Application note 011a,

Demonstration of graphs, integration, derivatives and root finding on the C47.

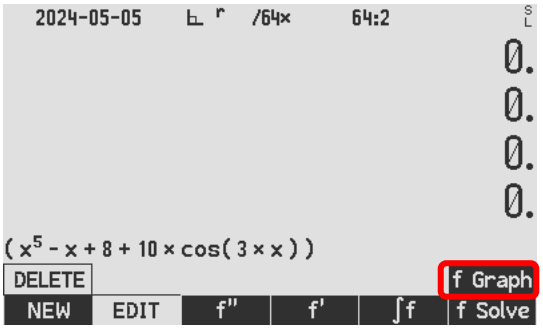
C47 supports two different user interface methods to use the solver and integrator, both discussed below. The methods are either the EQN menu or the ADV menu.

This Appnote, rev. "011a" is a re-write / update from "011", as the menu structure and UI commands significantly changed especially relating to graphs from the EQN menu. Additionally graphs for complex numbers were added.

Section 1: Graphing

Enter a formula in the EQN menu system:

With the EQN menu open:

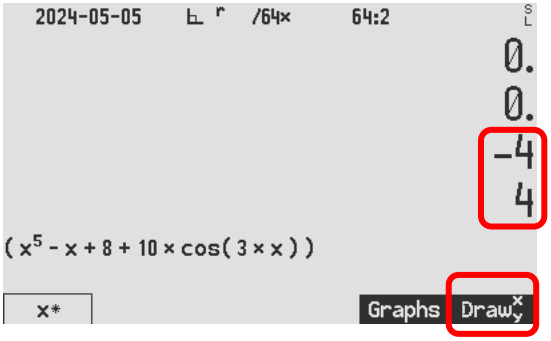


Go to: EQN, NEW add a new expression:

I entered this expression as it is fairly interesting, with complex roots and the opportunity to show the exhaustive method of root finding.

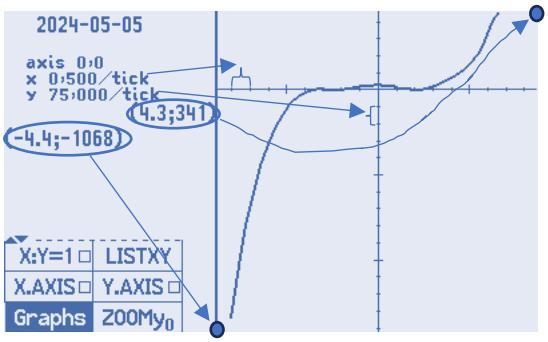
Enter the expression into the editor bearing in mind variables cannot be single letter capital letters. Abbreviations for known constants (α, π, φ, i, a, c, e, F, etc. from the constant list on 47calc.com) will represent their respective values.

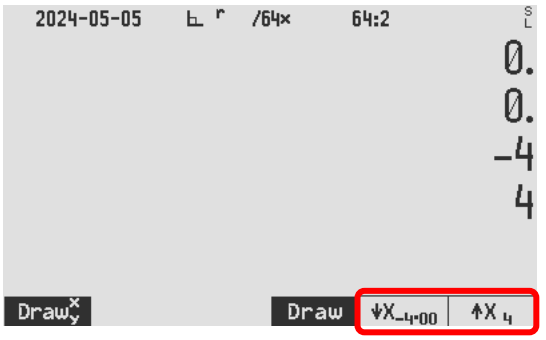
Using graphs in the EQN menu system:

