basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

INFORMATION TECHNOLOGY

GUIDELINES FOR PRACTICAL ASSESSMENT TASKS

GRADE 12

2020

These guidelines consist of 32 pages.

TABLE OF CONTENTS

1. INTRODUCTION

The 17 Curriculum and Assessment Policy Statements subjects which contain a practical component all include a practical assessment task (PAT). These subjects are:

- AGRICULTURE: Agricultural Management Practices, Agricultural Technology
- ARTS: Dance Studies, Design, Dramatic Arts, Music, Visual Arts
- SCIENCES: Computer Applications Technology, Information Technology, Technical Sciences
- SERVICES: Consumer Studies, Hospitality Studies, Tourism
- TECHNOLOGY: Civil Technology, Electrical Technology, Mechanical Technology and Engineering Graphics and Design

A practical assessment task (PAT) mark is a compulsory component of the final promotion mark for all candidates offering subjects that have a practical component and counts 25% (100 marks) of the end-of-the-year examination mark. The PAT is implemented across the first three terms of the school year. This is broken down into different phases or a series of smaller activities that make up the PAT. The PAT allows for learners to be assessed on a regular basis during the school year and it also allows for the assessment of skills that cannot be assessed in a written format, e.g. test or examination. It is therefore important that schools ensure that all learners complete the practical assessment tasks within the stipulated period to ensure that learners are resulted at the end of the school year. The planning and execution of the PAT differs from subject to subject.

2. GUIDELINES

2.1 What is the PAT?

The practical assessment task (PAT) is a software development project in which you will have the opportunity to demonstrate your software development and programming skills.

The purpose of the PAT is to:

- Work extensively with content knowledge to improve your programming and organisational skills;
- Implement higher-order and critical-thinking skills, formulate strategies and solve problems on different levels; and
- Develop good working practices to prepare you for the real world, such as:
  - Time management
  - Thorough planning
  - Perseverance to achieve and to excel in what you set out in your plan
  - Presentation and marketing of your product

You will need to demonstrate knowledge and understanding of the software development life cycle through analysis, design, coding and testing of your project. You will have to show effective use of the software design tools and techniques which you have studied.

The PAT is divided into TWO phases, as explained below.

Phase 1: Outlines the project task, solution and a possible design of the project

Phase 2: A working, fully documented Delphi application that implements the planned solution

NOTE: Submission dates – Specific dates will be determined by your subject teacher.

Phase 1: No later than ONE week before the end of Term 1
Phase 2: No later than the FIRST week of Term 3

LEARNERS MUST ADHERE TO THE DUE DATES FOR EACH PHASE.

NOTE: You will be required to demonstrate and discuss your application during an interview session.

2.2 Mark allocation

The PAT counts 25% of your final examination mark for Information Technology. It is therefore crucial that you strive to produce work of a high standard.

PHASE	DEVELOPMENT PHASE	MAXIMUM MARK
Phase 1	Analysis and Design	48
Phase 2	Coding and Testing	78
General	Final Product and Impression	14
	TOTAL	**140**

NOTE:

- The PAT mark is a compulsory component of the final certification mark for all candidates registered for Information Technology.
- Your PAT will be externally moderated by subject experts and quality assured by Umalusi.

2.3 The topic

Tourism is an important industry in South Africa. It includes a wide range of aspects, such as organising tours, marketing tourist attractions, arranging transport and accommodation, etc. Software applications can be useful in assisting with activities within the tourism industry.

Projects in the scenario of tourism could include the following:

- Travel arrangements to and from destinations/tourist attractions, e.g. booking flights, buses, trains
- Accommodation details, such as hotels, B&Bs, campsites, lodges, timeshare, etc.
- Travel agencies organising vacations for their clients, e.g. destinations, distances to travel, time schedules
- Providing information and/or booking tourist attractions, such as theme parks, aquariums, flea markets, etc.
- Providing information and/or booking activities and events scheduled, such as music and sport
- Booking tours/providing information on tours in different parts of the country/world
- Health and safety, such as public awareness and the availability of resources and emergency services, etc.
- Leisure and entertainment companies making translators, interpreters and communication facilities available to tourists
- Preparation and management of travel documents, e.g. ID, passport, unabridged birth certificate, visa, travel insurance
- Providing information and/or booking visits to heritage and cultural sites in South Africa/other countries
- Any other topic related to any aspect of the tourism industry

Choose an application/environment that relates to the tourism industry and do research on the information system requirements.

