# Graphically Assisted Programming

A programming language developed for beginners

## What is GAP

Graphically Assisted Programming or **GAP** is a graphically based interactive environment where non-technical and semi-technical end users can learn and practice fundamental programming concepts.

About GAP

## About GAP

GAP was developed by a team of students at National University as their final project for a Bachelor of Science in Computer Science degree (BSCS).

## Project Members:

- **Kiet Diep**
- **Dave Elliott** **(Project Manager/Instructor)**
- **Cuong Pham**
- **Fara Saheb** **(Project Leader)**
- **Steve Schow**
- **Brian Zelenak**

# Contents

## File Menu Options

| | |
|---|---|
| New | Clears screen for creating a program |
| Open | Loads a previously saved program |
| Save | Writes current program to disk |
| SaveAs | Writes program to disk with a new name |
| Exit | Closes the GAP application |

**New Clears screen for creating a program**

The New command clears the GAP canvas and initializes all GAP editor memory for a new GAP program. It also allows the user to either open another GAP program or exit the GAP application.

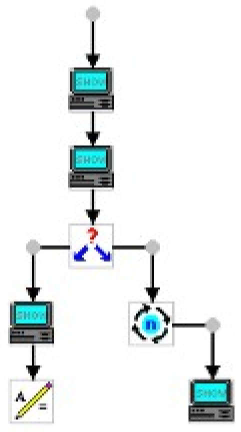
How to use New

## How to use New

- click on the New icon or move the mouse pointer to the File menu, and click the left button. This reveals the menu.
- Move the pointer to the New option, and click the left button again.

There is now a clear canvas with a Begin Marker to create a new program.

**Open** **Loads a previously saved program**



The **Open** command loads an existing GAP program file from disk for viewing, editing or execution. After selection, it will bring up a browse dialog box that will allow the user to search the computers disks for the desired program file to open.

This is what a opened GAP program looks like:

How to use Open

## How to use Open

- click on the Open icon or move the mouse pointer to the File menu, and click the left button. This reveals the menu.
- Move the pointer to the Open option, and click the left button again.
- When you select this option a dialog box will appear. You can select a file from a dialog box in two ways:

1. Type in a filename where the pointer is.

2. With the mouse, move the pointer to the file you want to open and click the left button to select it.

## **Save** Writes current program to disk

The Save command replaces the currently opened file with any changes that have occurred since the last save operation. If the GAP program has not been saved before, the user is prompted for a filename with the Save File dialog box. If the program has a filename, it will save the opened program and overwrite the file on disk with the same filename.

How to use Save

## How to use Save

- click on the Save icon or move the mouse pointer to the File menu, and click the left button. This reveals the menu.
- Move the pointer to the Save option, and click the left button again.
- Your current program is saved to the same directory.

## SaveAs    Writes program to disk with a new name

The SaveAs command saves the currently opened GAP program to a new filename. The user can specify a new filename and directory path in the SaveAs dialog box.

How to use SaveAs

## How to use SaveAs

- Move the mouse pointer to the File menu, and click the left button. This reveals the menu.
- Move the pointer to the SaveAs option and click the left button again.
- When you select this option a dialog box will appear. You can save your program under a different name or directory.
- Type in a the new filename where the pointer is.
- With the mouse, move the pointer to the directory you want and click the left button to select it.

**Exit  Closes the GAP application**

The **Exit** command leaves the GAP editor and returns the user to their previous environment.

How to use Exit

**How to use Exit**

- Move the mouse until the pointer on the screen is on the File menu, and click the left button. This reveals the menu.
- Move the pointer to the **Exit** option, and click the left button again.

## Edit Menu Options

| | |
|---|---|
| Undo | Reverses previous operation |

**Undo** **Reverses previous operation**

The **Undo** command reverses the last edit operation performed by the GAP user. If the user accidentally **Cut**s or deletes objects or text in dialog boxes, the action can be undone by invoking the **Undo** command.

