education

Department:
Education
REPUBLIC OF SOUTH AFRICA

NATIONAL CERTIFICATE (VOCATIONAL)

SUBJECT GUIDELINES

PRINCIPLES OF COMPUTER PROGRAMMING

NQF Level 3

September 2007

PRINCIPLES OF COMPUTER PROGRAMMING – LEVEL 3

CONTENTS

INTRODUCTION

1 DURATION AND TUITION TIME

2 SUBJECT LEVEL FOCUS

3 ASSESSMENT REQUIREMENTS

3.1 Internal assessment

3.2 External assessment

4 WEIGHTED VALUES OF TOPICS

5 CALCULATION OF FINAL MARK

6 PASS REQUIREMENTS

7 SUBJECT AND LEARNING OUTCOMES

7.1 Principles of electronic logic for computing

7.2 Program development environments

7.3 Data Structures

7.4 Program Design

7.5 Database application development

7.6 Error handling in a computer programming environment

7.7 User interface and output design

8 RESOURCE NEEDS FOR THE TEACHING OF PRINCIPLES OF COMPUTER PROGRAMMING - LEVEL 3

8.1 Physical resources

8.2 Human resources

8.3 Other resources

INTRODUCTION

A. What is Principles of Computer Programming?

Computer programming underpins the development of computer solutions to problems.

B. Why is Principles of Computer Programming important in the Information Technology programme?

Principles of Computer Programming is important in the Information Technology programme as it enhances the development of the IT environment. The subject will enable students to understand the principles of programming through the use of a current programming language. It will provide students with an opportunity to design and program well tested and user friendly computer based solutions to meet specific requirements.

C. The link between the Learning Outcomes for Principles of Computer Programming and the Critical and Developmental Outcomes

The student will be able to identify and solve problems, and collect, analyse, organise and critically evaluate information that is related to computer programming. The student will also demonstrate an understanding of the world as a set of related systems by recognising that problem-solving contexts do not exist in isolation.

D. Factors that contribute to achieving Principles of Computer Programming Learning Outcomes

- Analytical and logical ability
- Keen powers of observation
- Transferring of skills from familiar to unfamiliar situations
- Meticulous attention to detail
- Interest in computers and related topics

1 DURATION AND TUITION TIME

This is a one year instructional programme comprising 200 teaching and learning hours. The subject may be offered on a part-time basis provided the candidate meets all of the assessment requirements.

Course preparation should take consideration of students with special education needs.

2 SUBJECT LEVEL FOCUS

- Utilise computer-programming skills.

3 ASSESSMENT REQUIREMENTS

3.1 Internal assessment (50 percent)

3.1.1 Theoretical Component

The theoretical component will form 40 percent part of internal assessment.

Internal assessment of the theoretical component of Principles of Computer Programming Level 3 will take the form of observation, class questions, group work, (informal group competitions with rewards), individual discussions with Students, Class, topics and semester tests, internal examinations. Daily observation can be done when marking exercises of the previous day and class questions.

Assignments, case studies and tests can be done at the end of a topic. Tests and internal examinations must form part of internal assessment.

3.1.2 Practical/Application Component

Practical components include applications and exercises. All practical components must be indicated in a Portfolio of Evidence (PoE)

The practical component will form 60 percent part of internal assessment.

Internal assessment of the practical component of Principles of Computer Programming Level 3 will take the form of assignments, practical exercises, case studies, practical examination in a simulated business environment.

Students may complete practical exercises on a daily basis. Assignments and case studies can be done at the end of a topic. Practical examination can form part of internal assessment.

- **Some examples of practical assessments include, but are not limited to:**
  - Presentations (lectures, demonstrations, group discussions and activities, practical work, observation, role play, self activity, judging and evaluation)
  - Use of aids
  - Exhibitions
  - Visits
  - Guest speaker presentations
  - Research
  - Task performance in a simulated/structured environment

- **Definition of the term "Structured Environment"**

"Structured environment" for the purposes of assessment refers to an actual or simulated workplace, or workshop environment. It is advised that a practicum room is available on each campus for practical assessment.

- **Evidence in practical assessments**

All evidence pertaining to evaluation of practical work must be reflected in the students' Portfolio of Evidence. The tools and instruments constructed and used for the purpose of conducting such assessments must be clear from evidence contained in the PoE.

3.1.3 Processing of Internal assessment mark for the year

A year mark out of 100 is calculated by adding the marks of the theoretical component and the practical component of the internal continuous assessment.

3.1.4 Moderation of internal assessment mark

Internal assessment is subjected to both internal and external moderation procedures as contained in the *National Examinations Policy for FET College Programmes*.

3.2 External assessment (50 percent)

A national examination is conducted annually in October or November by means of a paper set externally and marked and moderated internally.

Details in respect of external assessment are contained in the *Assessment Guidelines: Principles of Computer Programming* (Level 3).

