Highest Common Factor

Teachers Teaching with Technology™
Professional Development from Texas Instruments

Teacher Notes and Answers

7 8 9 10 11 12

TI-Nspire | Coding | Student | 60 min

Introduction

Teacher Notes: The Highest Common Factor command is available on TI-Nspire directly[1], the purpose of the activity is to help students see that commands are backed by algorithms. Knowing that calculations produced by an electronic device are based on algorithms helps students understand their limitations. As questions become more sophisticated students need a greater understanding of the mathematics.

The activity encourages students to extend the algorithm so that it can be used to determine the highest common factor of three or more numbers. Students completing these extension components should consider putting their program in the public library so they can call it up from any document. Instructions on how to add content to the public library are available on the Texas Instruments Australia YouTube channel.

A video designed to support this activity is available on YouTube, students can use the QR code scanning capabilities of their mobile phone to watch the video or the link provided.

The Highest Common Factor (HCF), also known as Greatest Common Divisor (GCD) of two numbers is useful for many reasons. A common application relates to fractions, for example:

$$\frac{1}{12}+\frac{1}{18}=\frac{18}{216}+\frac{12}{216}=\frac{30}{216} \qquad \text{OR} \qquad \frac{1}{12}+\frac{1}{18}=\frac{3}{36}+\frac{2}{36}=\frac{5}{36}$$

The left-hand example prioritises a common denominator over highest common factor, the result is a fraction that still needs to be simplified. The right-hand example prioritises a highest common factor and then establishes the common denominator; the result is a fraction in its simplest form. This is just one example where a highest common factor is useful.

More than 2000 years ago, a mathematician by the name of Euclid created an algorithm that helps find the highest common factor of two numbers.

LINE #1: IF A = 0 THEN GCD(A,B) = B since GCD(0,B) = B

LINE #2: IF B = 0 THEN GCD(A,B) = A since GCD(A,0) = A

LINE #3: A = B x Q + R … where Q is the quotient and R is the remainder

LINE #4: GCD(B,R) = GCD(A,B), now find GCD(B,R)

https://bit.ly/EuclidAlgorithm

This algorithm will make more sense when some numbers are used for A and B. Suppose we want to find the highest common factor of (A) 1260 and (B) 385.

[1] Highest Common Factor is visible when the calculator setting is "English (UK)", otherwise Greatest Common Divisor is visible in the menu structure. In both cases the command is: GCD(#1,#2)

Author: P. Fox

As neither A = 0 or B = 0 we progress to LINE #3.

1260 = 385 x 3 + 105 [We can say that 105 is the remainder when 1260 is divided by 385]

According to LINE #4 of Euclid's algorithm: GCD(1260,385) = GCD(385,105)

We apply the algorithm again as 385 ≠ 0 and 105 ≠ 0 and proceed to LINE #3.

385 = 105 x 3 + 70 [We can say that 70 is the remainder when 385 is divided by 105]

According to LINE #4 of Euclid's algorithm: GCD(1260,385) = GCD(385,105) = GCD(105,70). As 105 ≠ 0 and 70 ≠ 0, then we return to LINE #3

105 = 70 x 1 + 35 [We can say that 35 is the remainder when 105 is divided by 70]

We are getting close! According to LINE #4 of Euclid's algorithm: GCD(1260,385) = … = GCD(70,35)

Applying the algorithm one more time, as 70 ≠ 0 and 35 ≠ 0, we proceed to LINE #3.

70 = 2 x 35 + 0. [This time the remainder is 0!]

Now we can apply LINE #1 or LINE #2 since we have GCD(35,0) = 35.

Our conclusion is that the Highest Common Factor or Greatest Common Divisor of 1286 and 385 is 35.

Question: 1.

Use Euclid's algorithm to identify the highest common factor of: 3850 and 3234.

Answer: 3850 ÷ 3234 = 1 + remainder 616
3234 ÷ 616 = 5 + remainder 154
616 ÷ 154 = 4 + remainder 0
Therefore the highest common factor of 3850 and 3234 is 154.

