basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

INFORMATION TECHNOLOGY

GUIDELINES FOR PRACTICAL ASSESSMENT TASKS

2012

These guidelines consist of 28 pages.

Information Technology

Practical Assessment Task (PAT)

2012

Guidelines to the teacher

This section consists of 7 pages.

A Introduction

The objective of the Practical Assessment Task (PAT) is to allow the teacher to directly and systematically observe and evaluate the applied competence of the learner. The PAT comprises the application of knowledge and skills.

In Information Technology the PAT counts 25% of the total promotion mark for the subject (i.e. 100 marks out of 400 marks). It is implemented across the first three terms of the Grade 12 school year and should be undertaken as one extended task, which is broken down into different phases or a series of smaller activities.

The IT PAT comprises three phases:

- Investigation and analysis
- Design
- Coding, implementation and internal documentation, including presentation and general evaluation

B Programming Project

1. What is the programming project about?

The programming project of the PAT in Information Technology requires the learners to use the development tools they have studied to develop a software solution for a particular problem within a given scenario.

2. Planning for the programming project

2.1 Description

Scenario for the 2012 Grade 12 PAT:

The learner is expected to write a program that could be used as an information system by the staff and/or visitors at any nature-oriented tourist attraction in South Africa, for example:

- A nature reserve
- A game reserve
- An aquarium
- A bird park
- A zoological garden
- Any other wildlife or nature-oriented tourist attraction in South Africa

The focus of the system should be to provide nature-related information to staff and/or visitors.

NOTE: It must **NOT** be a booking system.

The learner has to do research on the information needs of the staff and/or visitors at the chosen tourist attraction. A functional and useful system must be designed that could be successfully implemented at the specific tourist attraction.

The learner may choose his/her own topic/application within the context of the given scenario.

In completing the project, the learner will apply the following skills:

- Investigation
- Analysis
- Design
- Software development
  - Programming skills using the programming language studied
  - Database development
  - Graphical user interface (GUI) design

Sections of the assessment tool will penalise projects that are not related to the given scenario.

2.2 Development phases of the project

The programming project will be completed in three phases, indicated in the following table:

PHASE	MARKS	%
Phase 1: Investigation and analysis	30	17,5%
Phase 2: Design	30	17,5%
Phase 3: Coding, implementation and internal documentation, including presentation and evaluation	110	65%
TOTAL	**170**	**100**

Documentation/Evidence of what the learner did during each phase of development must be submitted at specified intervals. It is vital that evidence is provided for all phases of the work. The documentation required for each of these phases is given in the learner section of this document.

Deadlines for handing in the final product of each phase will be set by the teacher, taking into account the moderation dates for the different phases. The products of each phase will be assessed and the marks will be recorded.

2.3 Requirements for the project

The learners should adhere to the following minimum criteria:

Phase 1: Investigation and analysis

- Description of the problem in his/her own words outlining the main aspects in one paragraph
- Investigation of the topic by:
  - Identifying the nature and scope of the problem and gathering facts
  - Obtaining information from a real-world situation in which the end-user software product is to be used
- Analysis of the problem: What are the requirements and what should be provided by the programming solution?

(See Learner Section 2 Phase 1 and Assessment Tool Phase 1)

Phase 2: Design

- Design a solution: How will the program/system meet the requirements? Provide a well-planned solution to the problem.

(See Learner Section 2 Phase 2 and Assessment Tool Phase 2)

Phase 3: Coding, implementation and internal documentation

- The project must include the major development tools, i.e. a database and programming language appropriately integrated. (Other applications could be integrated with these development tools.)
- Other aspects of the programming project that will be assessed include:
  - Programming style
  - Graphical user interface (GUI)
  - Use of human-computer interaction (HCI) and software engineering principles
  - Functionality of the program
  - Level of expert programming
  - Robustness of the program, including the use of defensive programming techniques
  - Whether the project matches the original aims and goals
- Internal documentation to explain sections of the program

Presentation and general evaluation

- Evaluate the following:
  - Time management of the learner – Did he/she meet all the deadlines?
  - The ability of the learner to explain randomly selected pieces of code
  - Appropriateness of the solution in the context of the scenario

(See Learner Section 2 Phase 3 and Assessment Tool Phase 3)

3. Instructions to the learners

See Learner Section of this document.

4. Resources

The learner will need the following resources to complete the project:

- Access to a computer with the following programs:
  - Programming language: Java or Delphi
  - Office applications, such as a word processor (e.g. MS Word) and a presentation package (e.g. MS PowerPoint)
  - Database software, such as MS Access or MySQL
- IDE (for Delphi it is part of the programming language but for Java you will need additional software, such as Netbeans)

5. Assessment of the PAT

The task should be completed under controlled conditions and facilitated and continuously monitored by the teacher.

See Assessment Tool Section for the assessment sheets for the different phases.

Teachers must ensure that learners receive the following documents at the beginning of their Grade 12 year:

- The 'Instructions to the learner'-section included in this document
- The assessment sheets for all the phases included in this document

Learners should be allowed to reflect on the marks they have obtained and to address mistakes they have made before continuing with the next phase. The marks initially allocated to the evaluation of a specific phase will NOT be revised after the reflection.

Learners will be required to demonstrate their system at the end of Phase 3. Teachers should evaluate the projects according to the assessment tool provided for Phase 3.

