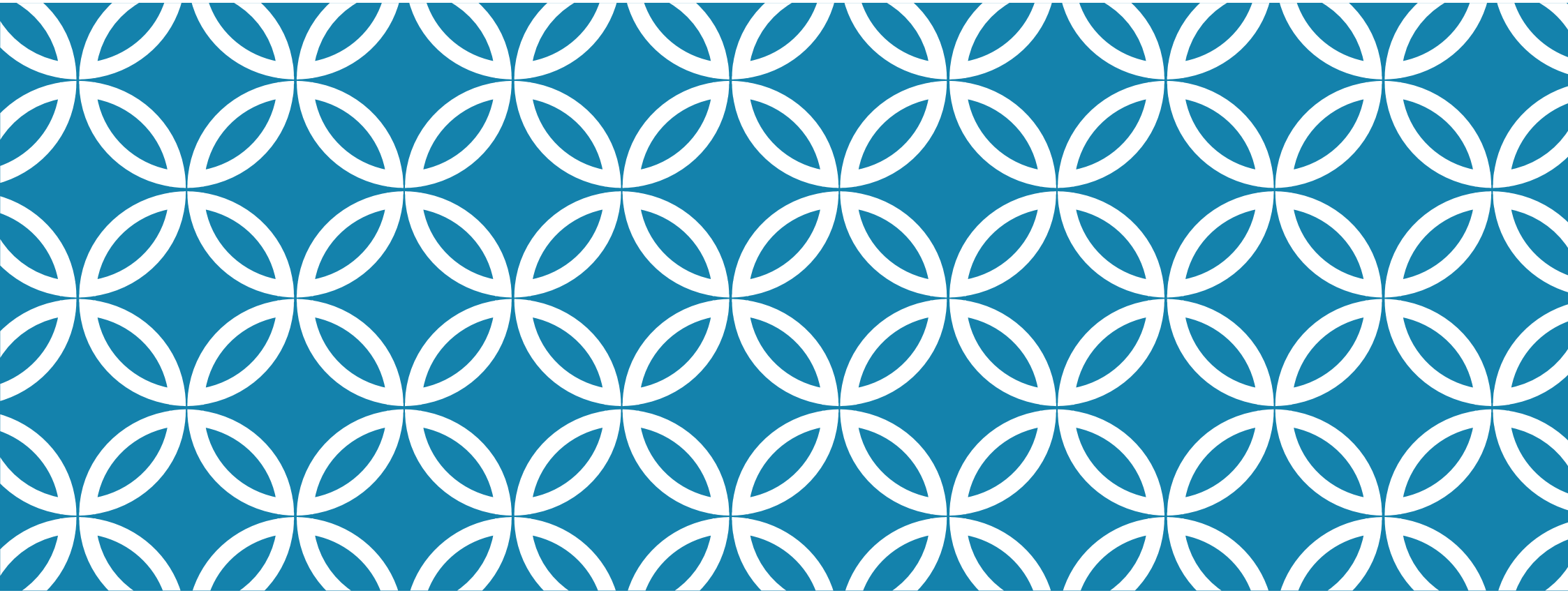

PSEUDOCODE

Chapter 8

WHY PSEUDOCODE?

- Cross-Platform Understanding
  - There are many programing languages e.g. Python, Java, C
  - Each programing language has its own instructions
  - Same algorithm needs to be implemented in different programming languages
  - It helps ensure that the core logic remains consistent across different implementations
- Clarity and Communication
  - It's a more human-readable and intuitive representation, making it easier for team members to collaborate and discuss ideas
  - Pseudocode provides a way for developers, designers, and stakeholders to communicate and understand the logic of an algorithm or program without getting bogged down by the syntax of a specific programming language

DATA TYPE

- Data types in pseudocode
  - INTEGER : a whole number
  - REAL : a number capable of containing a fractional part
  - CHAR : a single character
  - STRING : a sequence of zero or more characters
  - BOOLEAN : the logical values TRUE and FALSE
- Identifiers
  - The names given to variables, constants, procedures and functions
  - Identifiers should be meaningful that tell the purpose of them
    - Name ← “Alex”; this identifier stores a name of person
    - Class_score ← [10, 20, 15]; this identifier stores scores from a classroom

VARIABLE AND CONSTANT DECLARATION

- Declaration is on the upper part of the coding
- It tells users that what variables/constants will be used in the program and what is the data type of those variable
- Examples
  - DECLARE Counter : INTEGER
  - DECLARE TotalToPay : REAL
  - DECLARE GameOver : BOOLEAN

  - CONSTANT HourlyRate ← 6.50
  - CONSTANT Tax ← 0.07

ARRAY DECLARATION

- Arrays are fixed-length structures of elements of identical data type
- DECLARE StudentNames : ARRAY[1:30] OF STRING
- DECLARE ClassScore : ARRAY[1:20, 1:3] OF INTEGER // 20 students with 3 subjects

- Using arrays
  - StudentNames[1] ← "Ali"
  - NoughtsAndCrosses[2,3] ← 'X'
  - StudentNames[n+1] ← StudentNames[n]

ASSIGNMENT STATEMENTS

- The assignment operator is ←
- In Python is "="

- Counter ← 0
- Counter ← Counter + 1
- TotalToPay ← NumberOfHours * HourlyRate

INPUT AND OUTPUT

- Values are input using the INPUT command as follows:
  - INPUT <identifier>
- Values are output using the OUTPUT command as follows:
  - OUTPUT <value(s)>

- INPUT Answer
- OUTPUT Score
- OUTPUT "You have ", Lives, " lives left"

ARITHMETIC OPERATIONS

- Standard arithmetic operator symbols are used:
  - + addition
  - – subtraction
  - * multiplication
  - / division
  - ^ raised to the power of

- Answer ← Score * 100 / MaxMark
- Answer ← Pi * Radius ^ 2

DIV AND MOD FUNCTIONS

- DIV(<identifier1>, <identifier2>)
  - Returns the quotient of identifier1 divided by identifier2 with the fractional part discarded
  - OUTPUT DIV(10, 3) // show 3 on the screen

- MOD(<identifier1>, <identifier2>)
  - Returns the remainder of identifier1 divided by identifier2
  - OUTPUT MOD(10, 3) // show 1 on the screen

LOGICAL OPERATORS

- The following symbols are used for logical operators:
  - = equal to // Python ==
  - < less than
  - <= less than or equal to
  - > greater than
  - >= greater than or equal to
  - <> not equal to // Python !=

BOOLEAN OPERATORS

- The only Boolean operators used are AND, OR and NOT

```
IF Answer < 0 OR Answer > 100
   THEN
         Correct ← FALSE
   ELSE
         Correct ← TRUE
ENDIF
```

STRING FUNCTIONS

- LENGTH(<identifier>)
  - Returns the integer value representing the length of string. The identifier should be of data type string
    - LENGTH("Happy Days") will return 10
    - Name ← “Alex”
    - OUTPUT LENGTH(Name) // show 4 on the screen

- SUBSTRING(<identifier>, <start>, <length>)
  - Returns a string of length *length* starting at position start. The identifier should be of data type string, length and start should be positive and data type integer.
    - SUBSTRING("Happy Days", 1, 5) will return "Happy"

STRING FUNCTIONS

- LCASE(<identifier>)
  - Returns the string/character with all characters in lower case
  - The identifier should be of data type string or char
    - LCASE('W') will return 'w'

- UCASE(<identifier>)
  - Returns the string/character with all characters in upper case
  - The identifier should be of data type string or char
    - UCASE("Happy") will return "HAPPY"