How to use Undo

**How to use Undo**

- Move the mouse over the text or object you want to undo**.** This will highlight that area.
- Move the mouse until the pointer on the screen is on the Edit menu, and click the left button. This reveals the menu.
- Move the pointer to the **Undo** option, and click the left button again.

## Action Menu Options

| | |
|---|---|
| Run | Execute current program |
| Show Variables | Displays current status of variables |
| Panic Button | Stops executing program |

**Run** **Execute current program**

The Run option pops up an execution window to display output and executes the current GAP program. The program will display its output to the execution window.

The values of variables can be seen in the show variables window by selecting this icon and will be updated as the program progresses. The program execution may be stopped at any time by invoking the Panic Button .

How to use Run

## How to use Run

- Point the mouse arrow on the Run icon and use the left click button.
- Or move the mouse until the pointer on the screen is on the action menu, and click the left button. This reveals the menu.
- Move the pointer to the Run option and click the left button again. This will execute your GAP program.

## Show Variables     Displays current status of variables

The Show Variables option displays the current status of variables. A scrolling window is popped up which contains a list of all program variables and their current values. If the variables have the initial values, then those values will be indicated. As a GAP program executes, the values will be updated to reflect their current values during the program.

<u>How to use Show Variable</u>

## How to use Show Variable

- Point the mouse arrow on the Show Variable icon and use the left click button.
- Or move the mouse until the pointer on the screen is on the action menu, and click the left button.
- Move the pointer to the Show variable option and click the left button again.

**Panic Button** **Stops executing program**

The Panic option is like a halt. During program execution, the user may attempt to issue an emergency halt by using the keys. There are menus also, but keys will be more reliable. The current execution will be halted. This is invaluable for when a user has written a program with an endless loop.

How to use Panic Button

**How to use Panic Button**

- Point the mouse arrow on the Panic icon and use the left click button.
- Or move the mouse until the pointer on the screen is on the action menu, and click the left button.
- Move the pointer to the Panic halt option and click the left button again.

## Commands

| | |
|---|---|
| Assign | Equates a value or expression to a variable |
| Show | User defined output to screen |
| Read | Gets input from the user |
| Conditional Loop | Loops until condition is false |
| Iterative Loop | Loops a defined number of times |
| Alternate Path | Changes program flow |
| Trace | Animated execution of program |

## Assign — Equates a value or expression to a variable

By selecting the Assign command, the user will add a statement to the GAP program which assigns a value or expression to a variable.

Some examples of valid expressions are:
123
123 + X
"Hello World"
A valid expression may contain literal text or numbers, variable references and the operators:
+ - / * ^ %
to represent the plus, minus, divide, multiply, raise and mod operators.
Literal text and variables which contain text will be considered to be zero when used with one of the operators listed above.

How to invoke Assign

## How to invoke Assign

- Use the mouse to click on the assignment icon which is located on the vertical toolbar or use the Action menu.
- Fill out the assignment command dialog box by providing a variable NAME and VALUE.
- The value may be a number, <u>Literal text</u> or an <u>expression</u>.
- If an expression is desired, the expression radio button should be toggled on. The literal option will be toggled by default.
- Press OK button when done.

An object will be painted on the canvas in the appropriate place.

If there is a syntax error in an expression, an error message will be displayed and the user will be popped back to the assign command dialog box to correct the error before continuing.

## Show User defined output to screen

The Show command provides a way to display output from a GAP program. The entire show statement will be displayed on only one line of output. A new line will be appended after the last item. The show command can be literal or variables. Up to 5 variables and litterals can be displayed by the GAP program within one show statement.

How to invoke Show

## How to invoke Show

- Use the mouse to click on the show command icon in the vertical toolbar or use the pull down Action menu.
- Fill out the show command dialog box
- Click the OK button.

An object will be painted on the canvas in the appropriate place.

If the user attempts to specify an undefined variable as a show argument, then an alert dialog box will be displayed. The user will be popped back to the Show command dialog box to correct the error before continuing.

## Read Gets input from the user

The Read command prompts the user at the execution time for values which will be assigned to variables and used by the program. There will be a prompt message which is provided by the GAP programmer in the Read command. The Read command will accept lliteral text or numbers from the user at runtime.

