

## 11.1.2 Built-in functions and library routines

---

Name: \_\_\_\_\_ Class: \_\_\_\_\_ Date: \_\_\_\_\_

Total: 51 marks

**KEY WORDS** built-in functions • library routines • parameters  
• concatenates • program library

### Objective

---

Learn to use the **built-in functions** and **library routines** of pseudocode: the string functions, the conversion functions and the numeric functions that the Paper 2 **insert** gives you. This is the last sheet of 11.1. You need the variables of sheet 11.1 and the operators of sheet 11.1.1.

**You must be able to:**

- use the string functions **LENGTH**, **LEFT**, **RIGHT**, **MID**, **TO\_UPPER** and **TO\_LOWER**
- join strings with **&**, and convert between numbers and strings with **NUM\_TO\_STR** and **STR\_TO\_NUM**
- use **ASC**, **CHR**, **INT** and **RAND**
- work out an expression in which one function is inside another
- state the benefits of using a program library

### 1 Key knowledge and worked examples

---

Study these first. Each one shows a **method** that a question later on this sheet needs.

#### ■ What a function is

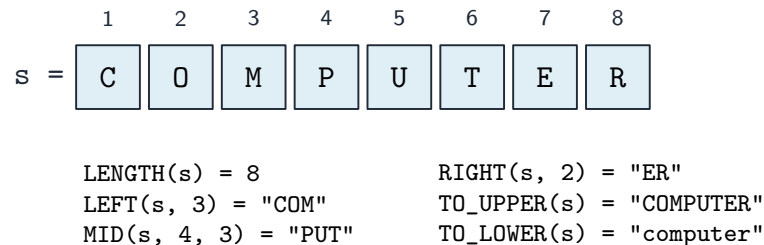
A **function** takes values (its **parameters** ), does a job, and **returns** a result. The result **replaces** the call:

`Size ← LENGTH("Computer")` stores 8 in `Size`.

The exam gives this list on the insert: do not learn it by heart, but use the **exact names** and the **correct order** of the parameters.

## ■ The string functions

- Positions are counted from **1**, and a space counts as a character: `LENGTH("Happy Days")` is 10.
- `TO_UPPER` and `TO_LOWER` take a **CHAR** or a **STRING**. The Pseudocode Guide also has `UCASE` and `LCASE`, which take **one CHAR only**.



| function                      | returns                                           | example                                     |
|-------------------------------|---------------------------------------------------|---------------------------------------------|
| <code>LENGTH(s)</code>        | the number of characters                          | <code>LENGTH("Computer")</code> is 8        |
| <code>LEFT(s, n)</code>       | the first <code>n</code> characters               | <code>LEFT("Computer", 3)</code> is "Com"   |
| <code>RIGHT(s, n)</code>      | the last <code>n</code> characters                | <code>RIGHT("Computer", 2)</code> is "er"   |
| <code>MID(s, start, n)</code> | <code>n</code> characters from <code>start</code> | <code>MID("Computer", 4, 3)</code> is "put" |
| <code>TO_UPPER(x)</code>      | <code>x</code> in capital letters                 | <code>TO_UPPER("hi")</code> is "HI"         |
| <code>TO_LOWER(x)</code>      | <code>x</code> in small letters                   | <code>TO_LOWER("Hi")</code> is "hi"         |

## ■ Joining and converting

- The operator **&** **concatenates** (joins) two strings: `"Sum" & "mer"` is "Summer".
- **+** does **not** join strings, and a number cannot be joined to a string. Convert it first:

| function      | returns                                    | example                      |
|---------------|--------------------------------------------|------------------------------|
| NUM_TO_STR(x) | the number <b>x</b> as a string            | NUM_TO_STR(87.5) is "87.5"   |
| STR_TO_NUM(s) | the string <b>s</b> as a number            | STR_TO_NUM("23.45") is 23.45 |
| IS_NUM(s)     | TRUE if <b>s</b> is a valid number         | IS_NUM("12a") is FALSE       |
| ASC(c)        | the ASCII code of the character <b>c</b>   | ASC('A') is 65               |
| CHR(n)        | the character with the ASCII code <b>n</b> | CHR(66) is 'B'               |

"Total: " & Score is an error when Score is an INTEGER. The correct statement is "Total: " & NUM\_TO\_STR(Score).

### ■ Numeric functions

| function | returns                                    | example               |
|----------|--------------------------------------------|-----------------------|
| INT(x)   | the whole-number part of <b>x</b>          | INT(27.5415) is 27    |
| RAND(n)  | a random <b>real</b> number below <b>n</b> | RAND(6) could be 4.27 |

RAND(6) returns a real number from 0 up to, but **not including**, 6, so a random **integer** needs INT. Dice, 1 to 6: INT(RAND(6)) gives 0 to 5, so add 1. From Low to High: INT(RAND(High - Low + 1)) + Low.

