

**Unit 303 Develop Software using Pascal Level 3 (Core)****Rationale**

The aim of this unit is to enable candidates to understand the principles required to develop software using the 'Pascal' programming language. Candidates will develop the skills required to design, create and test software to solve a given problem.

There are 6 outcomes to this unit. The candidate will be able to:

1. interpret program specifications to develop software
2. develop software components
3. use functions
4. save and retrieve data from disk
5. describe the principles of software testing and management
6. produce documentation.

**Guided learning hours**

The recommended guided learning hours for this unit are 90 hours.

**Connections with other awards****NVQ links**

| Outcome                     | This award contributes to the knowledge and understanding of the following elements of NVQ(s)  |
|-----------------------------|--|
| 1<br>2, 3, 4<br>5<br>5<br>6 | <i>C&amp;G 4300 Developing IT Programs Level 3:</i><br>309.1 Prepare for the creation of software<br>309.2 Create software components from program designs<br>309.3 Assemble the software components of program designs<br>309.4 Test software against functional requirements<br>309.5 Produce maintenance documentation for software |

**Key Skills links**

|                        |                     |
|------------------------|---------------------|
| Communication          | C1.3, C3.2          |
| Application of number  | N1.1                |
| Information technology | None                |
| Working with others    | None                |
| Improving own learning | LP3.1, LP3.2, LP3.3 |
| Problem solving        | PS3.1, PS3.2, PS3.3 |

**Assessment**

Assessment will be by means of a **set assignment** covering practical activities, and a **multiple choice test** covering underpinning knowledge.

*Outcome 1: Interpret program specifications to develop software.*

|  | Candidate's signature | Date |
|--|-----------------------|------|
| <p><b>Practical activities</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. use program specifications to design a solution</li> <li>2. use a program design method in the design</li> <li>3. use the compiler directive options</li> <li>4. create software from the design using a 'Pascal' text editor</li> <li>5. use good coding practice in the implementation of the design.</li> </ol>   |                       |      |
| <p><b>Underpinning knowledge</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. specify the components required to create software from project specifications</li> <li>2. identify the need for program design including pseudocode</li> <li>3. define the term 'compiler directive/option'</li> <li>4. describe the need for validation in a program</li> <li>5. explain the need for comments in a program</li> <li>6. explain the need for indentation in a program</li> <li>7. identify that all languages use algorithms</li> <li>8. identify the content and purpose of syntax diagrams</li> <li>9. identify that control structures take on a similar role between languages i.e.: - <ul style="list-style-type: none"> <li>• selection</li> <li>• sequence</li> <li>• iteration.</li> </ul> </li> </ol> |                       |      |

*Outcome 2: Develop software components.*

|  | Candidate's signature | Date |
|--|-----------------------|------|
| <p><b>Practical activities</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. use the control structures for iterative execution in a program: - <ul style="list-style-type: none"> <li>• <b>REPEAT...UNTIL</b></li> <li>• <b>WHILE DO</b></li> <li>• <b>FOR TO DO</b></li> <li>• <b>FOR DOWN TO DO</b></li> </ul> </li> <li>2. use a control structures as part of a decision making process</li> <li>3. use nested loops in a program</li> <li>4. use a <b>CASE</b> statement for selection</li> <li>5. use procedures in a program</li> <li>6. use <b>ROUND, TRUNC, PRED</b> and <b>SUCC</b> in a program</li> <li>7. use validation/range as part of data entry</li> <li>8. use logical operators</li> <li>9. use system-defined constants</li> <li>10. design algorithms to validate user input</li> <li>11. design and use one-dimensional and two-dimensional arrays</li> <li>12. create a program that uses the keywords: - <ul style="list-style-type: none"> <li>• <b>MAXINT</b></li> <li>• <b>TRUE</b></li> <li>• <b>FALSE</b></li> </ul> </li> <li>13. use the keyword <b>TYPE</b> to declare an enumerated type</li> <li>14. use the <b>TYPE</b> block to declare a programmer-designed data type and an array of records.</li> </ol> |                       |      |

**Underpinning knowledge**

The candidate will be able to:

1. identify the declaration of a user defined **TYPE**
2. describe the meaning of **TYPE**
3. state why validation is used
4. explain the meaning of the keywords: -
  - **ROUND**
  - **TRUNC**
  - **PRED**
  - **SUCC**
5. state the difference between a one and two-dimensional array
6. explain the meaning of a system-defined constant
7. describe the operation of the loops: -
  - **WHILE DO**
  - **FOR TO DO**
  - **REPEAT...UNTIL**
8. describe the operation of a **CASE** statement
9. explain how control structures can assist in the validation of user input
10. explain why procedures are used in a program.
11. explain the difference between global and local variables
12. explain that **MAXINT**, **TRUE** and **FALSE** are system-defined constants
13. explain that enumerated data types are ordinal.

