Strings – Length

The ability to convert any data type to a string has advantages. It is possible to manipulate strings character by character. This can be useful with problems involving words as well as just numbers.

length(*string*) – determines the length of *string*

```
quote := "To be or not to be"
quoteLen := length(quote)

% output that it is 18 characters long
put quote, " : ", quoteLen, " characters"
```

Strings – Joining Strings Together (Concatenation)

We have already output strings together using the put command. In order to combine strings and save the result in a variable, use the '+' operation to concatenate the strings.

```
quote1 := "To be or not to be"
quote2 := "that is the question."

quote := quote1 + ", " + quote2
% added the ',' and space for formatting
```

Strings – Accessing Substrings

```
var quote : string
quote := "To be or not to be"

put quote          % output whole quote
put quote(1..*)    % output whole quote
put quote(1)       % output first char
put quote(1..1)    % output first char
put quote(1..5)    % output first 5 chars
put quote(3..5)    % output 3rd, 4th, 5th
put quote(*-4..*)  % output last 5 chars
```

Strings – Accessing Substrings (saving to a new variable)

```
var fullQuote, quote1, quote2 : string
var newQuote : string
fullQuote := "To be or not to be"

put fullQuote            % output whole quote
quote1 := fullQuote(1..5)
put quote1               % output "To be"
quote2 := fullQuote(*-4..*)
put quote2               % output "to be"

newQuote := quote1 + quote2
put newQuote             % output "To beto be"
```

Strings – Looking for a Substring

A set of letters (or a single letter) that is part of a larger string is called a substring. To search for a pattern in a string, Turing has:

index(*string, pattern*) – returns the starting position of the *pattern* in *string*

```
var location : int
location := index("chair", "ai")
put location
% outputs 3, since "ai" starts at the 3rd
% character
```

String – Changing Case

When comparing strings, it is often inconvenient to have to worry about upper and lower case (e.g., "yes" vs "YES" vs "Yes").

```
var word : string
put "Word? "..
get word : *

put "Your word is ", word
put "Your word is ", Str.Upper(word)
put "Your word is ", Str.Lower(word)
```

String – Integer Conversions

strint – converts a string to an integer

intstr – converts an integer to a string

```
numString := "17"
intNum := strint(numString)
put intNum * 2    % output 34 to screen
```

String – Real Conversions

strreal – converts a string to a real

realstr – converts a real to a string

```
numString := "3.14"
realNum := strreal(numString)
put realNum * 2     % output 6.28 to screen
```