Learners will NOT be allowed to change the topics of their projects once Phases 1 and 2 have been completed and assessed. If a learner should decide to change his/her topic after Phase 1 or Phase 2 has been assessed, the learner has to redo the earlier phases for the new topic. In this case, the teacher will NOT re-assess the updated Phase 1 and/or Phase 2. The marks for the original phases must be recorded. However, Phase 3 will not be assessed unless Phases 1 and 2 reflect investigation that was done for the new topic.

Correlation between all the phases of a project should be strictly and continuously checked during assessment. Evidence of work done during previous phases must always be available during each assessment and moderation of a specific phase of development.

Guidelines for the demonstration and internal evaluation of the project:

- The teacher must schedule dates and times for demonstrations. Allow approximately 15 minutes per project for the demonstration as well as about 5 minutes for setting up the project and getting feedback from the teacher afterwards.
- The development of the project is a continuous process. The teacher should always look at the work that has been done in the previous phases when assessing a specific phase of development. The teacher should monitor the progress of the project closely in relation to the work that was done during the previous phases.
    - The requirements identified and set out in Phase 1 should be reflected in the design phase of the project, i.e. Phase 2.
    - The work done during the design phase, i.e. Phase 2, should be reflected in the coding and implementation phase of the project, i.e. Phase 3.
- The learner should have all previous documentation (Phase 1 and Phase 2 documentation) handy when the demonstration takes place as part of Phase 3.
- The demonstrations must be done electronically on the computer.
- The learner must run his/her computer program and show all the features of the program to the teacher for assessment.
- The teacher must ask the learner to perform test strategies to test all the facets of the program.
- The teacher can require of the learner to execute other additional test procedures to make sure that the entire program is working correctly.
- The teacher must use the mark sheet for Phase 3 to allocate marks during the demonstration.

- The teacher must **identify random pieces of programming code (excluding the 10% borrowed code) in the project.** Borrowed code is any programming code that is not supplied by the standard version of Java or Delphi and which the learner did not program him-/herself. **The learner must then explain the purpose and the working of the randomly selected code to the teacher**. This is done to ensure that learners do the coding themselves. A similar type of procedure will be followed during the external moderation. If the learner cannot explain the code used in the project, **no marks can be awarded to the learner for the project.**
- The learner must hand in the electronic copy of the project that was demonstrated. The teacher will use this copy to allocate any outstanding marks in order to finalise the mark.

NOTE: Once the product of a phase has been handed in and assessed, that phase will not be re-assessed.

6. Recording and reporting

For each phase the teacher will assess the phase, record the mark and give feedback to the learner.

The marks for the different phases are added together and converted to a mark out of 100 which will be the final mark.

7. Guidelines for managing the PAT

There are two ways to go about managing the PAT:

Option 1:

- The teacher could dedicate one or two periods per week to the PAT while continuing with normal teaching to complete the Grade 12 curriculum during the rest of the week.
- If he/she chooses this option, he/she should start with the PAT as soon as possible in the first term, completing one phase per term.

Option 2:

- The teacher could dedicate a continuous period of time to the PAT, e.g. the last part of each term, completing one phase per term or by first completing the Grade 12 curriculum and then use a continuous period in the third term.

The teacher will have to plan his/her work schedule according to the option that he/she prefers.

It is suggested that the teacher documents the learners' topics when they start with Phase 1 to avoid 'instant projects' that might occur and could possibly not be the learner's own work. Teachers should also ensure that learners 'register' projects which they are capable of completing, in order to prevent topic changes due to the initial topic being too difficult or time-consuming.

8. HINTS

- Before learners start with Phase 1, first explain the PAT and provide an overview of the process of development to the learners.
- Discuss the task and the topic with the learners. Allow them to ask questions and ensure that they clearly understand the problem to be solved.
- Discuss examples of possible applications within the given scenario with the learners. Let the learners come up with some ideas and discuss the appropriateness thereof.

- Although a different scenario was used, it might be useful to let learners see both good and bad examples of PATs from previous years.
- Be very strict with the handing in of the documentation required in each phase.

Information Technology

Practical Assessment Task (PAT)

2012

Guidelines to the learner

This section consists of 10 pages.

Information Technology – Practical Assessment Task (PAT)

The IT PAT comprises three phases:

- Investigation and analysis
- Design
  - Coding, implementation and internal documentation, including presentation and general evaluation

1. Planning for the programming project

1.1 Description

Scenario for the 2012 Grade 12 PAT:

Youhave to write a program that could be used as an information system by the staff and/or visitors at any nature-oriented tourist attraction in South Africa, for example:

- A nature reserve
- A game reserve
- An aquarium
- A bird park
- A zoological garden
- Any other wildlife or nature-oriented tourist attraction in South Africa

The focus of the system should be to provide nature-related information to staff and/or visitors.

NOTE: It must **NOT** be a booking system.

You have to do research on the information needs of the staff and/or visitors at the chosen tourist attraction. A functional and useful system must be designed that could be successfully implemented at the specific tourist attraction.

You may choose your own topic/application within the context of the given scenario.

In completing the project, you will apply the following skills:

- Investigation
- Analysis
- Design
- Software development
  - Programming skills using the programming language studied
  - Database development
  - Graphical user interface (GUI) design

Sections of the assessment tool will penalise projects that are not related to the given scenario.

NOTE: Your final program must comprise ONE single, logically related piece of software. Projects which consist of two or more unrelated programs will only obtain marks for one of the parts since only one of the programs will be regarded as the actual project.

