Section 11.2 - Constructs

Layer 6: High-Order Language

Syllabus Content Section 11: Programming

S11.2.1 Use pseudocode to write:

- an 'IF' statement including the 'ELSE' clause and nested IF statements
- a 'CASE' structure
- a 'count-controlled' loop:
- a 'post-condition' loop
- a 'pre-condition' loop

> IF statements

```
// IF statements without an ELSE clause
IF <condition> THEN
    <statement(s)>
ENDIF

//IF statements with an ELSE clause
IF <condition> THEN
    <statement(s)>
ELSE
    <statement(s)>
ENDIF
```

- EXAMPLE – nested IF statements

```
IF ChallengerScore > ChampionScore THEN
    IF ChallengerScore > HighestScore THEN
          OUTPUT ChallengerName, " is champion and highest scorer"
    ELSE
          OUTPUT ChallengerName, " is the new champion"
    ENDIF
ELSE
    OUTPUT ChampionName, " is still the champion"
```

```
    IF ChampionScore > HighestScore THEN
          OUTPUT ChampionName, " is also the highest scorer"
    ENDIF
ENDIF
```

CASE statements

```
// CASE statements are written as follows:
CASE OF <identifier>
    <value 1> : <statement1>
                <statement2>
                ...
    <value 2> : <statement1>
                <statement2>
                ...
    ...
ENDCASE

// An OTHERWISE clause can be the last case:
CASE OF <identifier>
    <value 1> : <statement1>
                <statement2>
                ...
    <value 2> : <statement1>
                <statement2>
                ...
    OTHERWISE : <statement1>
                <statement2>
                ...
ENDCASE

// Each value may be represented by a range, for example:
<value1> TO <value2> : <statement1>
                       <statement2>
                       ...
```

- EXAMPLE – formatted CASE statement

```
INPUT Move
CASE OF Move
      'W' : Position ← Position – 10
      'S' : Position ← Position + 10
      'A' : Position ← Position – 1
```

```
      'D' : Position ← Position + 1
      OTHERWISE : CALL Beep
ENDCASE
```

Count-controlled (FOR) loops

```
FOR <identifier> ← <value1> TO <value2>
    <statement(s)>
NEXT <identifier>
//An increment can be specified
FOR <identifier> ← <value1> TO <value2> STEP <increment>
    <statement(s)>
NEXT <identifier>
```

- Example – nested FOR loops

```
Total ← 0
FOR Row ← 1 TO MaxRow
  RowTotal ← 0
  FOR Column ← 1 TO 10
      RowTotal ← RowTotal + Amount[Row, Column]
  NEXT Column
  OUTPUT "Total for Row ", Row, " is ", RowTotal
  Total ← Total + RowTotal
NEXT Row
OUTPUT "The grand total is ", Total
```

Post-condition (REPEAT) loops

```
REPEAT
    <statement(s)>
UNTIL <condition>
```

- Example – REPEAT UNTIL loop

```
REPEAT
  OUTPUT "Please enter the password"
  INPUT Password
UNTIL Password = "Secret"
```

Pre-condition (WHILE) loops

```
WHILE <condition>
    <statement(s)>
ENDWHILE
```

- Example – REPEAT UNTIL loop

```
WHILE Number > 9
  Number ← Number - 9
ENDWHILE
```

S11.2.2 Justify why one loop structure may be better suited to solve a problem than the others

---

Make our programs more efficient and take up less memory (reduce time complexity and space complexity)