How to invoke Read

## How to invoke Read

- Use the mouse to click on the Read button 📖 located on the left tool bar.
- Fill out the Read command dialog.
- Click on the OK button.

The Read object will automatically be placed on the diagram.

**Conditional Loop   Loops until condition is false**

A conditional loop is designed to run indefinitely as long as some condition remains true. The conditional loop needs only to test a condition that will eventually change. If the condition never changes the program never stops. Any numbers of commands may be executed within each loop iteration.

A condition will be represented by a comparison of two expressions. A valid expression may contain literal text or numbers, variable references and the operators to represent plus, minus, divide, multiply, raise or mod. If literal text or variables which contain literal text are used together with the operators, then they will be considered zero.
Two expressions may be compared with the following comparison operators:
=      !=      <      >
Examples of some valid comparisons are:
x   !=   5     Is x not equal to 5
x + y   >   34       Is x plus y greater than 34

N   =   "Hello World"      Does N match the literal text "Hello World"

How to invoke Conditional Loop

## How to invoke Conditional Loop

- Use the mouse to click on the conditional loop button on the vertical tool bar.
- Fill out the conditional loop command dialog with loop conditions
- Click the OK button

The command object will be created on the diagram.

Syntax errors in the conditional expressions may cause errors. These errors will be detected before the object is ever drawn on the screen. An alert dialog box will be displayed and the user will have to correct or cancel.

**Iterative Loop** **Loops a defined number of times**

An Iterative Loop repeats a group of program statements as long as a specified condition is true. Generally, you use it to specify a fixed number of repetitions. Any number of commands may be contained within the loop. The loop will be displayed left to right on the diagram. The user has to specify the number of iterations.

How to invoke Iterative Loop

## How to invoke Iterative Loop

- Use the mouse to click on the iterative loop button on the vertical toolbar.
- Fill out the iterative Loop command dialog .
- Click on the OK button.

If the user attempts to enter a non-integer value for the number of iterations, an alert dialog box will be displayed before the object is even placed on the diagram.   The user must then correct or cancel.

## **Alternate Path** **Changes program flow**

The Alternate Path command is analogous to an IF..THEN statement. It provides a choice of program flow depending on the condition of an expression. If the condition is met (true), it will execute the preceding command(s) or another path is executed. If the condition is not met it will skip the preceding commands and execute the nextobject on the main program list. The diagram will show both paths.

Although GAP is meant as a graphical programming tool, this step will require some textual prowess. A condition will be represented by a comparison of two expressions. A valid expression may contain literal text or numbers, variable references and the operators
+ - / * ^ %
to represent plus, minus, divide, multiply, raise and mod operators. If literal text or variables which contain literal text are used together with the operators listed above, then they will be considered to be zero.

Two expressions can be compared with the following comparison operators:

= != < >

Examples of some valid comparisons are:

| | |
|---|---|
| X != 5 | Is X not equal to 5 |
| X + Y > 34 | Is X plus Y greater than 34 |
| N = "Hello World" | Does N match the literal text "Hello World" |

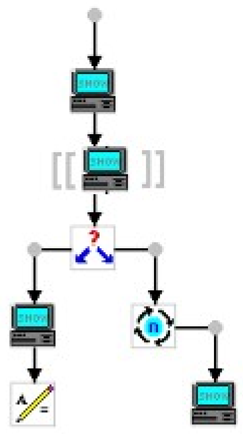
How to invoke Alternate Path

## How to invoke Alternate Path

- Use the mouse to click on the Alternate Path button on the vertical toolbar.
- Fill out the Alternate Path dialog box.
- Click on the OK button.

Syntax errors in the condition expression may cause errors. These errors will be detected before even drawing the object on the diagram. An alert dialog will be displayed and the user will have to correct or cancel.

**Trace** **Animated execution of program**



The Trace option pops up an execution window to display output and executes the current GAP program. This option executes the GAP program slowly. The diagram canvas will be animated to demonstrate how the GAP program executes.