4 WEIGHTED VALUES OF TOPICS

	TOPICS	WEIGHTED VALUE
1.	Principles of electronic logic for computing	10%
2.	Program development environments	15%
3.	Data Structures	15%
4.	Program Design	15%
5.	Database Application Development	15%
6.	Error handling in a computer programming environment.	15%
7.	User interface and output design	15%
	TOTAL	**100**

5 CALCULATION OF FINAL MARK

Continuous assessment:	Student's mark/100 x 50/1 = a mark out of 50	(a)
Theoretical examination mark:	Student's mark/100 x 50/1= a mark out of 50	(b)
Final mark:	**(a) + (b) = a mark out of 100**	

All marks are systematically processed and accurately recorded to be available as hard copy evidence for, amongst others, purposes of moderation and verification, as well as purposes of reporting.

6 PASS REQUIREMENTS

The student must obtain at least fifty (50) percent in ICASS and fifty (50) percent in the examination.

7 SUBJECT AND LEARNING OUTCOMES

On completion of Principles of Computer Programming Level 3 the student should have covered the following topics:

Topic 1: Principles of electronic logic for computing.
Topic 2: Program development environments.
Topic 3: Data structures.
Topic 4: Program Design.
Topic 5: Database application development
Topic 6: Error handling in a computer environment.
Topic 7: User interface and output design

7.1 Topic 1: Principles of electronic logic for computing

7.1.1 Subject Outcome 1: Perform Boolean logic algebra operations

Learning Outcomes

The student should be able to:

- Demonstrate algebraic laws by using Boolean algebra.
- Explain and demonstrate how to translate Boolean expressions to logic diagrams and vice versa.
- Explain and demonstrate how to translate truth tables using Boolean expressions.
- Simplify Boolean expressions by using decision tables.
- Explain the output of a truth table and logic diagram for a given series of inputs.
- Describe a logic diagram for a simple truth table and simple written statement.

7.1.2 Subject Outcome 2: Demonstrate knowledge of logic gates.

Learning Outcomes

The student should be able to:

- State the symbol, truth table, logic diagram and Boolean expression for logic gates.
- Combine logic gates to perform the functions of the XOR gate.
- Identify the integrated circuits that implement gates for the computer systems.
- Explain the memory chips in terms of their use, functions, and operational characteristics.

7.1.3 Subject Outcome 3: Convert numbers using decoders and encoders

Learning Outcomes

The student should be able to:

- Demonstrate how to convert numbers between the combinations of decimal, binary and hexadecimal, using encoders.
- Demonstrate how to convert numbers between the combinations of decimal, binary and hexadecimal, using decoders.
- Add and subtract binary numbers.
- Apply two's complement as a data representation in the subtraction of binary numbers.

7.1.4 Subject Outcome 4: Demonstrate knowledge of the general principles of logic devices

Learning Outcomes

The student should be able to:

- Explain and draw symbols for logic devices according to industry conventions.
- Identify operations by describing different logic devices.

7.1.5 Subject Outcome 5: Demonstrate knowledge of integrated circuit specifications

Learning Outcomes

The student should be able to:

- Explain and identify the instruction sets for the integrated circuit.
- Explain and identify the method that a CPU uses for addressing of instructions and the method for addressing data.

7.2 Topic 2: Program development environments

7.2.1 Subject Outcome 1: Demonstrate an understanding of different data representations used in computer programs

Learning Outcomes

The student should be able to:

- Explain and apply different number conversion techniques between data types.
- Explain the purpose of a logical data type.
- Explain and differentiate between different internal representations of data types.

7.2.2 Subject Outcome 2: Demonstrate an understanding of high level programming language concepts

Learning Outcomes

The student should be able to:

- Explain constants and variables.
- Explain and illustrate the concepts of operators and expressions.
- Explain and illustrate different modular programming features and parameter passing.

7.2.3 Subject Outcome 3: Explain and use visual programming language concepts

Learning Outcomes:

The student should be able to:

- Explain the concept of rapid application development.
- Explain event driven programming.
- Explain and use visual programming language.
- Explain and illustrate rapid application development.
- Explain and implement event driven programming.

7.3 Topic 3: Data Structures

7.3.1 Subject Outcome 1: Demonstrate an understanding of computer data structures

Learning Outcomes

The student should be able to:

- Identify and describe the concept data structures.
- Identify different data structure types.
- Explain and illustrate features of the different data structures.
- Differentiate between types and uses of data structures.
- Explain how to use data structures when demonstrating simple searching techniques.
- Explain how to use data structures when demonstrating simple sorting techniques.
- Explain how to use data structures when demonstrating the manipulation of data.

7.4 Topic 4: Program Design

7.4.1 Subject Outcome 1: Analyse the given problem

Learning Outcomes

The student should be able to:

- Describe the given problem.
- Identify the program requirements
  Range: design specifications, IPO, data requirements.