### ■ A function inside a function

Work from the **inside out**: work out the inner function first, and put its result in the outer one.

*Worked example:* TO\_UPPER(RIGHT("Program", 2)).

1. Inner: RIGHT("Program", 2) is "am".
2. Outer: TO\_UPPER("am") is "AM".

*Worked example:* LENGTH("Hello") + STR\_TO\_NUM("4"). LENGTH("Hello") is 5 and STR\_TO\_NUM("4") is 4, so the result is 9.

### ■ Program libraries

A **program library** is a collection of routines that are already written, tested and compiled. Benefits of using one:

- it **saves time**: no new code to write and test
- the routines are already **tested**, so they are reliable
- they are written by **experts**, and may do a job the programmer could not write
- a routine can be **used many times**

■ **Key knowledge: dates use a DATE value**

SETDATE(day, month, year) takes three integers in that order and returns a DATE, not a string. DAYINDEX(DateValue) gives the weekday number, where Sunday is 1; MONTH(DateValue) gives the month number. Read the insert for the exact names and types.

**Worked example.** Meeting ← SETDATE(6, 10, 2026) stores 6 October 2026.  
MonthsLeft ← 12 - MONTH(Meeting) stores 2.

## 2 Practice

Now use the methods above. The first questions need one idea each.

**2.1** Name the function that does each job. [5]

- (a) returns the number of characters in a string
- (b) returns the last few characters of a string
- (c) changes a string into a number
- (d) returns the ASCII code of a character
- (e) returns the whole-number part of a real number

C o m p u t e r

**2.2** Give the value of each expression. [4]

- (a) LENGTH("Computer")
- (b) RIGHT("Computer", 3)
- (c) MID("Computer", 4, 2)
- (d) INT(STR\_TO\_NUM("9.7"))

"Computer"

string

"9.7"

string

**2.3** The variable `Word` holds "Keyboard". Work out the value of each expression. [4]

(a) `LEFT(Word, 3)`

(b) `MID(Word, 4, 2)`

(c) `TO_UPPER(RIGHT(Word, 5))`

(d) `LENGTH(Word) DIV 3`

Word 

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| K | e | y | b | o | a | r | d |
|---|---|---|---|---|---|---|---|

---

---

**2.4** Work out the value of each expression. [4]

(a) `ASC('C')`, given that `ASC('A')` is 65

(b) `CHR(70)`

(c) `"AB" & "12"`

(d) `STR_TO_NUM("12") + 3`

|     |
|-----|
| 'A' |
|-----|

 → 

|    |
|----|
| 65 |
|----|

---

---

**2.5** Write the pseudocode statement that assigns to `Dice` a random integer from 1 to 6, and state why `INT` is needed. [2]



---

---

**2.6** A student writes `Message ← "Total: " + Score`, where `Score` is an `INTEGER`. State what is wrong and write the correct statement. [2]

Total: 

|  |
|--|
|  |
|--|

---

---

[3]

[3]

[4]

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| S | a | t | u | r | d | a | y |
| N | o | t | e | b | o | o | k |

**3.3** Write pseudocode that reads a word and outputs it in **capitals** followed by its **length**, using built-in functions. [3]



**3.4** A school builds each student's username from their surname, their first name and their year group: the **first three letters of the surname in capitals**, then the **first letter of the first name**, then the **year group**.

Write a single pseudocode statement that assigns this username to **Username**, using the variables **Surname**, **FirstName** and **Year**, where **Year** is an **INTEGER**. [4]

 Surname: Garcia  
First name: Lina  
Year: 12

**3.5** State **three** benefits of using a program library rather than writing the routines yourself. [3]

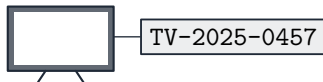
program library

**3.6** A shop gives every product a code such as "TV-2025-0457". The code has three parts: two letters for the type of product, the year the product was first sold, and a product number. The code is stored in the variable **Code**.

(a) Work out the value of each expression when **Code** is "TV-2025-0457". [3]

| Expression                      | Evaluates to |
|---------------------------------|--------------|
| LEFT(Code, 2)                   |              |
| STR_TO_NUM(MID(Code, 4, 4)) + 1 |              |
| LENGTH(Code) - 8                |              |

(b) Write **one** pseudocode statement that stores the product number, as an **INTEGER**, in the variable **ProductNumber**. [3]



(c) The shop makes a short label from the code: the two letters in small letters, then the last two digits of the year. For "TV-2025-0457" the label is "tv25". Write **one** pseudocode statement that stores the label in the variable **Label**. [4]

---



---



---