As the program advances slowly, the current command will be moving side to side on the canvas. As the user watches the canvas, execution window and Show Variable window together it will be easier to understand how the program works.

How to use Trace

## How to use Trace

- Point the mouse arrow on the Trace icon and use the left click button.
- Or move the mouse until the pointer on the screen is on the action menu, and click the left button.
- Move the pointer to the Trace option and click the left button again.

## Help Menu

The Help option is a Windows application that was built with the GAP application as its user. Information about all GAP features is included and can be viewed by a GAP user by selecting Help in the main menu bar.

The Help button in pop-up dialog boxes will jump directly to the relative help topic page in the help file. Hopefully, the information will assist the user with the task at hand by providing useful information.

# Tutorial

By using the **Tutorial** you can become familiar with making simple **GAP** programs.

## How to display HELLO WORLD

You can display **Hello World** in several different ways. Below are two examples:

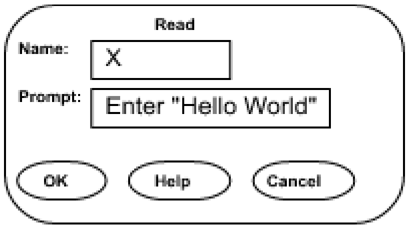
**I. Program to Display Hello World**

**1. Ask user for text "Hello World"**

**2. Show text on user screen**

Select the **Read** icon

**1.** Enter into the read dialog box



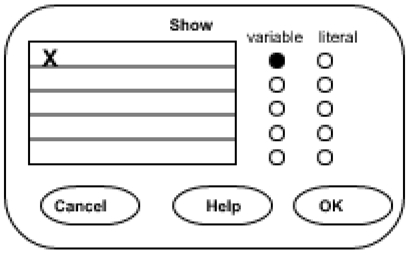
Name: X

Prompt: Enter "Hello World"

Select the **OK** button

Select the **Show** icon

**2.** Enter into the show dialog box



X [✓ variable]

Select the **OK** button

Push the **Run** icon to execute your program

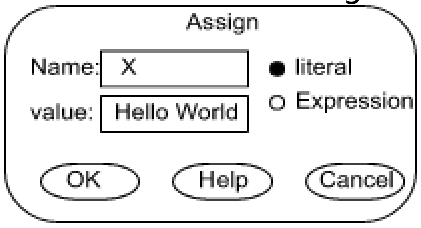
**II. Program to Display Hello World**

**1. Assign "Hello World" to variable X.**

**2. Show variable X on user screen**

Select the **Assign** icon

**1.** Enter into the assign dialog box



Name: X

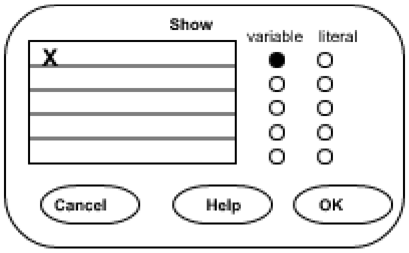
value: Hello World

[✓ literal] Expression

Select the **OK** button

Select the **Show** icon

**2.** Enter into the show dialog box



X [✓ variable]

Select the **OK** button

Push the **Run** icon to execute your program

## How to use loops

There are two different loops **GAP** uses. The **conditional loop** that is used to run indefenitely as long as the condition is true and there is also the **Iterative loop** that is used to run for a fixed number of times.