You are not limited to the list of ideas above, but you need to keep within the overall theme – the tourism industry. Note that you need to choose data and functionalities (services) in such a way as to develop a well-rounded application related to the topic.

NOTE: Your final program must comprise **one** single project with logically related parts.

2.4 What you need to be able to do the PAT

To be able to do the PAT, you need the following:

- The Delphi IDE (integrated development environment)
- An office suite with the following software:
  - Word processing software
  - Database software
- Storage media to save and backup your work electronically, e.g. a flash drive, rewritable CD/DVD

2.5 Malpractice

As the PAT is an individual project that is part of your final promotion mark, you may NOT:

- Get help from others without acknowledgement
- Allow others to do programming code for you
- Submit work which is not your own
- Share your work with other learners
- Include work directly copied from books, the internet or other sources without acknowledging it.

The above actions constitute malpractice, for which a penalty will be applied, depending on the seriousness of the offence.

NOTE: If you use work from other resources, it may not exceed 10% of the work that you submit.

2.6 Non-compliance

You will be given up to a part of Term 3 to submit outstanding work or to present yourself for the PAT. Should you fail to fulfil the requirements for the practical assessment task, you will be awarded a zero ('0') for the PAT component of IT.

2.7 PAT requirements

The project must include the following:

- A database connection and database manipulation that entail performing different CRUD (Create, Read, Update and Delete) operations
- A multiform GUI with good functionality and usability, based on sound HCI principles
- The use of a text file for input/output purposes, e.g. to populate data structures and to provide reports
- Other data structures that will be relevant to your program

Database

The database must:

- Have at least TWO linked tables (relational tables implementing referential integrity)
- Contain sufficient data volumes and uses a variety of field types (approximately 5 fields and at least 10 records per table)
- Be accessed and manipulated by the program using code constructs AND SQL statements

GUI

The graphical user interface (GUI) must:

- Have at least THREE forms/screens that allow for navigation between forms depending on the user choices
- Interact with the database and other data structures to provide the necessary input, processing and output
- Comply with general HCI principles

Text files

Your application must use a text file(s) for input and/or output.

Classes and objects

The appropriate use of one or more object(s) created within a class and used in other forms of the application.

NOTE: The object(s) must **NOT** be a duplication of one of the tables contained in the database and must add value to the program.

Other data structures/Advanced programming concepts

The suitable use of other data structures that are not already used, e.g. an array. Advanced programming concepts can also be applied, e.g. inheritance, polymorphism, overloaded methods.

NOTE: The mark obtained for your project will be greatly influenced by the quality of the programming code that manipulates the data successfully in order to adhere to the user requirements in the best possible way. Quantity cannot replace variety, effectiveness and quality.

2.8 Instructions for Phase 1

During this phase you have to show that you have done a proper and thorough user requirement analysis. This needs to be done in order to determine who the users are and what the users of the system would require it to do. The following can be used as a guideline:

Choose a topic from the provided TOPIC list or any related topic within the provided scenario.

SCENARIO AND SCOPE: DEFINE THE TASK

Write a brief description (approximately 200 words) in your own words to describe in general terms the problem/task, and how the project will solve the problem.

Your explanation must highlight that:

- You understand the needs of the task that you have chosen
- Your solution will solve the needs of the task

Also provide a simple/brief description of the scope of the project.

USER REQUIREMENTS

The ***user*** is the target audience and will thus determine the needs and requirements of the program. Determine the clients/users and their requirements.

The aim is to identify the user(s), user needs, acceptable limitations and processing requirements of the system. Use a table or a use case diagram to explain the role, activity and limitations of each user of the system.