7.4.2 Subject Outcome 2: Design and code a computer program to solve a given problem

Learning Outcomes

The student should be able to:

- Identify an appropriate programming language to solve a given problem.
- Implement the appropriate techniques and data structures to solve the given problem.
- Develop an algorithm for the given problem.
- Implement appropriate tools, techniques and data structures in coding the solution to the problem.
  Range: sequential, selection structures, iteration structures and appropriate techniques such as searching and sorting and data manipulation techniques
- Code the solution to the problem.
- Test and debug the solution to the problem.

7.5 Topic 5: Database application development

7.5.1 Subject Outcome 1: Plan and design a database to provide a solution to a given problem

Learning Outcomes

The student should be able to:

- Develop a working plan to meet the requirements of a supplied brief.
- Explain the purpose of the database.
- Outline the database specifications and/or features required to provide a solution.

7.5.2 Subject Outcome 2: Create a database according to the design using a database package

Learning Outcomes

The student should be able to:

- Create appropriate tables.
- Set up appropriate keys.
- Set up a relationship between two tables.
  Range: 1:1, 1:Many

7.5.3 Subject Outcome 3: Create database forms

Learning Outcomes

The student should be able to:

- Create applicable forms for data input and manipulation.
  Range: simple forms, master-detail, Sub-forms
- Modify the form design.
  Range: header, footer, layout, fields, components
- Capture data into the relevant tables using the developed form.

7.5.4 Subject Outcome 4: Use different methods to create different queries.

Learning Outcomes

The student should be able to:

- Create a query with a query wizard
- Create a query using simple SQL statements written by the developers
- Save the query
- Execute a query
- Edit a query
- Provide the query results in different forms

7.5.5 Subject Outcome 5: Retrieve information from a database by applying a filter

Learning Outcomes

The student should be able to:

- Create a filter
- Apply a filter to the database to filter out specific records
- Demonstrate how to remove a filter

7.5.6 Subject Outcome 6: Create a report

Learning Outcomes

The student should be able to:

- Design a report to address the requirements of a given problem.
- Create a report according to the design.
- Modify a report.
- Group data within a report.
- Save a report.
- Edit a report.

7.5.7 Subject Outcome 7: Perform advanced print options for a database

Learning Outcomes

The student should be able to:

- Print a database form.
- Print results of a query.
- Preview a database report to ensure the presentation meets the given specification.
- Preview a database report to ensure the presentation meets the given specification.
- Print a database report.

7.6 Topic 6: Error handling in a computer programming environment

7.6.1 Subject Outcome 1: Explain different errors and apply debugging techniques

Learning Outcomes

The student should be able to:

- Explain the difference between a logical error and a syntax error.
- Explain and apply different debugging techniques.

  Range: trace tables, compiler debugging tools, watches, stop breaks, control stops

7.6.2 Subject Outcome 2: explain and apply the concept of data validation and data validation techniques.

Learning Outcomes

The student should be able to:

- Iedntify and explain different input errors.
- Identify and explain sources of induced errors in calculations.
- Explain data validation techniques to limit input errors.
- Implement data validation techniques to limit input errors.

7.6.3 Subject Outcome 3: Demonstrate how calculation errors are induced in the computer.

Learning Outcomes

The student should be able to:

- Explain and demonstrate overflow errors.
- Explain and demonstrate underflow errors.
- Explain and demonstrate conversion errors.
- Explain and demonstrate errors found in computers as a result of advances in processor word sizes.

7.7 Topic 7: User interface and output design

7.7.1 Subject Outcome 1: Explain and implement user interface and output design concepts

Learning Outcomes

The student should be able to:

- Explain user interface design concepts and principles.
- List the guidelines for user interface design.
- Describe various user interface techniques.
- Discuss input design concepts, techniques and methods.
- Discuss output design issues and various types of output.

7.7.2 Subject Outcome 2: Explain and apply defensive programming

Learning Outcomes

The student should be able to:

- Develop a solution implementing user interface and output design principles.
- Explain defensive programming.
- Anticipate typical human behaviour that necessitates defensive programming.
- Apply different defensive programming techniques.

8 RESOURCE NEEDS FOR THE TEACHING OF PRINCIPLES OF COMPUTER PROGRAMMING - LEVEL 3

8.1 Physical resources

The following teaching aids should be made available, if possible:

- Lecture room
- Facilitator's computer
- Overhead projector
- Computer laboratory
- Computer per leaner linked to the internet
- Multimedia application software
- Programming software
- Animation software
- Activate IIS setting on computers
- Networked laser printer

8.2 Human resources

- The facilitator must have as a major subject Computer Programming at NQF Level 5.
- It will be to the advantage of facilitator if they have already been declared competent as assessor and/or moderator.
- Training in OBE

8.3 Other resources

- File per student for PoE
- 1 ream of paper per student
- 1 GB flash disk per student
- 1 DVD –RW per student
- 2 toners per LaserJet printer per year.
- Computer programming related magazines and journals.