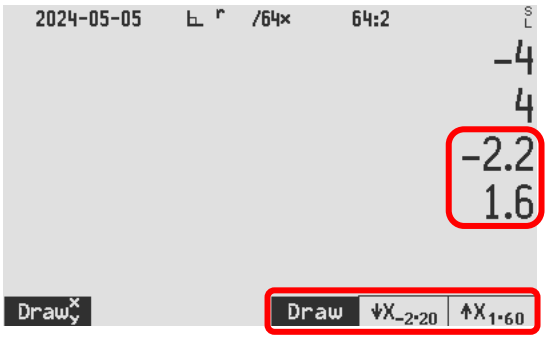


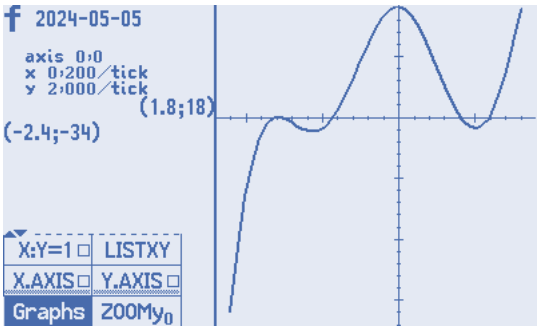
EQN, f Graph goes to variable selection

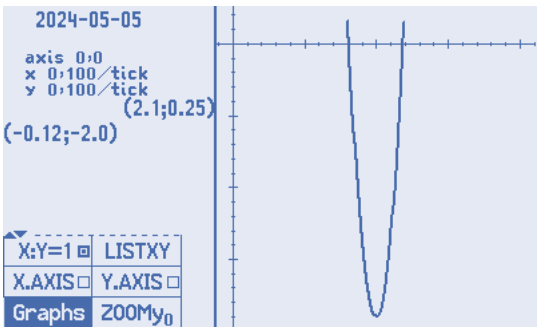
The simplest way to get a graph is: EQN, f Graph, select the graph variables (pre-selected for single variable x). Use Draw to plot the graph for x being from stack level Y to X. The example shows -4 to +4 on the stack, then press Draw.











Note the bottom left point on the cartesian plane (-4.4; -1068) and the top right hand point on the plane (4.3; 341). The resultant x-range of -4.4 to 4.3 is 105% to 110% of the range specified.

"Ticks" mean the distance/width of the 'minor' tick marks on the respective axis.

The graphic functions are located in the **Graphs** menu, which is accessible from every main menu in EQN. Note the prior x-range remains on the stack as entered, and also found its way into the ↓X and ↑X registers as -4 and 4 after every use of **Draw**. The purpose of **Draw** is to draw the graph from the lower limit ↓X to the upper limit ↑X.

inside the **ToolS** and **Tool∫** menus in the solver and ∫f menus respectively, also in the **Graphs** menus found in the **f'** and **f''** menus.

Example:

1. Enter -2.2 to 1.6 into the ↓X and ↑X registers.

2. Press **Draw**.

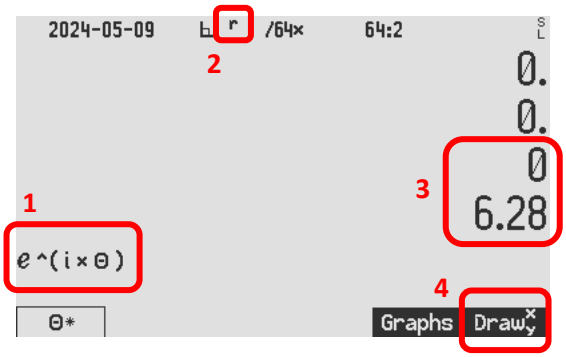
Note the Y-axis appears compressed with y-tick of size 2.0, whereas the x-ticks are 0.2 each! This is done as default to enable you to best see the shape of the curve and the roots.

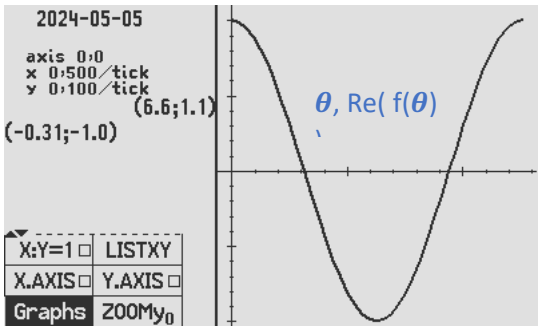
Forcing the X and Y axis the same scale, needs the selection of option **X:Y=1** *on the menu.*

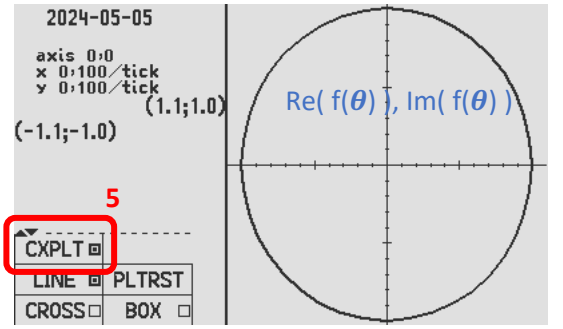
For equal scale autoscaling like "graph paper" spacing, we need to choose an X-range that will result in a much reduced Y-range.

