Section 10.1 - Data Types and Records

Layer 6: High-Order Language

Syllabus Content Section 10: Data Types and Structures

S10.1.1 Select and use appropriate data types for a problem solution

- including integer, real, char, string, Boolean, date (pseudocode will use the following data types: INTEGER, REAL, CHAR, STRING, BOOLEAN, DATE, ARRAY, FILE)

---

data type		examples
INTEGER	a whole number	-5, 3
REAL	a number capable of containing a fractional part	4.7, 0.3, -4.0, 0.0
CHAR	a single character	'x', 'C', '@'
STRING	a sequence of zero or more characters	"This is a string", ""
BOOLEAN	the logical values TRUEand FALSE	TRUE, FALSE
DATE	a valid calendar date	dd/mm/yyyy

S10.1.2 Show understanding of the purpose of a record structure to hold a set of data of different data types under one identifier

- Write pseudocode to define a record structure.
- Write pseudocode to read data from a record structure and save data to a record structure

---

Composite data type

```
TYPE <identifier1>
        DECLARE <identifier2> : <data type>
    DECLARE <identifier3> : <data type>
    ...
ENDTYPE
```

- EXAMPLE - declaration of composite type

```
        TYPE Student
                DECLARE LastName : STRING
                DECLARE FirstName : STRING
                DECLARE DateOfBirth : DATE
                DECLARE YearGroup : INTEGER
                DECLARE FormGroup : CHAR
        ENDTYPE
```

Using user-defined data types

- Example – using user-defined data types

```
        DECLARE Pupil1 : Student
        DECLARE Pupil2 : Student
        DECLARE Form : ARRAY[1:30] OF Student
        DECLARE ThisSeason : Season
        DECLARE NextSeason : Season
        DECLARE MyPointer : TIntPointer

        Pupil1.LastName ← "Johnson"
        Pupil1.Firstname ← "Leroy"
        Pupil1.DateOfBirth ← 02/01/2005
        Pupil1.YearGroup ← 6
        Pupil1.FormGroup ← 'A'

        Pupil2 ← Pupil1

        FOR Index ← 1 TO 30
              Form[Index].YearGroup ← Form[Index].YearGroup + 1
        NEXT Index

        ThisSeason ← Spring
        MyPointer ← ^ThisSeason
```

```
        NextSeason ← MyPointer^ + 1
        // access the value stored at the memory address
```