


Functions	Syntax	Uses
1. Asymptotes (Asymp) ⚠	<code>asympt(function, variable)</code>	Determines the vertical and horizontal asymptotes of a function.
2. Composite Function Check (ccheck)	<code>ccheck(Outer function, inner function)</code>	Determines if a composite exists, and if not, determines the maximal domain for which a composite exists.
3. Discriminant (Discrim)	<code>discrim(function, variable, 15)</code>	Calculates the discriminant of an inputted quadratic expression.
4. Domain and Range (Domrang) ⚠	<code>domrang(function, variable)</code>	Determines the domain and range of a function. Note: Due to CAS approximation bounds may not be entirely accurate.
5. Intercepts (Intercepts)	<code>intercepts(function, variable)</code>	Finding the x and y intercepts of a function.
6. Intersects (Intersects)	<code>intersects(function1, function2, variable)</code>	Determines the points of intersection of two functions across their maximal domains.
7. Intersects with domain (intersectsd)	<code>intersectsd(function1, function2, variable, lower, upper)</code>	Determines the points of intersection between two functions in a restricted domain.
8. Inverse Function (inverse)	<code>inverse(function, variable, x in domain of f)</code>	Determines the inverse of a given function.
9. Inverse Intersections (invints) ⚠	<code>inverse(function, number of intersections with inverse)</code>	Determines the values of a parameter, k , required for a function and its inverse to have a specified number of intersections. Works with: logs, exponentials, sqrt, parabolas, cubics, hyperbolas.
10. Angle between two lines (lineang)	<code>lineang(Line 1, Line 2, Variable)</code>	Determines the angle between two lines, and assumes your CAS is in radians mode not degrees mode.
11. Unique, None, Infinite Solution (Linesolve)	<code>linesolve(Equation1, Equation2)</code>	Determines when two equations will have an unique, none or infinitely many solutions.
12. Property Check (pcheck)	<code>pcheck(function, variable, LHS, RHS)</code>	Determines which function satisfies a specific property. Note: You must define the function outside of the program.
13. Point Information (pointinfo)	<code>pointinfo(x_1, y_1, x_2, y_2)</code>	Determines the gradient, perpendicular gradient, line, x and y intercepts of a line, midpoint, distance.
14. Polynomial Fit (polyfit)	<code>polyfit($\{x_1, y_1, \dots\}$)</code>	Determines a polynomial equation which passes through all the inputted pairs of points.
15. Transformations (transform)	<code>transform(function, {transformations})</code>	Determines the transformed function after applying certain transformations. Note: The transformations do not use real math, for example, $2y$ corresponds to a dilation by a factor of 2 from the y axis in the program.
16. Transformation Solve (transolve) ⚠	<code>trans_solve(Original, Image, Variable)</code>	Determines the set of transformations which take the original function to the image.
17. Bisection ⚠	<code>bisec(function, variable, lower, upper, iterations)</code>	Applies the bisection method on the function for a specified number of iterations.