You can use loops to display **Hello World** several times to the user screen. Below are two examples:

**I. Program to use**

**II. Program to use**

**Interative Loop**

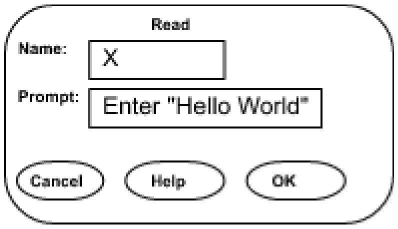
**1. Ask user for text "Hello World"**

**2. Loop text on screen 5 times**

**2.1 Show text on user screen**

Select the **Read** icon

**1.** Enter into the read dialog box

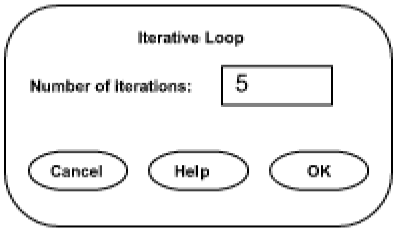

Read

Name: X

Prompt: Enter "Hello World"

Select the **OK** button

**2.** Select the **Iterative Loop** icon

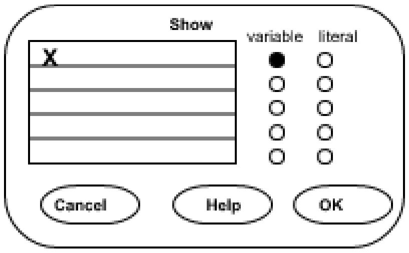

Iterative Loop

Number of Iterations: 5

Select the **OK** button

Select the **Show** icon

**2.1** Enter into the show dialog box


Show

| | variable | literal |
|---|---|---|
| X | ● | ○ |
| | ○ | ○ |
| | ○ | ○ |
| | ○ | ○ |
| | ○ | ○ |

Select the **OK** button

Push the **Run** icon to execute your program

**Conditional Loop**

**1. Assign "Hello World" to variable X.**

**2. Assign "1" to variable NUM.**

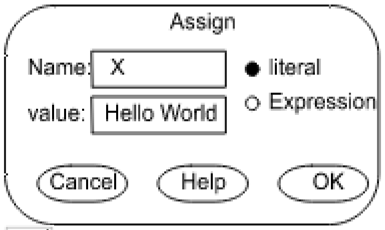
**3. Loop text on screen until NUM < 5**

**3.1 Show variable X on user screen**

**3.3 Increment variable NUM**

Select the **Assign** icon

**1.** Enter into the assign dialog box


Assign

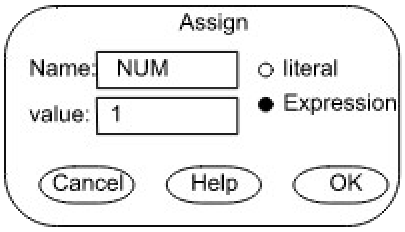
Name: X

value: Hello World

[✓ literal] Expression

Select the **Assign** icon

**2.** Enter into the assign dialog box


Assign

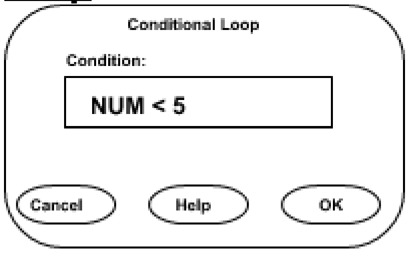
Name: NUM

value: 1

literal [✓ Expression]

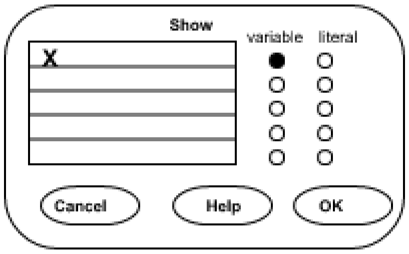
Select the **OK** button

**3.** Select the **Conditional Loop** icon


Conditional Loop

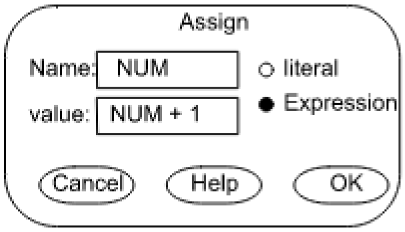
Condition: NUM < 5

Select the **Show** icon
**3.1** Enter into the show dialog box



X

Select the **OK** button
**3.2** Enter into the assign dialog box



Name: NUM
value: NUM + 1
○ literal
● Expression

Select the **OK** button
Push the **Run** icon to execute your program

## How to use Alternate Path

The **Alternate Path** command will provide a choice of program flow depending on the condition of an expression.