1.2 Development phases of the project

The programming project will be completed in three phases, indicated in the following table:

PHASE		MARKS	%
Phase 1:	Investigation and analysis	30	17,5%
Phase 2:	Design	30	17,5%
Phase 3:	Coding, implementation and internal documentation, including presentation and evaluation	110	65%
	TOTAL	**170**	**100**

Documentation/Evidence of what you did during each phase of development must be submitted at specified intervals. The evidence and output for each of the phases are discussed below. Marks can only be awarded if you supply this evidence to your teacher **and** if the work to be assessed is in relation to what has been done during previous phases.

Dates for submitting the documentation/evidence of the work done during each phase will be set by the teacher.

Study the assessment tools beforehand to make sure that you address all the relevant requirements according to the assessment tools.

Consider the feedback from the teacher indicated on the assessment tools and improve your work for the next phase accordingly. In a number of instances, marks are awarded for correcting work done incorrectly in previous phases.

All the documentation of the previous phases must be available to the teacher during each assessment.

1.3 Resources required for the project

You will need the following resources to complete the project:

- Access to a computer with the following programs:
  - Programming language: Java or Delphi
  - Office applications, such as a word processor (e.g. MS Word) and a presentation package (e.g. MS PowerPoint)
  - Database software, such as MS Access or MySQL
  - IDE (for Delphi it is part of the programming language but for Java you will need additional software, such as Netbeans)

The project must be completed under **controlled conditions** and facilitated and continuously monitored by the teacher.

You need to adhere to the following minimum criteria:

Phase 1: Investigation and analysis

- Description of the problem in your own words outlining the main aspects in one paragraph
- Investigation of the topic by:
  - Identifying the nature and scope of the problem and gathering facts from potential users, including their needs and any limitations they might have
  - Obtaining information from a real-world situation in which the end-user software product is to be used.

- Analysis of the problem: What are the requirements and what should be provided by the programming solution?

(See the following section Phase 1 and Assessment Tool Phase 1)

Phase 2: Design

- Design a solution: How will the program/system meet the requirements? Provide a well-planned solution in terms of:
  - Input, processing and output
  - Structure and contents of the database
  - GUI and the flow of events

  (See the following section Phase 2 and Assessment Tool Phase 2)

Phase 3: Coding, implementation and internal documentation

- The project must include the major development tools, i.e. a database design and programming appropriately integrated. (Other applications could be integrated with these development tools.)
- Other aspects of the programming project that will be assessed include:
  - Programming style
  - Graphical user interface (GUI)
  - Use of human-computer interaction (HCI) principles
  - Functionality of the program
  - Level of expert programming
  - Robustness of the program, including the use of defensive programming techniques
  - Whether the project matches the original aims and goals
- Internal documentation to explain sections of the program

Presentation and general evaluation

- A printout of the source code, including comments (internal documentation)
- Demonstration and debriefing (by the teacher) of the final product
- The teacher will evaluate the following:
  - Your time management – Did you meet all the deadlines?
  - Your ability to explain randomly selected pieces of code
  - The appropriateness of the solution in the context of the scenario

(See the following section Phase 3 and Assessment Tool Phase 3)

2. Instructions for the phases of the programming project

The instructions for the different phases are as follows:

PHASE 1:

Investigation and Analysis Due date: __________

In completing this phase you need to find some background information on your topic and determine *what* the program/system should do and provide:

1. Problem Statement

   - Identify and describe/explain in your own words the task and the problem to be solved. This description should not be a description of any computer code or the solution. All that is required is a description of the problem that you are investigating in the real-world context or situation where it has been identified.
   - Your statement should:
     - Clearly state what the problem entails (provide some brief background information, i.e. why you choose the topic)
     - State what you are trying to do/solve and for whom
     - Outline the aspects that should be solved/expectations of the software
     - Indicate what the purpose of the software will be

2. Investigation

   - Investigate the topic to gather some facts and to determine the nature and scope of the problem, e.g. what will be part of the software and what it will not include. The aim is to determine requirements, needs and limitations.
   - You might want to find out the following:
     - What systems or processes are used in the absence of software (e.g. paper-based solutions) and what are the limitations/challenges with these
     - The key areas to be addressed/solved
     - Information specific to your topic for which you will need to construct a solution (e.g. what an information kiosk is, what information people normally want from such a kiosk, what data you would need, etc.) and how or where/from whom you will find/obtain this information
   - Demographics of users of your system (age, gender, disabilities, literacy level, etc.)
   - You must keep a comprehensive reference list of all resources (websites, books) used.
   - Evidence of investigation: Well-formatted and presented summary of all the findings

3. Determine requirements for the program

   - Conduct discussions with end-users, make notes, gather documents and compile summaries and draw conclusions from this information.
   - Observe how the system or a similar system currently works. If possible, make notes.
   - Make notes on input, processing and output that will be required.
   - Possible evidence to be submitted: Questionnaires, notes (or recordings) made during interviews, photographs, documents/forms from current system, audio/video recordings from interviews/how the system currently works, etc.
   - Information must be supplied on details of the interviews, questionnaires, users, etc. Information such as the time and date the questionnaire/interview was conducted, as well as the names, location and contact details of participants. Your teacher may use this information to verify that the investigation has been done in the way indicated.

4. Possible solution
   - Using your investigation and responses from users, write a possible/intended brief solution for the problem in your own words.
   - Possible evidence to be submitted: Problem definition and/or requirements list. This is not a detailed specification with input and output but rather a simple list of features that the intended solution will provide to users.