Calculus	Syntax	Uses
1. Average Rate of Change (avgroc)	$\text{avgroc}(\text{function}, \text{variable}, \text{lower}, \text{upper})$	Determines the average rate of change of a function.
2. Average Value (avgval)	$\text{avgval}(\text{function}, \text{variable}, \text{lower}, \text{upper})$	Calculates the average value of a function.
3. Bound Area (boundarea)	$\text{boundarea}(\text{function1}, \text{function2}, \text{variable})$	Determines the area bound by two graphs (if any) across their maximal domains.
4. Bound Area with domain (boundaread)	$\text{boundaread}(\text{function1}, \text{function2}, \text{variable}, \text{lower}, \text{upper})$	Determines the bound area between two functions in a restricted domain.
5. Integral Guess (intguess)	<p>One integral given, find transformed integral.</p> <p>$\text{intguess}(\{\text{lower1}, \text{upper1}, \text{value1}\}, \{\text{transformations}\}, \{\text{lower2}, \text{upper2}\})$</p> <p>Two integrals given, find another integral of untransformed function.</p> <p>$\text{intguess}(\{\text{lower1}, \text{upper1}, \text{value1}\}, \{\text{lower2}, \text{upper2}, \text{value2}\}, \{\text{lower3}, \text{upper3}\})$</p> <p>Case 3: Two integrals given, then find another integral of transformed function.</p> <p>$\text{intguess}(\{\text{lower1}, \text{upper1}, \text{value1}, \text{lower2}, \text{upper2}, \text{value2}\}, \{\text{transformations}\}, \{\text{lower3}, \text{upper3}\})$</p>	Determines the answer for the integration multiple choice questions.
6. Newton's Method (newtons)	$\text{newtons}(\text{function}, \text{variable}, x_0, \text{iterations})$	Estimates the root of a function using newton's method.
7. Number of Roots (nroot) ⚠	$\text{nroot}(\text{polynomial}, \text{variable}, \text{parameter}, \text{no. roots})$	Determines the value(s) of the parameter for which a polynomial will have n roots.
8. Number of Stationary Points (nstp) ⚠	$\text{nstp}(\text{polynomial}, \text{variable}, \text{parameter}, \text{no. stp})$	Determines the value(s) of the parameter for which a polynomial will have n stationary points.
9. Number of Points of Inflection (nstp) ⚠	$\text{npoi}(\text{polynomial}, \text{variable}, \text{parameter}, \text{no. stp})$	Determines the value(s) of the parameter for which a polynomial will have n points of inflection.
10. Points of Inflection (pois) ⚠	$\text{pois}(\text{function}, \text{variable})$	Determines the points of inflection of a function.
11. Sign Table (signtab)	$\text{signtab}(\text{function}, \text{variable})$	Uses a sign table to determine the nature of the stationary points of a function.
12. Stationary Points (stps)	$\text{stps}(\text{function}, \text{variable})$	Determines the stationary points of a function.
13. Tangent Solve (tangsolve)	$\text{tangsolve}(\text{function}, \text{variable}, x_0, y_0)$	Determines the equation of the tangents of the function which pass through the specified point.
14. Trapezoid Approximation (trapapprox)	$\text{trapapprox}(\text{function}, \text{variable}, \text{lower}, \text{upper}, \text{number of trapezia})$	Approximates an integral using the trapezoidal rule.

Continuous Probability	Syntax	Uses
1. Continuous Conditional Probability (ccondpr)	Probability Density Function: ccondpr(<i>Probability Density Function, Lower Bound, Upper Bound</i>) Normal Distribution: ccondpr(" ", <i>Mean, Standard Deviation, Condition 1, Condition 2</i>)	Determines conditional probability for a continuous distribution.
2. Confidence Interval (confint)	confint(<i>Sample Size, \hat{P}, . confidence</i>)	Determines a confidence interval as well as the z-score, margin of error and standard deviation.
3. Confidence Interval Solve (confintsolve)	confintsolve(<i>Lower Bound, Upper Bound, Sample Size or Sample Standard Deviation or . Confidence</i>)	Determines the sample size, standard deviation or percentage confidence depending on the provided data. Note: This program assumes no confidence levels less than 50% will even be used.
4. Continuous Distribution Information (continfo)	continfo(<i>function, variable, lower, upper</i>)	Determines the expected value, mean, variance, standard deviation of a continuous probability distribution.
5. Inverse Normal (invnormvals)	invnormvals(<i>mean, standard deviation, probability</i>)	Determines the left, right and centre possibilities for probability of a distribution.
6. Normal Solve (normsolve)	Case 1: Both lower and upper given. normsolve(<i>Lower, Probability of Lower, Upper, Probability of Upper</i>) Case 2: Lower and μ given. normsolve(<i>Lower, Probability of Lower, μ, " "</i>) Case 3: Lower and σ given. normsolve(<i>Lower, Probability of Lower, " ", σ</i>) Case 4: Upper and μ given. normsolve(<i>μ, " ", Upper, Probability of Upper</i>) Case 5: Upper and σ given. normsolve(<i>" ", σ, Upper, Probability of Upper</i>)	Determines the mean and standard deviation for lower and upper type questions. Note: If required, convert into this format using complement.

