



Australian Government

Department of Education, Employment and Workplace Relations

ICAPRG418A Apply intermediate programming skills in another language

Release: 1

ICAPRG418A Apply intermediate programming skills in another language

Modification History

Release	Comments
Release 1	This Unit first released with <i>ICA11 Information and Communications Technology Training Package version 1.0</i>

Unit Descriptor

This unit describes the performance outcomes, skills and knowledge required to undertake intermediate programming tasks using another programming language. The language may be an object-oriented language.

Application of the Unit

This unit applies to programmers in a variety of fields who are required to produce software programs.

Licensing/Regulatory Information

No licensing, legislative, regulatory or certification requirements apply to this unit at the time of endorsement but users should confirm requirements with the relevant federal, state or territory authority.

Pre-Requisites

Not applicable.

Employability Skills Information

This unit contains employability skills.

Elements and Performance Criteria Pre-Content

Element	Performance Criteria
<i>Elements describe the essential outcomes of a unit of competency.</i>	<i>Performance criteria describe the performance needed to demonstrate achievement of the element. Where bold italicised text is used, further information is detailed in the required skills and knowledge section and the range statement. Assessment of performance is to be consistent with the evidence guide.</i>

Elements and Performance Criteria

1. Code using user defined data structures	1.1 Design, define and use data structures that are an aggregate of other data types 1.2 Code using an array of user-defined data types 1.3 Use the facilities in the language to create, manipulate and destroy dynamic variables, such as arrays
2. Code using standard algorithms	2.1 When coding, use a modular programming approach, including pass-by-reference parameter passing 2.2 Write code to create and manipulate a 2-D array 2.3 Create and maintain a sorted array and use language-provided facilities for sorting an array of ordered elements 2.4 Code a simple binary search technique for use with an array of sorted data 2.5 Code binary file-handling solutions using random access algorithms
3. Debug code	3.1 Use stand-alone debugging tools or tools provided by integrated development environment to debug code 3.2 Use a debugger to trace code execution and examine variable contents to detect and correct errors
4. Document activities	4.1 Follow organisational guidelines for developing maintainable code and adhere to the provided coding standard when documenting activities 4.2 Apply internal documentation suitable for use by peers to all code created 4.3 Use documentation tools available in the target language when documenting activities
5. Test code	5.1 Design and document tests 5.2 Undertake limited testing of produced code to ensure compliance with program specification 5.3 Capture and record test results
6. Create an application	6.1 Build an application in response to a user requirement 6.2 Access multiple source code files 6.3 Employ integrated development environment project maintenance facilities or make files to automate program building 6.4 Develop a program specification solution when provided with a basic design document

	6.5 Design the algorithm, and document, construct and test applications in response to a problem description using the language
	6.6 Document the completed application

Required Skills and Knowledge

This section describes the skills and knowledge required for this unit.

Required skills

- analytical skills to:
 - read and interpret program specifications
 - undertake user testing
- communication skills to deal with clients and colleagues
- initiative and enterprise skills to produce a solution to the programming requirement
- literacy skills to:
 - follow organisational guidelines
 - produce documentation required for the chosen methodology
- problem-solving skills to debug problems with object-oriented code
- technical skills to:
 - apply a programming methodology to a project or scenario
 - interpret program specifications
 - translate from problem space to machine space
 - use a range of debugging tools
 - use an integrated development environment
 - use program-coding techniques.

Required knowledge

- detailed knowledge of:
 - dynamic variables
 - medium-size application development
 - standard array and file handling algorithms
 - user-defined data structures
- limited range of development methodologies and their application
- limited knowledge of language development.

Evidence Guide

The evidence guide provides advice on assessment and must be read in conjunction with the performance criteria, required skills and knowledge, range statement and the Assessment Guidelines for the Training Package.

Overview of assessment	
Critical aspects for assessment and evidence required to demonstrate competency in this unit	Evidence of the ability to: - design and build application programs to a problem scenario and program specification - generate code documentation and test it to confirm that created application meets original specification and solves original problem - produce technical documentation for the application program.
Context of and specific resources for assessment	Assessment must ensure access to: - software development environment - technical requirements - appropriate learning and assessment support when required - modified equipment for people with special needs.
Method of assessment	A range of assessment methods should be used to assess practical skills and knowledge. The following examples are appropriate for this unit: - verbal or written questioning to assess candidate's knowledge of: - arrays - dynamic variables - modular programming - evaluation of candidate's application - review of candidate's technical documentation.
Guidance information for assessment	Holistic assessment with other units relevant to the industry sector, workplace and job role is recommended, where appropriate. Assessment processes and techniques must be culturally appropriate, and suitable to the communication skill level, language, literacy and numeracy capacity of the candidate and the work being performed. Indigenous people and other people from a non-English speaking background may need additional support. In cases where practical assessment is used it should be combined with targeted questioning to assess required

	knowledge.
--	------------

Range Statement

The range statement relates to the unit of competency as a whole. It allows for different work environments and situations that may affect performance. Bold italicised wording, if used in the performance criteria, is detailed below. Essential operating conditions that may be present with training and assessment (depending on the work situation, needs of the candidate, accessibility of the item, and local industry and regional contexts) may also be included.

Data types may include:	• language-provided data types • standard data types • database management system (DBMS)-specific data types.
Integrated development environment may include:	• Borland C++ • Code Warrior • Eclipse • Visual C++ • Visual Studio suite.
Organisational guidelines may include:	• communication methods • content of emails • dispute resolution • document procedures and templates • downloading information and accessing particular websites • financial control mechanisms • opening mail with attachments • personal use of emails and internet access • virus risk.
Coding standard may include:	• ANSI C • GNU.
Documentation may follow:	• audit trails • International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), and Australian Standards (AS) standards • naming standards • project-management templates • report writing • version control.
Language may include:	• C • C++ • Java • Small Talk • VB • VB.NET.

Unit Sector(s)

Programming and software development