5. Submit a planning document

   Combine the results of your investigation in a planning document consisting of:

   - Description of the problem
   - Background information on the topic with references
   - Evidence of information gathered
   - Broad overview of a possible solution to the problem

PHASE 2:

Design Due date: __________

In completing this phase, you need to determine the specifications for the program/system and *how* the program/system will accomplish the goals set during analysis. **Study the assessment tool for Phases 2 and 3 to make sure that your project meets the requirements that all tasks must adhere to.**

1. Convert requirements/features into specifications:

   Specify the solution in terms of:

   - Input
   - Processing
   - Output (tables, graphs, currency, units of measurement, etc. included)

   Include the following as part of the specifications:

   - Defensive programming techniques such as:
     - Data validation (indicate where and how it should be applied)
     - Error messages
   - Data structures to organise and store data:
     - Database design:
       - Fields (types, names and sizes)
       - Relationships
       - Keys (primary and foreign)
       - ER diagrams
     - Data types and structures used for programming:
       - Primitive types – single value, e.g. integer, real/double
       - Structured data types – collections of data, e.g. arrays, matrices, records, sets, combo boxes, lists, vectors
       - Abstract data types – collections with set of data and set of operations that can be performed on data, e.g. classes and objects
       - Design of classes – methods with parameters and return data types, attributes, constructors, class diagrams, inheritance hierarchies if applicable (higher order).
   - Graphical user interface (GUI) design:

     Drawings and/or screenshots of GUI indicating the following:
     - Layout of components
     - Navigation (e.g. How does the user get from one screen to another and back?)
     - Flow of events (from one screen to another, from one event to another)
     - Data flow between units/modules (What data is passed between modules?)
     - Justification for use of input and output components (e.g. Why was a combo box used instead of a text field, or why was a check box used instead of a radio button?)

2. Submit a document with all the program specifications as listed above. The document must include the following:

   - Input, processing and output requirements
   - Error checking and validation procedures
   - Database design
   - List of data structures used (including class diagrams)
   - Design and layout of the GUI

   The document could include IPO tables, flow charts, TOE charts and diagrams such as ER diagrams, class diagrams, use case diagrams and/or screen dumps or any other appropriate tools that will give a clear representation of the system, including required notes, as well as descriptions of specifications.

PHASE 3:

Coding and Implementation Due date: __________

In completing this phase, you will have to code the solution including the GUI, as planned in the previous phase, create the data structures, debug and test the program. **Study the assessment tool for Phase 3 to make sure that your project meets the requirements that all tasks must adhere to.**

Suggested steps to complete this phase:

1. Break up the solution, as outlined in Phase 2, into modules (e.g. according to options).

2. Code/Create the GUI.

3. Create the data structures to organise and store data.

4. Coding – Code the solution according to requirements and specifications (input, processing and output) compiled in the previous phase. Among other things, marks will be awarded for the following:

   - Appropriate input strategies
   - Database access
   - Appropriate and effective algorithms
   - Good programming principles: Code re-use, variable names, commenting
   - Parameter passing: Independence of modules

5. Error handling – Ensure that input is validated and that exception handling is applied, where required.

6. Testing/Debugging – Ensure that the program runs correctly and is error free.

   - Apply a test strategy to ensure that the program/system does what it should by using different ranges of data including extreme/erroneous test data. Keep track of this data as it will be needed for your documentation.

7. Internal documentation

8. Evaluate your program/system – Does the program do what it should/meet the requirements? Did you apply good programming principles?

   - Review program code. Have good programming principles been applied? Study the assessment tool for Phase 3 intensively.
   - Does the program/system operate properly?
   - Does the program comply with what was stated in Phases 1 and 2?
   - Well-designed GUI?

   NOTE: You are allowed to make use of borrowed code but it cannot exceed more than 10% of your programming code. Borrowed code is any programming code that is not supplied by the standard version of Java or Delphi and that you did not program yourself. This will typically be code to perform unusual functions, such as playing a video clip, etc. **Your program will not be accepted by the teacher if it exceeds the limit of 10% of borrowed code**.

9. Presentation and evaluation

1. Hand in:

- Electronic copy of programs and all applicable files, such as data files, etc.

2. Demonstrate the program for evaluation and debriefing

Guidelines for the demonstration of the project:

- The teacher will schedule dates and times for demonstrations. About 20 minutes per project will be allowed.
- You should hand in all the documentation before the demonstration takes place – at least ONE week in advance.
- The demonstrations must be done electronically on the computer.
- You must execute your computer program and show all the features of the program to the teacher for evaluation.
- The teacher can use a test strategy provided as a guideline and ask the learner to perform parts of or all of the test strategy.
- The teacher can require you to execute other additional test procedures to make sure that the entire program is working correctly.
- The teacher can use the mark sheet for Phase 3 as a guideline and allocate marks accordingly during the demonstration.
- As part of the demonstration, the teacher will identify random pieces of programming code (excluding the 10% borrowed code) in the project. You must then explain the purpose and working of the randomly selected code to the teacher. This is done to ensure that you did the coding yourself. A similar type of procedure will be followed during the external moderation. If you cannot explain the code used in the project, no marks can be awarded for the project.
- You must hand in the electronic copy of the project that was demonstrated. The teacher will use this copy to allocate any outstanding marks in order to finalise the mark.

3. Final general evaluation

The teacher will evaluate the following:

- Time management – Did you meet all the deadlines?
- Utility value – Is your solution appropriate in the context of the given scenario?
- Do the different phases of development correlate and lead to the final solution as a continuous process?

