Working with Strings

Data Types:

A string is a collection of one or more characters that can be output (printed) or input (via the keyboard).

Although we typically think of strings as names, words, and punctuation, they can also refer to numbers, mathematical symbols, or other special characters.

String Examples

```
var example : string

example := “Hello”

example := “123456”

example := “123 Brookfield Road”

example := “9 + 3 * 4 – 12 / 6”
```

Strings are... everything

We type our programs and input using the keyboard. The output of our programs appears on the screen.

Although we distinguish between different data types (which is very important), at some point everything was input as a string, or will be output as a string.

String Conversions

There are many conversions between strings and other data types (e.g., integer, real) that occur behind the scenes. We don't need to be involved, and it makes our programs easier to create.

There are times, however, when our programs may need a specific knowledge, and control, of the conversion process.

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
```

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 – 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*

```
put index("chair", "air")
% outputs 3, since "air" starts at the 3rd
% character
```

Strings – Accessing Substrings

We can also specify the part of the string we want by position. Each string has characters from (1..*length*) or (1..*), where '*' is the last character.

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

put quote                 % output whole quote
put quote(1..*)           % output whole quote
put quote(1..5)           % output "To be"
put quote(*-4..*)         % output "to be"
```

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
```