Discrete Probability	Syntax	Uses
1. Binomial Solve	<code>binomsolve(outcome, probability of success, threshold value)</code>	Determines the number of trials required to achieve a certain probability.
2. Discrete Conditional Probability	<p>Binomial: <code>dcondpr(number of trials, probability of success, condition 1, condition 2)</code></p> <p>Probability Table: <code>dcondpr({List containing outcomes}, {List containing probabilities}, condition 1, condition 2)</code></p> <p>Probability Mass Function: <code>dcondpr({List containing outcomes}, PMF, condition 1, condition 2)</code></p>	Determines conditional probability for a discrete distribution.
3. Binomial Distribution Information	<code>binominfo(Sample Size, Probability of Success)</code>	Determines the expected value, variance, standard deviation, sample expected value, and sample standard deviation for a binomial distribution.
4. Hypergeometric Cumulative Probability Function	<code>hypergeocdf(sample size, population size, number of successful items, lower bound, upper bound)</code>	Determines the probability of selecting items without replacement, but over an interval of outcomes.
5. Hypergeometric Probability Density Function	<code>hypergeopdf(sample size, population size, number of successful items, outcome)</code>	Determines the probability of selecting items without replacement, but for specific outcomes.
6. Inverse Binomial	<code>invbinomial(number of trials, probability of success, known probability value)</code>	<p>Determines the outcome required to achieve the probability.</p> <p>Note: Will not work if probability is too accurate. In this case just decrease accuracy by decreasing number of decimal places.</p>
7. Probability Table	<code>prtable({outcomes}, {probabilities})</code>	Determines the mean, variance, standard deviation of a probability table.
8. Sample Distribution Binomial	<code>samplebinom(Sample Size, Probability of Success)</code>	Determines the distribution for the sample proportion of a binomially distributed sample.
9. Sample Binomial Probability	<code>samplebinompr(Sample Size, Probability of Success, Lower, Upper)</code>	Determines the probability for the sample proportion for a binomially distributed sample.
10. Sample Distribution Hypergeometric	<code>samplehypergeo(Sample Size, Population Size, Number Successful)</code>	Determines the distribution for the sample proportion of a hypergeometrically distributed sample.
11. Sample Hypergeometric probability	<code>samplehyppr(Sample Size, Population Size, Number Successful, Lower, Upper)</code>	Determines the probability for the sample proportion for a hypergeometrically distributed sample.

Miscellaneous	Syntax	Uses
1. Column Augment	<code>ca(Ans, {variables})</code>	If there is a long list of x and y values, then this will convert it into an easy-to-read matrix form.
2. Domain Solve	<code>dsolve(Equation, Variable, Lower Bound, Upper Bound)</code>	Determines the solutions to an equation in a restricted domain.
3. Graph Information	Maximal Domain: <code>graphinfo(function, variable, " ", don't_care)</code> Restricted Domain: <code>graphinfo(function, variable, lower, upper)</code>	Determines axes intercepts, stationary points, points of inflection, and endpoints of a function.
4. Trigonometric Solve	<code>trigsolve(Equation or inequality, variable, lower, upper)</code>	Determines the exact solutions to trigonometric equations and inequalities which the CAS cannot solve properly. For example, $\sin(x) = \cos(2x)$
5. Trigonometry Information 	<code>triginfo(circular function, variable, lower, upper)</code>	Determines the amplitude, period, middle value, endpoints, x-intercepts, y-intercepts, asymptotes, and turning points of a circular function .