You can use **Alternate Path** to display **Hello World** several times or only once to the user screen. Below is an example:

**I. Program to demo Alternate Path**

**1. Assign "Hello World" to variable X.**

**2. Read from user what variable NUM is equal to**

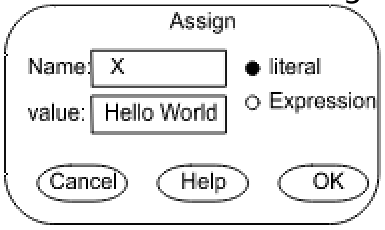
**3. Display NUM times or once**

**3.1 If YES; Loop text on screen NUM times**

**3.2 If NO Show text on**

**screen once**

Select the **Assign** icon
**1.** Enter into the assign dialog box



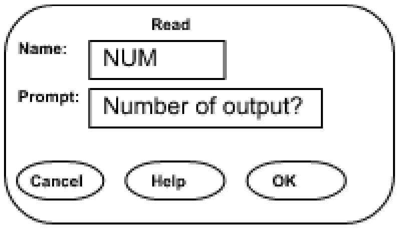
X

Hello World

● literal ○ Expression

Select the **OK** button

Select the **Read** icon
**2.** Enter into the read dialog box

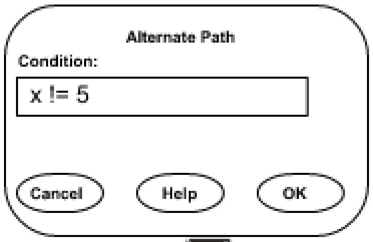


NUM

Number of output?

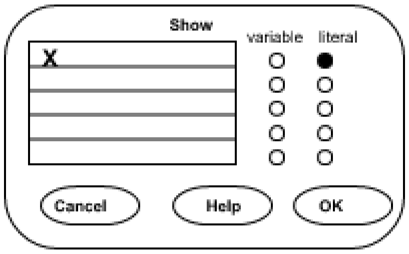
Select the **OK** button

**3.** Select the **Alternate Path** icon



x != 5

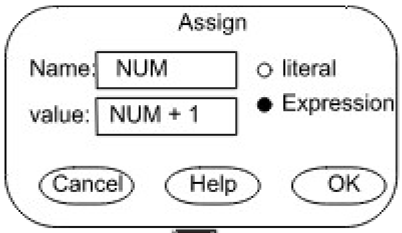
**3.1** If **YES** Select the **Show** icon
**3.1.1** Enter into the show dialog box



X ● literal

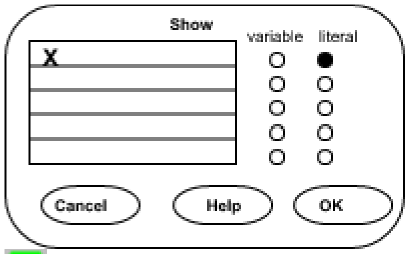
Select the **OK** button

Select the **Assign** icon
**3.1.2.** Enter into the assign dialog box



**3.2** If **NO** Select the **Show** icon

**3.2.1** Enter into the show dialog box



Push the **Run** icon to execute your program

## Glossary

Assign

Canvas

Expression

object

Literal Text

Loop

National University

Operators

Value

Variable

## Assign

Sets a letter or string of letters to be equal to a number or string of text for a purpose.

## Canvas

The **canvas** is the work area of the GAP application. This is the area where programmers/users will create program diagrams and prepare to execute them.

## Expression

The designation of any symbolic mathematical form, such as an equation.

## object

A graphical object is a symbol or drawing that visually represents what it is for. In the GAP programming language graphical objects will be seen on the screen representing commands **Assign**, **Conditional Loop**, **Show**, **Read**, **Iterative Loop** and **Alternate Path**.

## Literal Text

Is the set of text characters equal to the English alphabet like A-Z , a-z and 0-9. The text may or may not have literal meaning.