Information Technology

Practical Assessment Task (PAT)

2012

Assessment tools

This section consists of 10 pages.

Assessment tools for the programming project

Assessment for Phase 1 **Name of learner:**__________________________

<table>
<tr><th colspan="16">Investigation and Analysis: Criteria</th><th>Possible Mark</th><th>Mark Obtained</th></tr>
<tr><td rowspan="3">Problem statement</td><td colspan="15">The problem statement</td><td rowspan="3">4</td><td rowspan="3"></td></tr>
<tr><td colspan="3">4</td><td colspan="3">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="3">0</td></tr>
<tr><td colspan="3">The problem is clearly stated and described and unambiguous – clearly states what the problem entails;
Outlines what he/she are trying to do and the aspects that should be solved;
Clear statement of what the purpose of the software will be</td><td colspan="3">The statement is clear but with minor shortcomings.</td><td colspan="3">The statement is vague, leaving the reader unsure of what the purpose of the system will be.</td><td colspan="3">The statement is so vague that no discernable purpose can be found.</td><td colspan="3">No statement or description</td></tr>
<tr><td rowspan="9">Investigation</td><td colspan="15">Current systems/processes</td><td rowspan="3">2</td><td rowspan="3"></td></tr>
<tr><td colspan="5">2</td><td colspan="5">1</td><td colspan="5">0</td></tr>
<tr><td colspan="5">Clear description of current systems/processes, including limitations and challenges and how the program could address these</td><td colspan="5">Minor shortcomings;
Description not always clear</td><td colspan="5">No description of current systems/processes or limitations and challenges</td></tr>
<tr><td colspan="15">Key areas pertaining to the topic</td><td rowspan="3">4</td><td rowspan="3"></td></tr>
<tr><td colspan="3">4</td><td colspan="3">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="3">0</td></tr>
<tr><td colspan="3">Investigation clearly and comprehensively defines/explains key areas pertaining to the topic;
Shows good insight and understanding into all key areas of the topic</td><td colspan="3">Minor shortcomings;
Shows insight in most of the key areas of the topic</td><td colspan="3">Shows some insight in some of the key areas</td><td colspan="3">Vague;
Shows little understanding of key areas</td><td colspan="3">Key areas not defined</td></tr>
<tr><td colspan="15">List of references</td><td rowspan="3">2</td><td rowspan="3"></td></tr>
<tr><td colspan="5">2</td><td colspan="5">1</td><td colspan="5">0</td></tr>
<tr><td colspan="5">Comprehensive list;
In acceptable format</td><td colspan="5">Less comprehensive – minor shortcomings</td><td colspan="5">No references</td></tr>
</table>

<table>
<tr><td rowspan="9">Requirements</td><td colspan="10">Input requirements</td><td rowspan="3">3</td><td rowspan="3"></td></tr>
<tr><td colspan="2">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="2">0</td></tr>
<tr><td colspan="2">Comprehensive list of input requirements obtained from users, well defined and obtained using an appropriate method such as questionnaire, interview</td><td colspan="3">Minor shortcomings;
Less comprehensive list of input requirements obtained from users;
Not that clearly defined but still acceptable</td><td colspan="3">Few input requirements obtained from users or requirements made up by learner;
Vague and not clearly defined</td><td colspan="2">No input requirements obtained</td></tr>
<tr><td colspan="10">Processing requirements</td><td rowspan="3">3</td><td rowspan="3"></td></tr>
<tr><td colspan="2">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="2">0</td></tr>
<tr><td colspan="2">Comprehensive list of processing requirements obtained from users, well defined and obtained using an appropriate method such as a questionnaire, interview</td><td colspan="3">Less comprehensive list of processing requirements obtained from users using at least one method;
Not that clearly defined but still acceptable</td><td colspan="3">Few processing requirements obtained from users or requirements made up by learner;
Vague and not clearly defined</td><td colspan="2">No processing requirements obtained</td></tr>
<tr><td colspan="10">Output requirements</td><td rowspan="3">3</td><td rowspan="3"></td></tr>
<tr><td colspan="2">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="2">0</td></tr>
<tr><td colspan="2">Comprehensive list of output requirements obtained from users, well defined and obtained using an appropriate method such as questionnaire, interview</td><td colspan="3">Less comprehensive list of output requirements obtained from users using at least one method;
Not that clearly defined but still acceptable</td><td colspan="3">Few output requirements obtained from users or requirements made up by learner;
Vague and not clearly defined</td><td colspan="2">No output requirements obtained</td></tr>
<tr><td rowspan="3">Possible solution</td><td colspan="10">Possible solution</td><td rowspan="3">4</td><td rowspan="3"></td></tr>
<tr><td>4</td><td colspan="3">3</td><td colspan="2">2</td><td colspan="3">1</td><td>0</td></tr>
<tr><td>Clear and comprehensive description – a clear overview of a possible solution is given;
Clearly indicates what the software will include and do</td><td colspan="3">Clear but with minor shortcomings</td><td colspan="2">Basic description is given but some aspects of the suggested solution are vague</td><td colspan="3">The description is vague;
No clear solution identified in the description</td><td>No possible solution given</td></tr>
<tr><td rowspan="6">General</td><td colspan="10">The scope (boundaries and assumptions/features that will be included)</td><td rowspan="3">3</td><td rowspan="3"></td></tr>
<tr><td colspan="2">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="2">0</td></tr>
<tr><td colspan="2">The scope of the suggested solution is clear and well defined;
Clearly indicates what will be part of the program and what it will not include</td><td colspan="3">Defined but some minor shortcomings;
Not always clear on what the program will include and not include</td><td colspan="3">Vaguely defined;
No clear scope can be determined</td><td colspan="2">Not defined</td></tr>
<tr><td colspan="10">Appropriateness of the suggested solution in the context of the scenario</td><td rowspan="3">2</td><td rowspan="3"></td></tr>
<tr><td colspan="3">2</td><td colspan="4">1</td><td colspan="3">0</td></tr>
<tr><td colspan="3">Most appropriate;
Good application for scenario</td><td colspan="4">Appropriate;
Application in scenario not always convincing</td><td colspan="3">Not appropriate;
Application in scenario not convincing</td></tr>
<tr><td colspan="11">Total:</td><td>30</td><td></td></tr>
</table>