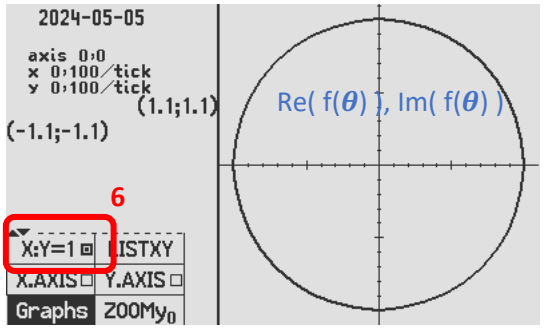
Use the menu item for **X:Y=1** and then **Graphs** returns to the graph selection screen to use ↓X and ↑X. Re-draw the curve for X:[0.8;1.2]. The C47 will keep the graph visible and centred on the Cartesian plane and increase the X or Y scale to keep both scales (tick size) the same.

Using graphs for complex numbers in the EQN menu system:









Go to: EQN, NEW to enter an expression which will results in complex number results when the expression is evaluated. A beautiful choice for illustration of this graph is Euler's: 1: $e^{i\theta}$.

C47: 2: Set to RAD, enter the expression as per the previous section of the Appnote and 3: plot for t from 0 to 6.28 using 4: Draw$_{y}^{x}$ or Draw.

The C47 graphing system will only draw the Real part of the expression on a Cartesian plane.

The C47 calculates complex results in rectangular form, maintaining the Real and Imag parts and not the size and angle. To no surprise, the graph left, of the Real part, is simply the cosine function, as stated per Euler:

$$e^{i\theta} = cos(\theta) + i.sin\ (\theta).$$

To plot the expression in the complex plane, simply select the option 5: CXPLT[o].

The resulting graph is not a polar plot per se. The complex plane drawing is of the Real and Imag values of $f(\theta) = e^{i\theta}$ for 40-80 regularly spaced values of t, in the specified θ range.

As a refinement to equalize the x and y scales and make the circle look properly dimensioned, select option 6: X:Y=1[o] from the menu.

A beautiful and easy exercise left to the user: Recall the demo formula provided in C47 Register 38. Use the X.SWAP function to enter the X register content into the expression editor and plot it as follows:

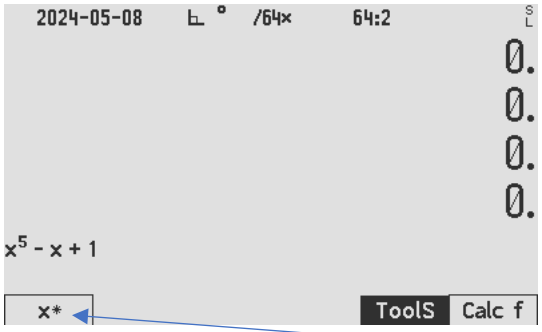
RCL 38 fMODE RAD fEQN NEW ↓ X.SWAP ENTER
f Graph 0 ENTER 6.28 Draw$_{y}^{x}$ ↑ CXPLT[]

Section 2: expression evaluation

Using Calc f to evaluate the current formula in the EQN menu system:

With the EQN menu open:

Add or select an expression, and press **f Solve** to access **[Calc f]** which evaluates the current expression using the current variable values.



Example: enter 5 into x, and press to evaluate:

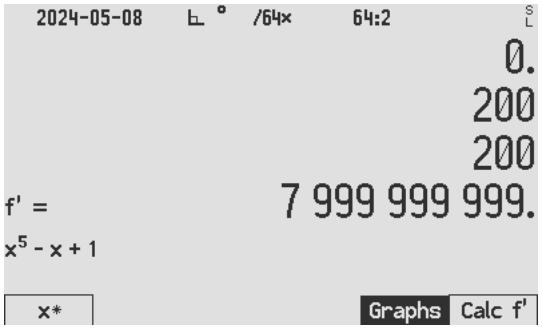
5 [x*] [Calc f] Resulting in 3121.0

Note the little * appearing, to indicate that x is the current variable.