## Loop

Most of a program's important work involves controlled repetition, in which a group of statements repeatedly does a particular job until the work is done. For example consider data entry routines of a database program, this group of statements used to received, validate and store data must be repeated as long as the user wants to enter new data records.   This set of repeating statement is called a LOOP, because it is executed as though the statements were arranged in a circle.

## National University

National University

4025 Camino del Rio South

San Diego, CA 92108-4107

(619)563-7100

## Operators

Plus

Minus

Divide

Multiply

Raise

Mod

**<u>Plus</u>**

'+' The PLUS operator is the process of computing the addition of numbers to find their sum. The plus sign causes the Two values on either side of the sign to be added together.

In the GAP language, assigning numbers to variables then performing mathematical operations on the variables can be easily accomplished (a=a+b, a=12, b=2, a=14).

## Minus

'-' The MINUS operator is the process of computing the subtraction of numbers to find there reduced value.   The minus sign causes the number after the (-) sign to be subtracted from the number before the sign.

In the GAP language, assigning numbers to variables then performing mathematical operations on the variables can be easily accomplished (a=a-b, a=12, b=2, a=10).

## Divide

'/' The DIVIDE operator subjects numbers to the process of division. The value to the left of the / is divided by the value to the right.

In the GAP language, assigning numbers to variables then performing mathematical operations on the variables can be easily accomplished (a=a/b, a=12, b=2, a=6).

## Multiply

'*' The MULTIPLY operator subjects numbers to the process of multiplication. It is placed between two real numbers in which the number of times either is taken in summation is determined by the value of the other.

In the GAP language assigning numbers to variables then performing mathematical operations on the variables can be easily accomplished (a=a*b, a=12, b=2, a=24).

**Raise**

'^' (Exclusive OR)   This operator produces a value in which each bit is set to 1 only if one or the other (but not both) of the corresponding bits of the two operands is 1.

**Mod**

'%' The Modulus operator is used in integer arithmetic.   It is the absolute value   of a complex number. Modulus gives the remainder that results when the integer to its left is divided by the integer to its right then multiplied by the integer to the right. To state it in math terms:

(a/b)*b   or a%b is equal to a.   If a=12.123 then 12%123 ==12/123*123 = 12.

## Value

Is a assigned or calculated numerical or text string quantity. What this means is a string of characters or numbers can be assigned to another name or variable.

In the GAP programming language the value of X, if X = 98, is 98 and the value of X, if X = "Hello World", is "Hello World".

## Variable

Has no fixed quantitative value. A variable is a quantity capable of assuming any of a set of values.

In the GAP programming language the variable X's value, if X = 98, is 98 and the value of X, if X = "Hello World", is "Hello World". In a programming environment, the variable X can change as the program executes. X = X + 2 will increase the value of X by two so that X now equals 100.   This also holds true for the string example,   X = "Hello World" can be re-assigned during programming execution to X = "Good-by World".

expression

Commands / Assign

Glossary / Expression

variable

Action Menu Options / Show Variables

Show Variables / How to use Show Variable

Commands / Assign

Glossary / Variable

value

Commands / Assign

Glossary / Value

assign

Commands / Assign

Assign / How to invoke Assign

Glossary / Assign

loop

Commands / Conditional Loop

Conditional Loop / How to invoke Conditional Loop

Commands / Iterative Loop

Iterative Loop / How to invoke Iterative Loop

Tutorial / How to use loops

Glossary / Loop

Read

Commands / Read

Read / How to invoke Read

Show

Action Menu Options / Show Variables

Show Variables / How to use Show Variable

Commands / Show

Show / How to invoke Show

Run

Action Menu Options / Run

Run / How to use Run

Iterative Loop

Commands / Iterative Loop

Iterative Loop / How to invoke Iterative Loop

Conditional Loop

Commands / Conditional Loop

Conditional Loop / How to invoke Conditional Loop

Alternate Path

Commands / Alternate Path

Alternate Path / How to invoke Alternate Path

Tutorial / How to use Alternate Path