Assessment for Phase 2: **Name of learner:** ___________________________________

Design: Criteria					Possible Mark	Mark Obtained
Specifications	Input				3	
	3	2	1	0		
	User input and other sources of input clearly described in terms of what input is required and the format (e.g. date format) of the input	Described with minor shortcomings in terms of what is required and format	Description of input vague or incomplete	Only listed – no description		
	Processing				3	
	3	2	1	0		
	How data will be processed/manipulated clearly described in terms of requirements, format, calculations, formulas, etc.; Short, clear and correct description in all instances where applicable; User clear on result	Description of processing/ manipulation of data done/clear in most instances where required; Good effort, but can improve; Not always correct or applicable	Description of processing/ manipulation of data not clear in most instances; Not done in all instances; Descriptions vague/ incomplete; Not correct/applicable in most instances	Processing/Manipulation of data not described		
	Output				3	
	3	2	1	0		
	Identified and clearly described required output (screen, reports) according to requirements and in terms of format (currency, units of measurement, etc.)	Identified and described with minor shortcomings	The description is vague or most required output not identified or format of output not described	Output not identified and described		
	Data validation				3	
	3	2	1	0		
	Indicated for all input and described in detail; Meaningful and effective	Identified and described with minor shortcomings	Indicated in some cases where applicable and/or description of validation vague or incomplete	Not indicated		
	Error messages associated with data validation				2	
	2	1	0			
	Indicated for all applicable validation checks/errors and described/shown in detail	Indicated some error messages and/or description of error messages vague	No error message indicated for any data validation or error			

<table>
<tr><td rowspan="6">Database design overview</td><td colspan="12">Fields</td><td rowspan="3">2</td><td rowspan="3"></td></tr>
<tr><td colspan="4">2</td><td colspan="4">1</td><td colspan="4">0</td></tr>
<tr><td colspan="4">Well-chosen fields, field types and sizes to suit the application;
Field properties such as validation and masks indicated where applicable</td><td colspan="4">Mostly well-chosen fields, field types and sizes to suit the application;
Field properties such as validation and masks not always indicated where applicable</td><td colspan="4">Not well-chosen fields, field types and sizes to suit the application;
Field properties such as validation and masks not indicated</td></tr>
<tr><td colspan="12">Tables</td><td rowspan="3">3</td><td rowspan="3"></td></tr>
<tr><td colspan="3">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="3">0</td></tr>
<tr><td colspan="3">Fields well grouped into tables;
No repetition of fields in tables;
More than two tables with correct relationships between tables;
Shows correct primary and foreign keys</td><td colspan="3">Fields well grouped into tables with minor shortcomings;
Some repetition of fields in tables;
At least two tables with correct relationship between tables;
Shows correct primary and foreign keys</td><td colspan="3">Fields mostly well grouped into tables;
Extensive repetition of fields in tables;
At least two tables but no/incorrect relationship between tables;
Shows no/incorrect primary and foreign keys</td><td colspan="3">Only one table/only the fields listed</td></tr>
<tr><td rowspan="3">Data structures</td><td colspan="12">Organise and store data (collections with set of data and set of operations that can be performed on data, e.g. classes and objects)</td><td rowspan="3">3</td><td rowspan="3"></td></tr>
<tr><td colspan="3">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="3">0</td></tr>
<tr><td colspan="3">Used abstract data types;
Application well planned in terms of classes and objects;
Object well structured with relevant methods</td><td colspan="3">Used abstract data types – room for improvement;
Application not entirely planned in terms of classes and objects - not always applicable;
Entire program not object-oriented</td><td colspan="3">An effort to use abstract data types;
Objects not always well compiled;
Objects compiled but not used correctly/not used at all;
Very little of program is object-oriented</td><td colspan="3">No abstract data types (classes and objects)</td></tr>
<tr><td rowspan="6">GUI design</td><td colspan="12">Input</td><td rowspan="3">3</td><td rowspan="3"></td></tr>
<tr><td colspan="3">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="3">0</td></tr>
<tr><td colspan="3">Appropriate input components to support accurate/valid input in all instances where required;
Substantiated choices where required</td><td colspan="3">Appropriate components where required;
Choices mostly substantiated;
Minor shortcomings</td><td colspan="3">Appropriate in most instances where required;
Choices not always substantiated</td><td colspan="3">Not done/not appropriate input components identified</td></tr>
<tr><td colspan="12">Output</td><td rowspan="3">3</td><td rowspan="3"></td></tr>
<tr><td colspan="3">3</td><td colspan="3">2</td><td colspan="3">1</td><td colspan="3">0</td></tr>
<tr><td colspan="3">Appropriate output components identified in all instances where required</td><td colspan="3">Appropriate output components identified where required – minor shortcomings</td><td colspan="3">Appropriate in most instances where required;
Choices not always substantiated</td><td colspan="3">Not done/not appropriate output components identified</td></tr>
<tr><td rowspan="3">General</td><td colspan="12">Addresses the requirements specified in Phase 1</td><td rowspan="3">2</td><td rowspan="3"></td></tr>
<tr><td colspan="4">2</td><td colspan="4">1</td><td colspan="4">0</td></tr>
<tr><td colspan="4">All requirements addressed</td><td colspan="4">At least 50%</td><td colspan="4">Less than 50%</td></tr>
<tr><td colspan="13">Total:</td><td>30</td><td></td></tr>
</table>