Using 1st derivative and 2nd derivative of the current formula in the EQN menu system:

With the EQN menu open:

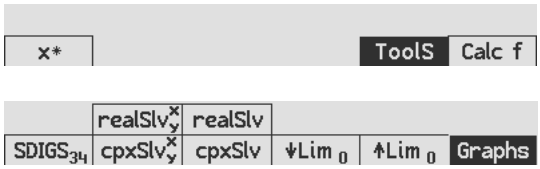
Add or select an expression, and press either **f'** or **f''** which evaluates the current expression's derivative using the value in X.



When opening the f' or f'' menu, if it is a simplistic single variable formula, then the X register will automatically be stored into the variable and the slope be calculated.

Store a new value for the variable **x*** and press **Calc f''** or **Calc f'** for the 2nd and 1st derivatives respectively, calculated using the current values of the variables.

Using solvers in the EQN menu system:

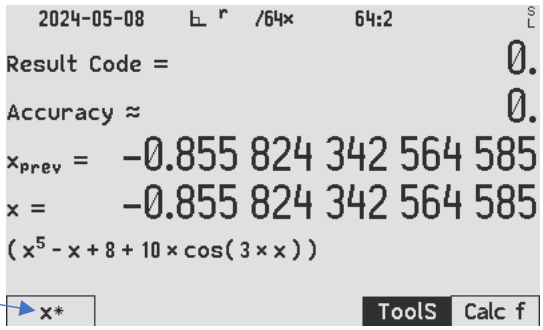


Press **EQN** **f Solve** to get to the solver menu

Press **guess1** **x** **guess2** **x*** **x*** to solve

Press **f Solve** **ToolS** to get to the ToolS menu

Real solver



Real root found using HP style real root finder

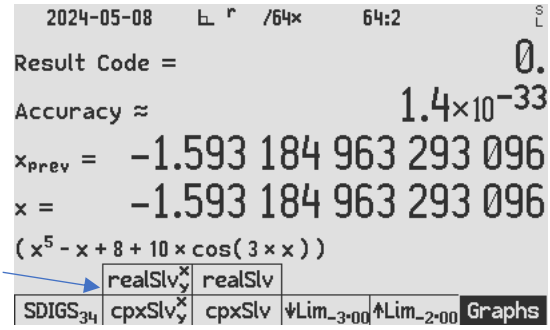
Find a root using the HP way of solving, pressing:

-2 **x** **-1** **x*** **x***

Multiple roots often are hard to find, as in this particular expression which lets many starting guesses gravitate to the same root!

A way to find different roots is to try another solver: The Complex Solver has a different algorithm which may find different roots.

Complex solver



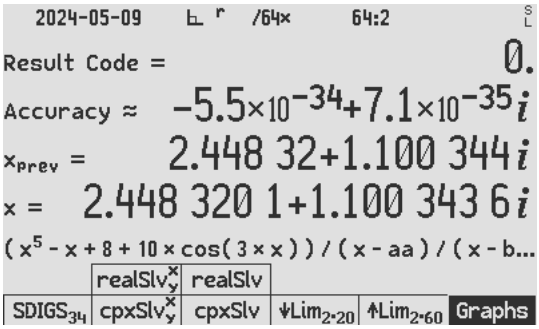
Real root using the complex root finder

Complex solver: **cpxSlv** or **cpxSlv**:

Type **-3 ENTER -2** **cpxSlv** to find a root at x = - 1.562. **cpxSlv** uses the YX stack input.

Once the limits are entered once using **cpxSlv**, the limits are updated on F5/F6 and then **cpxSlv** and **realSlv** can be used, using limits on F5/F6.

Exhaustive finding of all roots:



Complex root using the complex root finder

- Store the last root, example: `STO 'root'` and divide the expression by (x-root) found, i.e. for every root found, add to the expression "… / (x-root) …".

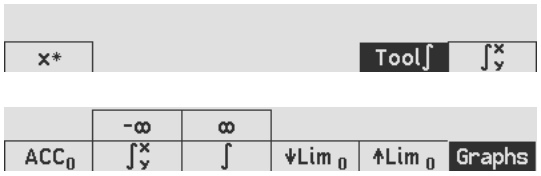
Tip: Use an actual complex number as starting guess to `cpxSlv_y` which will help to start in the complex domain, example: `1 ENTER i` `cpxSlv_y`.

