfil that purpose.

Seminar 1

Sum of one-digit natural numbers

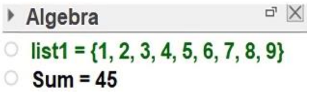
The sum of the natural numbers 1+2+ ... +9 is something students should know about in the end of school year 3 and is something most students should be acquainted to. Here can we investigate how programming may show this addition and summarizing and at the same time we will learn how to enlarge the situation.

Pseudo code:

```
Make the computer list the first nine natural numbers

Make the computer sum the listed numbers

Print the sum
```

This *pseudo code* works for GeoGebra, Python, and Wolfram Alpha.

GeoGebra



list1 = {1, 2, 3, 4, 5, 6, 7, 8, 9}

Sum = 45

Figure 1. GeoGebra summarize first nine natural numbers (Source: Author's own elaboration)

This is very easy to do in GeoGebra. Please try out the commands at your local GeoGebra program.

Commands: list1=Sequence (*k*, *k*, 1, 9) and Sum=Sum (list1).

Try out these commands at your local GeoGebra installation.

Figure 1 shows how GeoGebra summarizes the first nine natural numbers.

Python

```
x=9

print ("\n")

for i in range (1, x+1):

   numbers=range (1, i+1)

   print('+' .join (map (str, numbers)),'=', sum (numbers))
```

See https://repl.it/@ThomasLinge/The-sum-of-the-first-single-digit-natural-numbers for more details.

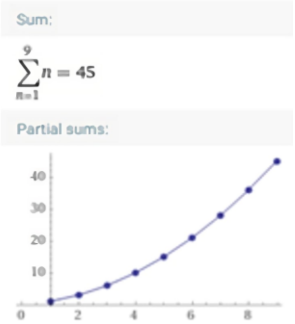
Results

```
1=1

1+2=3

1+2+3=6

1+2+3+4=10

1+2+3+4+5=15

1+2+3+4+5+6=21

1+2+3+4+5+6+7=28

1+2+3+4+5+6+7+8=36
```

```
1+2+3+4+5+6+7+8+9=45
```

The way we are coding Python to show the summation process also reveals another way of looking at numbers. We see the triangle numbers. Triangle numbers tend to occur when stocking round objects in piles or in combinatorics.

Wolfram Alpha



Sum:

$$\sum_{n=1}^{9} n = 45$$

Partial sums:

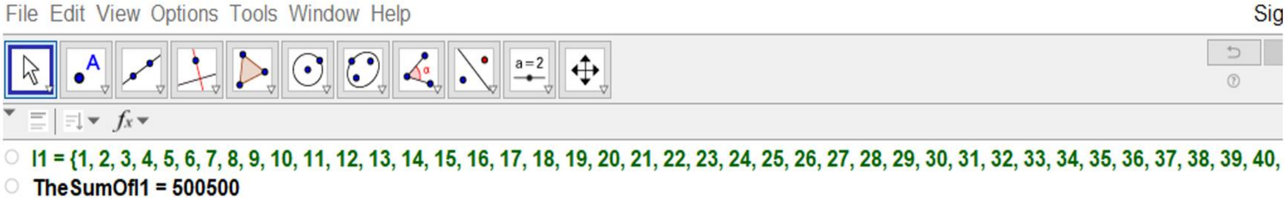
Figure 2. Wolfram Alpha summarize first 9 natural numbers (https://www.wolframalpha.com/input/?i=1%2B2%2B3%2B...%2B9)

Now you can use *tinkering* and expand the summation the way you like. You should be able to do that in all the three programming environments without too much work.

Tinkering: Sum the first 1,000 natural numbers.

GeoGebra

The commands are Sequence(*k*, *k*, 1, 1000) and Sum(l1). Try the commands in your local installation (See **Figure 3**).



○ l1 = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,

○ TheSumOfl1 = 500500

Figure 3. GeoGebra summarizes first 1,000 natural numbers (Source: Author's own elaboration)

The result is 500,500.