Teacher Notes: This problem was carefully selected to encourage students to estimate. Students should not need a calculator to determine that 3850 ÷ 3234 will divide only once, therefore the remainder can be calculated directly: 3850 – 3234.
In the next step students should be encouraged to estimate that 3234 ÷ 616 is approximately 5.

Creating the Program

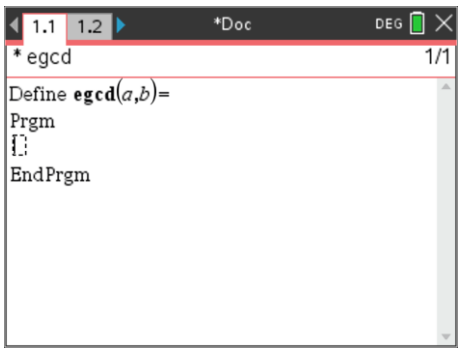
Instructions:

Start a new document; insert a new program.

Add Program Editor > New

Call the program: EGCD

Edit the program definition to include "a" and "b". (See opposite)



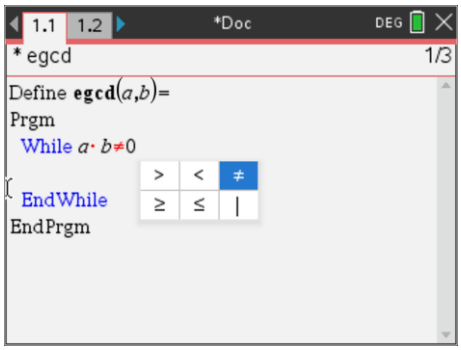
Author: P.Fox

Euclid's algorithm ceases when either a = 0 or b = 0, an easy way to check this is: a x b = 0. The null factor law states that "if the product of two numbers is zero, then one or both of the numbers must be zero."

The algorithm should continue to run while a x b ≠ 0.

Menu > Control > While … EndWhile

The "not equals" sign can be accessed from the inequality flyout menu.

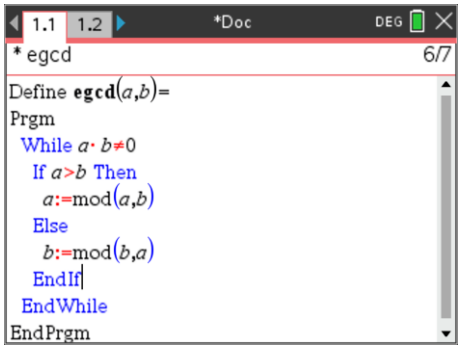


Modular arithmetic returns the remainder when a ÷ b (where a > b) so an If … Then … Else … statement can be used to process Line #3 of Euclid's algorithm.

Menu > Control > If...Then...Else... EndIf

The mod() command can be typed directly or accessed from the catalogue. Note carefully the respective orders for a and b.

That's the entire algorithm! The only thing remaining is to display the results. You can place: Disp a,b in the loop, between EndIf and EndWhile or outside the loop, between EndWhile and EndPrgrm.



Question: 2.

What is the difference in the output when the display command (Disp) is placed inside the loop compared with outside? Try it using 1914 and 7293.

Answer: If the display command is used in the loop each step towards the solution is displayed. If the display command is used outside the loop only the final result is displayed.

Teacher Notes: Students should be encouraged to include an "IF" statement and text to display the result in a more user-friendly format such as:

```
IF a = 0 Then
        Disp "Highest Common Factor", b
 Else
        Disp "Highest Common Factor", a
EndIf
```

Question: 3.

Test your program on some smaller numbers where you know the highest common factor. Record your test results here.

Teacher Notes: A very powerful lesson to share with students to help stimulate their 'testing' regime is the story of Knight Capital Group. On 1st August 2012 the company deployed a software update on their servers. The software contains algorithms that make rapid buy-sell decisions on the stock market. At 8:01am staff received email notifications about a potential error. At 9:00am the New York Stock Exchange opened, by 9:45am more than 4 million trades had been executed generating losses of more than USD$400,000,000!