Exhaustive finding of all roots example: Divide the expression by (x-root) for every root found. The following roots were found in the way described above, adding one factor in the denominator after each root was found, ending with:

```
"(x^5-x+8+10*cos(3*x))/(x-aa)/(x-bb)/(x-cc)/(x-dd)/(x-ee)/(x-ff)"

"aa"   0.80813371182864755275409047197092 79          "bb"   -0.85582434256458478788727792628095 08

"cc"   1.18528351613740675289967807504 3088          "dd"   -1.59318496329309631520192989833 0711

"ee"   -1.56215850315122964849446918944 0688

"ff" Complex root:   2.44832005227216378325214973553 0966 + i1.10034359703725393330321383476 2732

"gg" Complex root:   2.44832005227216378325214973553 0966 - i1.10034359703725393330321383476 2732
```

Get the numeric data from registers aa..gg: Once all the roots are stored, print all registers to a file on the flash drive using `I/O` `Allr`. To also include the formula in the list, first `EQN` `EDIT` the expression and use `X.SWAP` to swap the formula into the X register, then proceed printing all the registers.

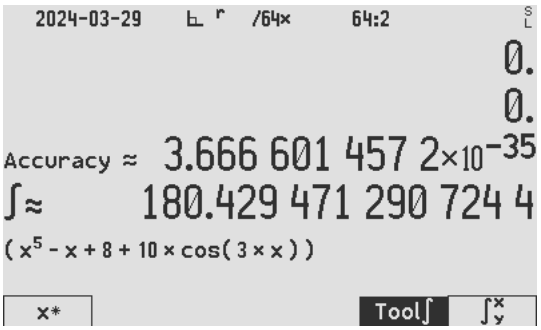
Using integration in the EQN menu system:



Press **EQN** **∫f** to get to the integration menu

Press **LowerLimit** **×** **HigherLimit** **x*** **∫xy**

Press **∫f** **Tool∫** to get to the int. tools menu

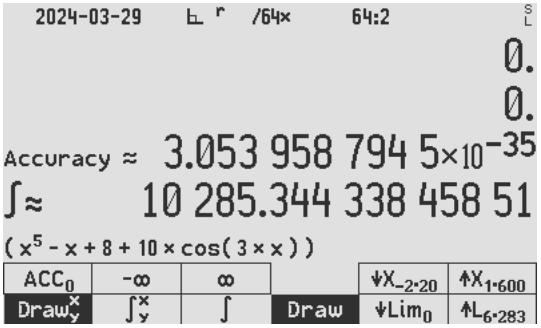


Standard integral found using HP style integrator

Find the numeric integral using the HP way of integrating, by typing:

0 **×** **π** **x*** **∫xy**

The **∫xy** function is meant for easy stack access to the interactive integrator.



Integral found using the **Lim** *buttons and* **∫**

Once **∫xy** is used once, the X and Y limits are populated into the **↓Lim** and **↑Lim** buttons for further usage of **∫**. Example:

Change the integral limits to 0 – 2**π** by typing: **Tool∫** **π** **2** ***** **↑Lim**. The new 6.283 value is showing in the **↑Lim** button. Press **∫** to use the limits.

Programmed root finding using RPN code and the ADV menu:

Typing instructions:

XEQ .. PRGM LBL 'WW' ENTER RAD STO K 5 y^x RCL K – 8 + RCL K 3 * COS 10 * + PRGM

Make a RPN program for the formula above, with the X register being the input. Function **WW** is created with X input and X output.

Typing instructions:

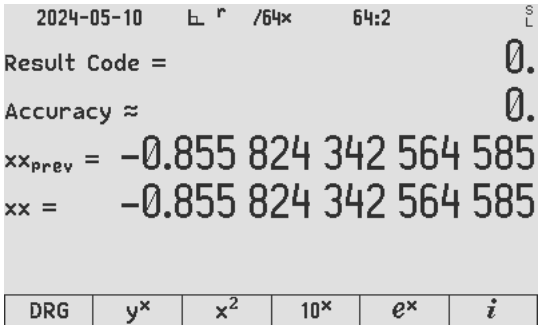
XEQ .. PRGM LBL 'WW_{MVAR}' MVAR 'xx' RCL 'xx' XEQ 'WW' PRGM

Make a MVAR program to facilitate the new formula: Add variables and recall the formula variable to feed to the formula as X. The MVAR code does root finding and integration.