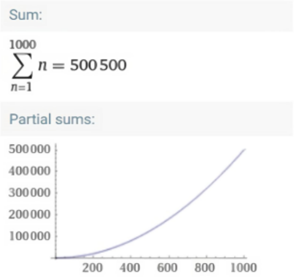
Python

```
num=1000

hold=num

sum=0

while num>0:

   sum=sum+num

   num=num-1;

print (“Sum of first”, hold, “natural numbers is:”, sum)
```

See https://repl.it/@ThomasLinge/sum-of-first-1000-natural-numbers for more details.

Sum of first 1,000 natural numbers is 500,500.

Wolfram Alpha



Sum:

$$\sum_{n=1}^{1000} n = 500500$$

Partial sums:

Figure 4. Wolfram Alpha summarize first 1,000 natural numbers (https://www.wolframalpha.com/input/?i=sum+%281%2B2%2B3%2B...%2B1000%29)

Seminar 2

The sum of even natural numbers

The sum of even natural numbers such as 2+4+ ... +*n*, where *n*=2×*k*, *k*∈**N,** is also possible to investigate with programming that can help us visualize possible sums when *n* is varying. We can ask ourselves questions like:

1. How many terms are needed to get a sum larger than 10,000?
2. Can we get the sum 100?

GeoGebra

The commands: Sequence (2*k*, *k*, 1, *n*), where *n* is a slider 1<*n*<1,000 with increment=1. Text in the graphics window is a mixture of text and command calls according to (where sum and *n* are local calls for variables): Text1: The sum of 2+4+… + 2×*n*=sum.

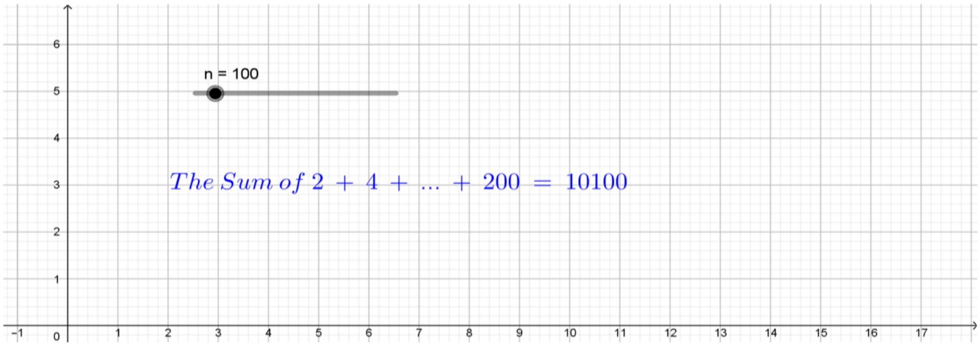


Figure 5. GeoGebra summarize even numbers

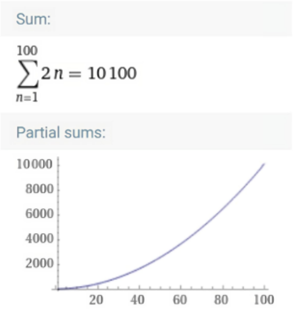
You find the GeoGebra file at the address: https://www.geogebra.org/m/qzcppgng.

Python

```
number=int (input (“Give last number for the sum 2 + 4 + ... + n:”))

n=number

if number<0:

   print (“A positive number is required”)

else:

   sum=0

   for even_number in range (0, number+2, 2):

            sum+=even_number

print (“\nThe sum 2 + 4 + ... +”, n, “=“, sum)
```

See https://repl.it/@ThomasLinge/The-sum-of-even-natural-numbers for more details.

Wolfram Alpha



Sum:

$$\sum_{n=1}^{100} 2n = 10100$$

Partial sums:

Figure 6. Wolfram Alpha summarize even numbers (https://www.wolframalpha.com/input/?i=2%2B4%2B...%2B200)

Example: Sum of all four digit numbers divisible with three.