NAVIGATION/DESCRIPTION OF FLOW DIAGRAM

Clearly indicate the logical program flow and navigation between screens. Use a flow diagram or any other form of illustration to present a global overview of the project/system.

DESIGN THE DATABASE

The aim is to design a relational database to serve as a data source, as well as to manipulate data contained in the database using programming code AND SQL statements.

Show the design of the database, including the tables, relationships, field names, field types and field sizes.

The database should provide data to the program to be processed and create reports.

The Delphi program must be able to manipulate the content of database tables, e.g. update/edit/ delete/add data, provide results of queries, provide reports.

DESIGN THE GRAPHICAL USER INTERFACE (GUI)

The aim is to produce a GUI design that considers good human-computer interaction (HCI) principles. Your design should include measures that prevent errors from occurring due to invalid input and that minimise the amount of information a user has to enter.

Use HCI design principles and design a GUI that considers the following:

- The user, type of user and context of the user
- User requirements, usability
- Dialogues – must be relevant, simple and clear
- Icon usage and presentation – well selected and relevant, well placed and purposely used
- Colour – appropriate use and combination of colours
- Feedback – neat, clear and well presented
- Helpful error messages
- Exits – clearly marked, placed correctly
- Shortcuts
- Flow of information on the screen – top to bottom and left to right
- Sensible use of space on the screen

Provide examples of planned data capture and data entry designs (screen dumps may be used from a prototype of the project but must be annotated) and of planned output design.

Show the GUI design following HCI principles of interface(s), excluding introductory screens.

DATA DICTIONARY

Text files

Your application must use a text file(s) for input and/or output. Explain where a text file(s) can be used in your application so that it adds value to the application.

Classes and objects

Your application must contain at least one object class. Explain where objects can be used in your application so that it adds value to the application.

Other data structures/Advanced programming constructs

Your program must use a one-/two-dimensional array/an array of objects OR apply programming concepts, such as inheritance, polymorphism, overloaded methods, method binding, etc.

SOFTWARE TOOL – INPUT, PROCESSING, OUTPUT (IPO)
(FORMAT, DATA TYPES/STRUCTURES, VALIDATION)

Use an IPO illustration/table to:

- Design the overall solution, considering all constituent parts and the interrelationships between the various parts of the program/system;
- Specify the format, data types, source of input, source of output, validation of input and error checking mechanisms;
- Specify processing that needs to be done and provide algorithm(s)/formulae to show how the processing will be done; and
- Provide a clear description to indicate the input, processing and output requirements of the system for at least TWO of the main interfaces.

HAND IN

Hand in a document that provides the following:

- A clear description of the chosen topic;
- User requirements – detailed information stating the role, activities and limitations of each user of the planned system;
- A clear description of data structures to be used:
  - A planned database design
  - The use of one or more text file(s)
  - The use of one or more class(es) and object(s)
  - The use of any other data structure/advanced programming concept
- A GUI design;
- A data dictionary; and
- The IPO design, including validation and error checking techniques.

2.9 Instructions for Phase 2 – Coding and testing

This is where you implement your design by using appropriate software tools (programming language, database software, IDE, etc.) and techniques to construct a solution to the problem.

DEVELOP THE DATABASE

Design and construct the database according to the planning document that was developed during Phase 1. Apply appropriate techniques and sound database development rules.

Pay attention to the following:

- Table names should start with the prefix 'tbl', e.g. *tblSuppliers*.
- The use of spaces in field names might affect reading data from fields into the Delphi application.
- The size of text fields must be restricted/limited as the columns in the DBGrid in the Delphi application will be affected by the field size.
- The data types of fields must be well thought out as this information will ultimately connect to components in the Delphi application, e.g. the difference between the Number and AutoNumber data types, the difference between saving a date as text or as a DateTime data type.
- Keep the purpose of the project in mind when setting up fields and tables.
- Ensure that the database connects correctly to the program and interacts with the program in a meaningful and effective way that supports the program once you have written the Delphi code.

DEVELOP THE GUI

Develop the GUI according to the planning document that was developed during Phase 1. Use appropriate components to ensure easy use and effective navigation. Follow HCI principles to ensure that the application is user-friendly and provides all necessary requirements for the user(s) to use the program effectively and navigate through the options/functionalities easily.