Author: P.Fox

Answer: Answers will vary here, but this is a VERY important part of coding! Students should test prime numbers and number combinations that are mutually prime. Students should also know how to 'break' a routine that is stuck in an endless loop in the event they identify a number or number combination that does not work as expected. One of the most common faults when a 'while' loop is used is when the condition is never met? Some programmers use a second condition that tracks the number of loops allowing the program to exit in the event of an infinite loop.

Students should also be reverse engineering the problem so they know the answer prior to testing the results.

Examples: 3 x 5 x 17 x 11 = 2805 and 5 x 17 x 19 x 23 = 37145. We know the highest common factor will be 5 x 17 = 85.

Similarly: 3 x 5 x 17 = 255 and 2 x 7 x 19 = 266. So 255 and 266 is mutually prime, so the highest common factor will be equal to 1.

Question: 4.

The **Number** menu in the Calculator Application contains a command to determine the highest common factor of **two** numbers. Adjust your program to find the highest common factor of three numbers or a list of numbers. Example: EGCD(a,b,c) or EGCD({#1,#2 … #n})

Test and evaluate your program.

Answer: Answers will vary. A sample program for three numbers and an entire list have been provided here. Other solutions may also work. In the case of a list of numbers, it is not necessary to sort first but it can speed up the program, depending on the numbers in the list. For example in a list that consists of say: {3600, 3000, 10}, it is obvious that the highest common factor is 10, so starting with 3600 and 3000 doesn't make sense. This doesn't make much difference when three numbers are used, but if a list of numbers are used, it can make the program run time a lot longer. Improvements in computational speeds over recent years have relied more upon efficient coding than actual computer processing speed. If a sorting algorithm is quicker than the 'highest common factor' routine, then improved performance is gained by using the sort routine.

Author: P.Fox

```
*egcd
Define egcd(a,b,c)=
Prgm
© First HCF Algorithm
While a≠0 and b≠0
 If a>b Then
  a:=mod(a,b)
 Else
  b:=mod(b,a)
 EndIf
EndWhile
© Replace a or b with c
If a=0 Then
 a:=c
Else
 b:=c
EndIf
© Repeat the HCF algorithm
While a≠0 and b≠0
 If a>b Then
  a:=mod(a,b)
 Else
  b:=mod(b,a)
 EndIf
EndWhile
If a=0 Then
 Disp "Highest Common Factor: ",b
Else
 Disp "Highest Common Factor: ",a
EndIf
```

This is the basic algorithm by Euclid. Notice that it is repeated later in the program. This introduces the notion that a list of numbers could be used by nesting this algorithm in a loop.

When the algorithm finishes either a = 0 or b = 0.
If a = 0 then b is the highest common factor of a and b, so the algorithm needs to be performed on b and c, or simply make a = c so as to repeat the original algorithm.

```
egcd
Define egcd(data)=
Prgm
d:=dim(data)
SortA data
a:=data[1]
For i,1,d−1
 If a=0 Then
  a:=data[i]
 Else
  b:=data[i+1]
 EndIf
 While a≠0 and b≠0
 If a>b Then
  a:=mod(a,b)
 Else
  b:=mod(b,a)
 EndIf
EndWhile
EndFor

If a=0 Then
 Disp "Highest Common Factor: ",b
Else
 Disp "Highest Common Factor: ",a
EndIf
EndPrgm
```

The FOR loop is controlled by the number of elements in the list (data). The list is sorted into ascending order and the first value in the list is assigned to 'a', the first value for 'b' is assigned in the loop.

This is LINE #1 and LINE #2 of Euclid's algorithm. The IF statement is only testing if a = 0, it assumes that b=0, which would be true after the While loop, but the first time through it is 'tricked' into assigning b.

This is the familiar LINE #3 of Euclid's algorithm

Author: P.Fox