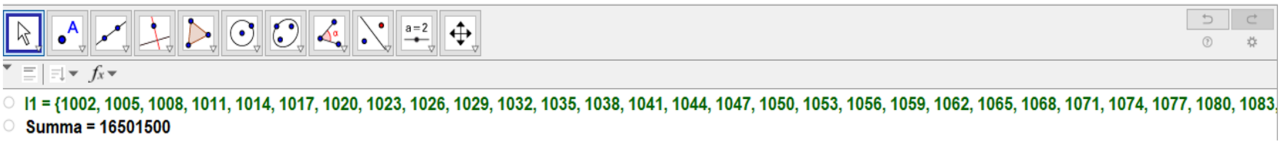
Pseudo code (the same for all three programming environments)

1. We need a variable that we call sum.
2. We define an area, where we summarize four digit numbers.
3. Numbers divisible with three comes every third step on the number line.
4. Which is the first four digit number divisible with three?

The number 1,002 is divisible with three and must be the first four digit number with this property. We divide 1,002 with three and we get 334. The largest four digit number divisible with three must be 9,999. We divide 9,999 with three and get 3,333.

$$\sum_{k=334}^{3,333}(3k), k \in N.$$

GeoGebra



```
l1 = {1002, 1005, 1008, 1011, 1014, 1017, 1020, 1023, 1026, 1029, 1032, 1035, 1038, 1041, 1044, 1047, 1050, 1053, 1056, 1059, 1062, 1065, 1068, 1071, 1074, 1077, 1080, 1083,
Summa = 16501500
```

Figure 7. GeoGebra summarize four digit numbers

Use the command Sequence. Please see similarities and differences in our three programming environments now. The command for GeoGebra is Sequence (3*k*, *k*, 334, 3,333).

The GeoGebra file is at https://www.geogebra.org/m/mj9nqzzm.

Python

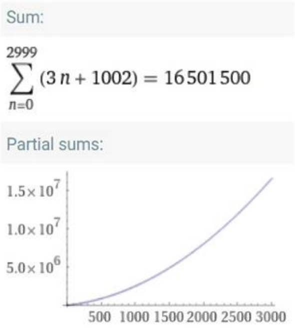
```
sum=0
for number in range (1,002, 10,000, 3):
   sum+=number
print (“The sum is:”, sum)
```

See https://repl.it/@ThomasLinge/Sum-of-four-digit-numbers-divisible-by-3 for more details.

The sum is 16,501,500

Wolfram Alpha

See the mathematical syntax with three dots.



Sum:

$$\sum_{n=0}^{2999} (3n + 1002) = 16501500$$

Partial sums:

Figure 8. Wolfram Alpha summarize four digit numbers (https://www.wolframalpha.com/input/?i=summarize+(1002+%2B1005+%2B+...+%2B+9999))

The same sum, 16,501,500, from all three environments.

Try *tinkering* to calculate another sum.

Sum all three digit numbers divisible by seven.

Results: GeoGebra: 70,336; Python: 70,336; and Wolfram Alpha: 70,336.

Seminar 3

Triangle with unknown angles

A triangle Δ*ABC* has three unknown angles called *a*, *b*, and *c* with the following relations. Angle *a* is 12° larger than angle *c*. Angle *b* is four times larger than angle *c*. Determine the size of the angles.

A graphical representation of a triangle ΔABC is shown in **Figure 9**.

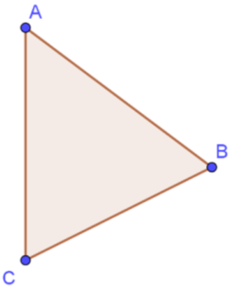


Figure 9. GeoGebra show us a triangle (https://www.geogebra.org/classic/gzwrdjug)

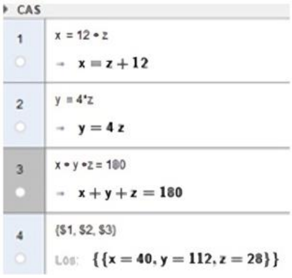
Here we may test if students know that *a*+*b*+*c*=180°, which they should add themselves.