Assessment for Phase 3: **Name of learner:**__________________________

NOTE: Phase 3 and 4 will only be assessed once Phases 1 and 2 have been completed AND Phases 1 and 2 are related to the topic covered in Phases 3 and 4.

	Coding and Implementation: Criteria					Possible Mark	Mark Obtained
Database	Database: tables					4	
	4	3	2	1	0		
	Number of tables speaks to an effective solution – tables normalised (appropriate number of tables to support effective solution); All the relationships well defined using the correct primary and foreign keys; Primary keys and foreign key in related tables effective and appropriate	Number of tables speaks to a good solution; At least one effective and appropriate relationship using the correct primary and foreign keys; Some of the primary keys or foreign key in related tables not appropriate	Number of tables indicates partial solution – tables not normalised (could have more tables for a more effective solution); Most of the primary keys or foreign keys not appropriate or only default keys used	Number of tables irrelevant to solution (only one table/a number of irrelevant tables with repetition of fields in tables; or more than one database with one table in each database instead of one database)	No database used		
	Field types					2	
	2	1	0				
	All fields with appropriate data types	Data types of some of the fields not appropriate	Only default types used or no database used				
	Field sizes – database					2	
	2	1	0				
	All fields with appropriate field sizes	Some of the fields are too large/ small	Only default field sizes used or no database used				
	Descriptive field names in database					1	
Programming solutions	NOTE: Assess the code if the program does not run. This section has to do with the programming in Delphi/Java excluding SQL.						
	User defined data structures (excluding database tables)					3	
	3	2	1	0			
	Used appropriate and most effective data structures to solve the problem in all instances	Used appropriate and most effective data structures in most instances or with minor shortcomings	Appropriate and most effective use of data structures in less than 50% of the instances	Inappropriate or ineffective use of data structures			
	Variables/data structure names meaningful					2	
	2	1	0				
	Variable/data structure names meaningful in all instances, throughout program	Minor shortcomings	Not meaningful				
	Selection structures					3	
	3	2	1	0			
	Used appropriate and most effective selection structures to solve the problem in all instances	Used appropriate and most effective selection structures in most instances	Inappropriate or ineffective use of selection structures in most instances	Not used			

	Repetition structures				3	
	3	2	1	0		
	Used appropriate and most effective repetition structures to solve the problem in all instances	Used appropriate and most effective repetition structures in most instances	Inappropriate or ineffective use of repetition structures in most instances	Not used		
	Complex programming techniques (e.g. play video clips, borrowed code, threads, networking)				3	
	3	2	1	0		
	It works correctly; Appropriately used and adds value to the solution	It works correctly; Not always appropriately used or does not really add value to the solution	Effort made but it does not work properly; Inappropriately used; Not relevant to the solution	No complex techniques used or exceeds more that 10% of code		
	Data flow and processes (user defined parameter passing)				3	
	3	2	1	0		
	Excellent interaction/ communication between modules/classes; Includes advanced data types as parameters or return types	Proficient/adequate/some communication between modules/classes with small flaws; Includes some parameter passing between modules	Limited communication between modules/classes; Only primitive data types passed as parameters	No communication between modules/classes; No parameters passed		
	Re-use of code (classes and methods/subprograms)				3	
	3	2	1	0		
	Appropriate and effective re-use of code and/or methods; Modules independent; Local variables used where applicable; Global variables only used when required	Re-use of code and/or methods but not always appropriate/effective; Modules not always independent; Some global variables restrict independence of modules	Re-use of code inappropriate/not effective; Modules could have been broken down into more modules; Almost all the variables declared globally	Linear programming – one continuous program, no modules; No re-use of code and/or methods		
	Solution algorithms				3	
	3	2	1	0		
	All solution algorithms used in solving problem are appropriate and effective	Appropriate and effective with minor shortcomings	Mostly inappropriate or not effective	Totally inappropriate solution algorithms or mostly ineffective		
	Correctness of solution algorithms				3	
	3	2	1	0		
	No logical error; All the results of processing correct	Minor shortcomings; Very few minor logical errors; Very few of the results not correct	Logical errors; Many results incorrect	Many logical errors; Almost all the results are incorrect/few of the required results are delivered		
Input	Input strategies				3	
	3	2	1	0		
	Most appropriate, effective input strategies used in all instances, e.g. input from text files, database tables, user input	Appropriate and effective with minor shortcomings	Some strategies could have been more appropriate/effective	Mostly inappropriate or not effective		