*Outcome 3: Use functions.*

|  | Candidate's signature | Date |
|--|-----------------------|------|
| <p><b>Practical activities</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. declare a function/subprogram in a program</li> <li>2. pass values to a function/subprogram: - <ul style="list-style-type: none"> <li>• by <b>value</b> (copy)</li> <li>• by <b>reference</b></li> </ul> </li> <li>3. construct function/subprogram that return a system data type</li> <li>4. select pre-defined function(s) available to create software</li> <li>5. create a function that returns a programmer-defined data type</li> <li>6. write a program that uses an arithmetic expression as one of the parameters passed to a function</li> <li>7. use local and global variables in functions and procedures.</li> </ol>     |                       |      |
| <p><b>Underpinning knowledge</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. identify the declaration of a function/subprogram</li> <li>2. describe the purpose of a function</li> <li>3. explain the difference between a function and a procedure</li> <li>4. explain the meaning of pass by value</li> <li>5. explain the meaning of pass by reference</li> <li>6. identify the number and types of arguments that are in a subprogram parameter list</li> <li>7. identify from a function header the data type of the value returned by the function</li> <li>8. explain the difference between local and global variables</li> <li>9. identify the order and precedence for mathematical operators.</li> </ol> |                       |      |

*Outcome 4: Save and retrieve data from disk.*

|  | Candidate's signature | Date |
|--|-----------------------|------|
| <p><b>Practical activities</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. prepare a program for file handling by declaring a variable to be of <b>FILE</b> data type</li> <li>2. create files of text</li> <li>3. display on screen the contents of a text file</li> <li>4. manipulate files: - <ul style="list-style-type: none"> <li>• write to a file</li> <li>• read from a file</li> <li>• append to a file</li> </ul> </li> <li>5. search files on disk</li> <li>6. use <b>READ</b> AND <b>READLN</b> statements to read from a text file</li> <li>7. use the <b>EOF</b> function to detect the end of a text file.</li> </ol> |                       |      |
| <p><b>Underpinning knowledge</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. identify the correct method to: - <ul style="list-style-type: none"> <li>• write to a file</li> <li>• read from a file</li> <li>• append to a file</li> </ul> </li> <li>2. identify the correct method to search a file on disk</li> <li>3. list the data types that can be input from or written to a text file</li> <li>4. explain the meaning of <b>EOF</b></li> <li>5. describe the difference between <b>READ</b> and <b>READLN</b>.</li> </ol>   |                       |      |

*Outcome 5: Describe the principles of software testing and management.*

|  | <b>Candidate's signature</b> | <b>Date</b> |
|--|------------------------------|-------------|
| <p><b>Practical activities</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. verify that the program conforms to the design specification</li> <li>2. dry run a program</li> <li>3. implement change control procedures</li> <li>4. test the software according to a prepared test plan</li> <li>5. determine the expected outcome from a program</li> <li>6. compare the expected outcome to the actual outcome and decide if the program is working correctly</li> <li>7. keep a log of the results for each test</li> <li>8. clearly identify any discrepancies and any amendments made to correct errors</li> <li>9. resolve logical and run-time errors found during testing.</li> </ol> |                              |             |

**Underpinning knowledge**

**The candidate will be able to:**

1. explain the reasons for testing a program prior to implementation
2. describe the meaning of a dry run
3. describe the meaning of a compilation error
4. describe the meaning of a run-time error
5. describe the purpose of test data
6. explain the need to test software in the target environment
7. state the difference between interfacing and integration
8. explain why software is integrated
9. state that interface testing is conducted to evaluate whether systems or components pass data and control correctly to one another
10. state that interface design is the activity concerned with the interfaces of the software system contained in the software requirements documentation
11. state that interfacing consolidates the interface descriptions into a single interface description of the software system as a whole.
12. explain the need for an overall control when developing software projects
13. identify the benefits of teamwork to avoid duplication of effort, when working on a large project
14. describe the need for control and documentation of any changes to programs
15. explain the need to determine expected output when testing programs.

*Outcome 6: Produce documentation.*

|   | <b>Candidate's signature</b> | <b>Date</b> |
|---|------------------------------|-------------|
| <p><b>Practical activities</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. produce end user instructions that are clear and unambiguous</li> <li>2. produce technical documentation to describe the purpose of the program and its operation</li> <li>3. print listing of code.</li> </ol> |                              |             |
| <p><b>Underpinning knowledge</b><br/> <b>The candidate will be able to:</b></p> <ol style="list-style-type: none"> <li>1. identify the need for end user instructions</li> <li>2. describe the need for technical documentation to aid future maintenance and reusability.</li> </ol>   |                              |             |