Pseudo code

```
$a=12°+c$

$b=4×c$

$a+b+c=180°$
```

GeoGebra

Through CAS in GeoGebra, we receive result in **Figure 10**.



```
1  x = 12 • z
   → x = z + 12
2  y = 4*z
   → y = 4 z
3  x • y •z = 180
   → x + y + z = 180
4  {$1, $2, $3}
   Lös: {{x = 40, y = 112, z = 28}}
```

Figure 10. GeoGebra's CAS tool have no problem with three unknown variables

You find the GeoGebra file at: https://www.geogebra.org/classic/gzwrdjug.

Python

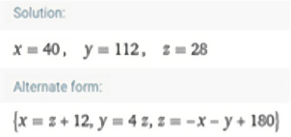
```
from sympy import*
```

```
var (“a b c”, integer=True, positive=True)

print (solve ((a-12-c, b-4×c, a+b+c-180), (a, b, c)))
```

See https://repl.it/@ThomasLinge/Unknown-angles for more details.

repl.it responds: {*a*: 40, *b*: 112, *c*: 28}

Wolfram Alpha



Solution:

$x = 40, \quad y = 112, \quad z = 28$

Alternate form:

$\{x = z + 12, y = 4\,z, z = -x - y + 180\}$

Figure 11. Wolfram Alpha has no problems with three unknown variables (http://www.wolframalpha.com/input/?i=%7Bx+%3D+12+%2B+z%7D,+%7By+%3D+4*z%7D,+%7Bx+%2B+y+%2B+z+%3D+180%7D)

Wolfram Alpha deals with unknown variables, an example of what programming is good at.

Tinkering: We now could construct a new triangle with other measurements and relations or perhaps extract the problem into a quadrangle.

Tinkering

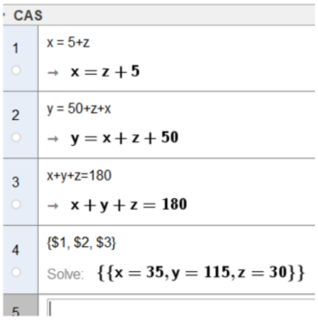
A triangle Δ*ABC* has three unknown angles *a*, *b*, and *c* with the relations: Angle *a* is 5° larger than angle *c*. Angle *b* is 50° larger than angle *c*+*a*. Determine the size of the angles.

Pseudo code

```
$a=5°+c$

$b=50+a+c$

$a+b+c=180°$
```

GeoGebra

We use the CAS tool in in GeoGebra.



CAS

1 x = 5+z
→ $x = z + 5$

2 y = 50+z+x
→ $y = x + z + 50$

3 x+y+z=180
→ $x + y + z = 180$

4 {$1, $2, $3}
Solve: $\{\{x = 35, y = 115, z = 30\}\}$

5

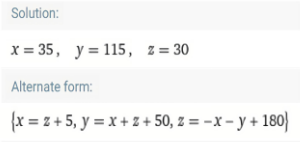
Figure 12. GeoGebra’s CAS tool solve with three unknown variables (https://www.geogebra.org/m/sg2jdjyz)

Python

```
from sympy import*

var (“a b c”, integer=True, positive=True)

print (solve ((a-5-c, b-a-c-50, a+b+c-180), (a, b, c)))
```

See https://repl.it/@ThomasLinge/Unknown-angles-II for more details. *repl.it* responds: {*a*: 35, *b*: 115, *c*: 30}

Wolfram Alpha



Solution:

$x = 35, \quad y = 115, \quad z = 30$

Alternate form:

$\{x = z + 5, y = x + z + 50, z = -x - y + 180\}$

Figure 13. Wolfram Alpha has no problems with three unknown variables (http://www.wolframalpha.com/input/?i=%7Bx+%3D+12+%2B+z%7D,+%7By+%3D+4*z%7D,+%7Bx+%2B+y+%2B+z+%3D+180%7D)

Wolfram Alpha deals with unknown variables, an example of what programming is good at.