Typing instructions:

XEQ .. PRGM LBL 'WW_{ROOTS}' PGMSLV 'WW_{MVAR}' SOLVE 'xx' PRGM

Make a real root finder program, specifying the MVAR formula to be used, as well as the variable to be used. **SOLVE** requires **PGMSLV** to name the function to be used, and **SOLVE** requires the starting values in Y & X.

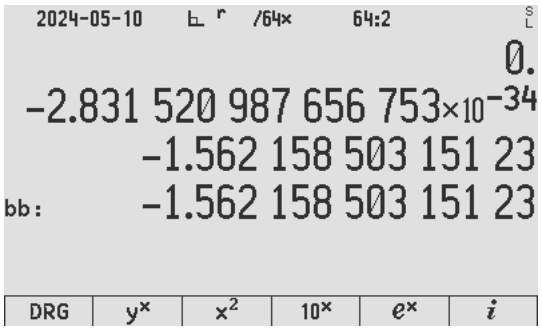


*Program **WW**$_{ROOTS}$, resulting in a root in X*

Run the search: **-2 ENTER -1 XEQ PROG 'WW**$_{ROOTS}$**'**

1st root -0.855 found. Store in 'aa'.

Change the RPN program **WW** to divide by (K - aa), in the way explained under 'Exhaustive Root Search' and continue to find real roots as follows:
… K 3 * COS 10 * + RCL K RCL 'aa' - / PRGM



*Program modified **WW**$_{ROOTS}$, resulting in a root in X*

Run the search: **-2 ENTER -1 XEQ PROG 'WW**$_{ROOTS}$**'**

2nd root -1.562 found. Store in 'bb'.

Continue the process to divide with (K – bb) to find more real roots.

…etc.

```
C47 Program file export: Export format version 2, C47
program version 1.
0000: { Prgm #49: 39 bytes / 17 steps }

0001:  LBL 'WW'
0002:    RAD
0003:    STO K
0004:    5
0005:    Y^X
0006:    RCL K
0007:    -
0008:    8
0009:    +
0010:    RCL K
0011:    3
0012:    *
0013:    COS
0014:    10
0015:    *
0016:    +
0017:  END
```

```
C47 Program file export: Export format version 2, C47
program version 1.
0000: { Prgm #50: 31 bytes / 5 steps }

0001:  LBL 'WWMVAR'
0002:    MVAR 'xx'
0003:    RCL 'xx'
0004:  XEQ 'WW'
0005:  END
```

```
C47 Program file export: Export format version 2, C47
program version 1.
0000: { Prgm #51: 37 bytes / 4 steps }

0001:  LBL 'WWROOTS'
0002:    PGMSLV 'WWMVAR'
0003:    SOLVE 'xx'
0004:  .END.
```

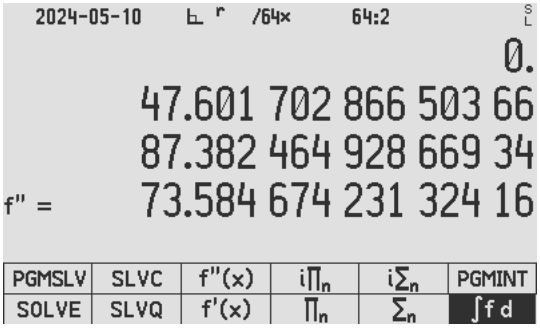
Programmed derivatives using RPN code and the ADV menu:

Typing instructions:

XEQ .. PRGM LBL 'WW$_{SLOPE}$' STO I XEQ PROG 'WW' RCL I ADV f'(x) PROG 'WW' RCL I ADV f''(x) PROG 'WW' STO K RCL J

To determine the slope (f') using RPN, and the slope's slope (f'') using RPN:

Create a small program WW$_{SLOPE}$ to sequence f' and f''. Note the stack is left unchanged after f' and f'' are pushed onto the stack.