Output	Output vs. requirements				3	
	3	2	1	0		
	Output meets all the requirements for the solution	Output meets most of the requirements for the solution	Output meets less than 50% of the requirements of the solution	No output		
	Structures for output (coding)				3	
	3	2	1			
	Output always well structured, readable with headings and subheadings; Headings repeated on following page/screen where applicable	Output in most of the cases well structured, readable with headings and subheadings; Headings repeated on following page/screen in most of the cases where applicable	Output not well structured; Headings and/or subheadings in most of the cases not well formulated or absent; Headings mostly not repeated on following page where applicable	No code to display output		
	Format of output – all values formatted appropriately where applicable, e.g. currency, units of measurement, etc.				1	
Database interaction	Database connection – string/path set and work correctly				1	
	Database interaction with program				2	
	2	1	0			
	Database interacts with program in a meaningful way, e.g. queries and reports	Not always meaningful	No interaction/not meaningful			
	Manipulate records correctly via SQL (2 marks for each correct SQL statement to a maximum of 6 marks)				6	
	Insert, delete, select, update List other:					
	Manipulate fields via SQL (2 marks for each correct SQL statement to a maximum of 6 marks)				6	
	Calculations on fields, change contents, show only one field, named fields, sort according to fields List other:					
Error handling and testing	Program compiles successfully – no syntax errors				1	
	Output errors				3	
	3	2	1	0		
	No run-time errors. All the options are executed successfully.	Some of the options produce errors when executed.	Only one or two of the options can be executed successfully.	None of the options execute successfully.		
	Input validation				3	
	3	2	1	0		
	All input that should be validated is validated using code.	Most of the input that should be validated is validated.	Less than 50% of the input that should be validated is validated.	No validation		
	Error messages				3	
	3	2	1	0		
	Appropriate and user-friendly error messages in all cases where data validation is applied	Appropriate and user-friendly error messages in most of the cases where data validation is applied	Appropriate and user-friendly error messages in very few of the cases where data validation is applied	No error messages		
	Program gives output (output is the result of processing, i.e. GUI windows with no functionality do not classify as output)				2	
	Correctness of output				3	
	3	2	1	0		
	Program gives correct and appropriate output in all cases	Program gives correct and appropriate output in most cases	Program gives correct and appropriate output in less than 50% of the cases	No output or only incorrect output		
	Exception handling				2	
	2	1	0			
	Used try … catch exception handling	Attempted exception handling or used if statements to handle error cases	No attempt			

Graphical user interface (GUI)	Different screens used (windows/panels/tab sheets, etc.) appropriately			1	
	Components used for input/output (implemented as per Phase 2 OR appropriate changes made from Phase 2 feedback)			2	
	2	1	0		
	Always appropriate, most effective	Minor shortcomings	Mostly not appropriate/effective		
	Labels/prompting with exact formats for input			2	
	2	1	0		
	Applied constantly throughout the entire project where required	Applied in most of the cases where required	Mostly not applied		
	Consistent layout (same look and feel throughout)			2	
	2	1	0		
	For all screens (same colours, fonts used throughout program)	Most of the screens (some screens different colours, fonts, etc.) – minor shortcomings	Layout mostly inconsistent		
	Informative output/reports are informative and easy to read and interpret			2	
	2	1	0		
	All output, all screens are informative and easy to read and interpret (appropriate font size, layout, colour, etc.)	Most of the output are informative and easy to read and interpret – minor shortcomings	Output mostly not informative or not easy to read and interpret		
	Grouping of input/output			2	
	2	1	0		
	Type of input/output grouped together, e.g. address information, for all screens	Type of input/output mostly grouped together – minor shortcomings	Not grouped together in most instances		
	Navigation between screens			2	
	2	1	0		
	Easy to navigate between screens – logical flow of events	Easy and logical with minor shortcomings	Not easy or no logical flow		
	Help available as part of the program			2	
	2	1	0		
	Help available as part of the program, works correctly with clear instructions	Not always available, clear or does not always work correctly	Not available		
	Context sensitive			2	
	2	1	0		
	Context sensitive help available (tool tip text included), appropriate and effective	Not always appropriate or effective	Not available		
	Design vs. target user			2	
	2	1	0		
	Design considers target user (age, literacy level, visual impairments, appropriateness of images, etc.); Design appropriate for target user	Consideration given, but minor shortcomings	Totally inappropriate		

Documentation	Internal documentation					2	
	2	1	0				
	Code is clearly commented/annotated to explain/describe for easy interpretation throughout the program	Commented/annotated but not throughout the program	Not commented/annotated				
	Separation of sections					2	
	2	1	0				
	Sections in the code of the program clearly separated to enhance readability (spacing, comments for method/subsection, etc)	Some sections separated	Not separated				
General evaluation	Time management					3	
	3	2	1	0			
	All deadlines met – all 3 phases and all the required work was done	Met 2 deadlines or submitted on time but some of the work was not done	Met at least one deadline or submitted on time but most of the work was not done	Always late, never done			
	Ability to explain purpose of working of randomly selected code					4	
	4	3	2	1	0		
	Explained all selected code clearly and with confidence; Shows excellent insight	Explained selected code with minor shortcomings; Shows insight	Explained some of the selected code; Shows some insight in some of the code	Not able to explain most of the selected code appropriately; Lacks insight	Unable to explain		
	Real-life application of system					3	
	3	2	1	0			
	The solution is a working system that can be applied in a real situation.	The solution is a system that can be applied in a real situation with some fine-tuning.	Some parts can be applied in the real situation.	Totally irrelevant; Will not work in a real situation			
					Total:	110	