THEORETICAL FRAMEWORK

Theory about technology as amplifier or reorganizer by Pea (1985, 1987), is a theory that fits well with programming. According to this theory, programming is not just a method for solving problems, but more of an integrated part of our social methods and our cultural and economic organization. When a technology as programming functions as an amplifier of existents structures it improve our mental structures or it reorganize the very same structures in our mind. Example of amplified structures are computers, and digital environments for communication. Example of technology that serves as reorganization are social media, artificial intelligence, and robots. Since all these artifacts are the result of programming, it is a natural theory.

RESULTS

I gave the students a short test after the course, following the guidelines by the university. It was a test with 10 questions: Summarize all three-digit numbers divisible by seven. The test was done in a large lecture hall, and I had my laptop with me, where I sat up front. The students had paper and pencil. No smartphones were allowed. When the students wanted to test code in any of GeoGebra, Python, and/or Wolfram Alpha they were allowed to come and do that on my laptop. Only one student at the time were allowed to come forward. No queueing was allowed and no talking between the students in the lecture hall. Altogether 19 of the students answered with a code for GeoGebra. About eight of them did not need to check the code.

Here are the arguments and the code from some of the students.

> The number 105 is divisible with seven and must be the first three-digit number that is. Divide 107 with 7=15. The largest three-digit number divisible with seven must be 994. Divide 994 with 7=142.
>
> The command is Sequence (7*k*, *k*, 15, 142).
>
> GeoGebra get the result: 70,336.

(https://www.geogebra.org/m/gpaanzpr)

Eight students answered with a code for Python: Only five of these students checked the code on my laptop.

```
sum=0

for number in range (105, 1,000, 7):

   sum+=number

print (“The sum is:”, sum)
```

See https://repl.it/@ThomasLinge/Sum-3-digit-numbers-divisible-by-7 for more details.

Python responds: The sum is: 70,336. Five of the students checked the code on my laptop.

No student selected Wolfram Alpha for their solution.

DISCUSSION & CONCLUSIONS

Is programming easy or hard? This question of course depends on who we are asking and what kind of programming we are discussing. Some humans probably think that programming is easy and intuitive while others consider programming to be challenging and hard. Our experiences and what kind of programming we are aiming at affects our vision of programming as easy or hard.

The more mathematics you have understood is probably helping you with the programming as well. I consider GeoGebra and Wolfram Alpha to be much easier to program than Python. That is of course depending on that the purpose with the programming in this manuscript is mathematical.

The programming seemed to be much easier for most of the students than I had imagined. Of course, some students disliked entering the mathematics into the environments of GeoGebra, Python, and Wolfram Alpha. The were probably considering mathematics to be something that should be done with paper and pencil and thus considering that letting GeoGebra, Python, and Wolfram Alpha solve the problem was cheating.

Funding: No funding source is reported for this study.

Ethical statement: The author stated that the study does not require any ethical approval. Informed consents were obtained from the participants.

Declaration of interest: No conflict of interest is declared by the author.

Data sharing statement: Data supporting the findings and conclusions are available upon request from the author.

REFERENCES

Lingefjärd, T. (2022). *Mathematics and statistics with GeoGebra, Python and Wolfram Alpha*. Studentlitteratur. ISBN: 978-91-44-15165-6.

Pea, R. D. (1985). Beyond amplification: Using the computer to reorganize mental functioning. *Educational Psychologist, 20*(4), 167-182. https://doi.org/10.1207/s15326985ep2004_2

Pea, R. D. (1987). Cognitive technologies in mathematics education. In A. H. Schoenfeld (Ed.), *Cognitive science and mathematics education* (pp. 89-122). Routledge.