XEQ the program 2 WW$_{SLOPE}$*, resulting in f''(2) on X and f'(2) on Y.*

```
C47 Program file export: Export format version 2,
C47 program version 1.
0000: { Prgm #45: 35 bytes / 8 steps }

0001:  LBL 'WWSLOPE'
0002:    STO I
0003:  XEQ 'WW'
0004:    RCL I
0005:    f'(x) 'WW'
0006:    RCL I
0007:    f''(x) 'WW'
0008:  .END.
```

Programmed integration using RPN code and the ADV menu:

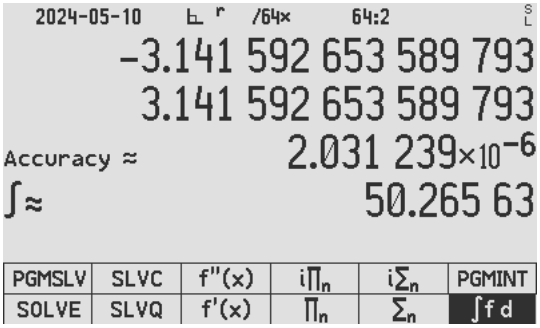
Now integrate the expression WW from - **π** to **π**:

XEQ .. PRGM LBL ‘WW_{AREA}‘ PGMINT ‘WW_{MVAR}’

fi 1E-6 ACC **π** CHS ↓Lim **π** DROPx Lim↑ ∫ ‘xx’

To determine the numeric integral using ADV menu, create a program WW_{AREA} sequencing it:

We need to first tell the C47 which RPN MVAR program (WW_{MVAR}) is to be used for integration, then we start the integration by specifying the integral variable.

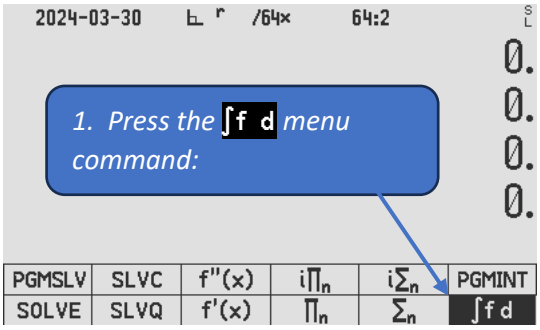


XEQ the program WW_{AREA}, resulting in the integral result from -π to π

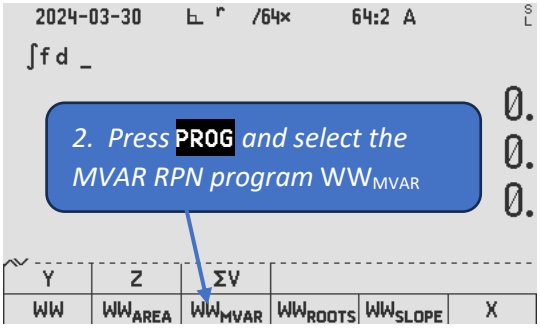
```
C47 Program file export: Export format version 2,
C47 program version 1.
0000: { Prgm #47: 68 bytes / 11 steps }

0001:  LBL 'WWAREA'
0002:    PGMINT 'WWMVAR'
0003:    1.*10^-6
0004:    STO 'ACC'
0005:    PI
0006:    CHS
0007:    STO 'vLim'
0008:    PI
0009:    STO '^Lim'
0010:    integralf d 'xx'
0011:  .END.
```

Manual interactive integration using RPN code and the ADV menu:

To determine the numeric integral manually (i.e. without an integration program) using **ADV** menu, use the **∫f d** command in **ADV**:



1. Press the **∫f d** *menu command, will open:*

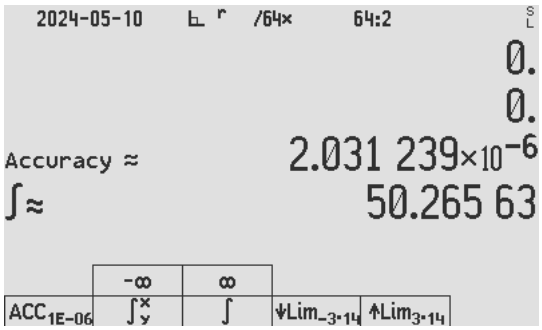


2. Select the MVAR program WW$_{MVAR}$

```
0001: LBL 'WWMVAR'
0002:   MVAR 'xx'
0003:   RCL 'xx'
0004: XEQ 'WW'
0005: END
```

RPN

3. It will show the variables; select the integral variable, press **twice** on this variable (xx), which will open the integration menu in step 4:



4. Enter the integration limits, either in Y & X, the press **∫xy** or using the **↓Lim / ↑Lim** fields

5. Then press **∫**.

[Author Jaco Mostert]