WRITE THE CODE

Write code to develop the program/system according to the planning document that was developed during Phase 1.

Note the following:

- Use good programming techniques and structures.
- Implement effective algorithms and sound defensive programming techniques to produce a robust program.
- Use appropriate structures to satisfy the requirements of the algorithms.
- Use multinested loops and conditional structures.
- The following data structures are compulsory in addition to the database:
  - Text file – reading OR writing OR appending
  - Class(es) and object(s)
  - The use of any other data structure not already used/advanced programming construct

- Use OOP principles, re-use code, use functions, procedures, methods and objects.
- Use relevant validation procedures and components.
- Develop a well-designed and user-friendly GUI.
- Rename relevant components to add to readability and documentation of your code.
- Input data using the most effective method, e.g. a text file, database, keyboard, most suitable GUI components.
- Process the data using the most appropriate methods.
- Generate output of data using the correct components and structures, with formatting where needed.
- Ensure smooth interaction between classes/forms/tabs.
- Correctly manipulate and query the database.

DOCUMENT THE PROGRAM

Project notes for the user:

These project notes must describe how the user should interact with the program. It can include notes on how to navigate through the program, specific requirements, such as passwords and installation procedures if applicable. The notes must also describe any known bugs or problems. Project notes can be written as part of the help function of the program. Tool tip texts can also be provided.

Project notes for developers:

These project notes could include specifications/limitations applicable to the project to ensure that the program is installed and set up correctly, e.g. the connection to the database.

Project notes related to the programming code should be embedded as comments in the code. Document the code so that other programmers will be able to interpret the code and understand the purpose of individual pieces of code. It should also include comments to explain sections of complex code.

TEST THE PROGRAM/SYSTEM

Test the program/system using clearly defined typical data, erroneous data and boundary (extreme) data.

HAND IN

Hand in:

- The completed Delphi project (Delphi code, text files, database and any other resources required to execute the program successfully) and project notes;
- The declaration of help received (**ANNEXURE B**); and
- The declaration of authenticity (**ANNEXURE C**)

INTERVIEW

Demonstrate your program and answer questions about the program and the code during an interview session.

Guidelines for the demonstration of the project:

- The teacher will schedule dates and times for demonstrations. About 15 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 a computer.
- You must execute your computer program and show all the features of the program to the teacher for evaluation.
- The teacher can require you to execute test procedures to make sure that the entire program is working correctly.
- The teacher can use the mark sheet for Phase 2 as a guideline and allocate marks accordingly during the demonstration.
- As part of the demonstration, the teacher will identify random pieces of the programming code in the project and ask you to explain the purpose and working thereof. This is done to ensure that you did the coding yourself. A similar type of procedure will be followed during 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 to finalise the mark.

2.10 ANNEXURE A: ASSESSMENT TOOLS

Phase 1:	**Name of learner:**						
SCENARIO/SCOPE (± 200 words)	**4**	**3**	**2**	**1**	**0**		
Scenario and Scope • Topic is clearly stated • Thorough description of what the problem/task involves (purpose) • Describe a possible solution for the problem/task • Brief description of the scope	An excellent presentation of all FOUR points listed	All FOUR points were presented with shortcomings OR A good presentation of THREE points	THREE points were presented with shortcomings OR A good presentation of TWO points	TWO points were presented with shortcomings OR A good attempt to present ONE of the points	Totally inadequate or not applicable Poor or no coverage of the aspects No scope or extremely vague and unclear	**4**	
USER REQUIREMENTS	**4**	**3**	**2**	**1**	**0**		
State WHO the users are. Role, activity and limitations of the users (In table format OR a use case diagram)	• Role, activity and limitations of at least TWO different types of users of the system discussed • Well documented, neat and to the point	• Minor shortcomings in the discussion of role, activity and limitations of at least TWO different types of users of the system • Well documented, but can improve slightly	• Shortcomings in the discussion of role, activity and limitations of users, e.g. sections left out • Only ONE user of the system discussed • Not well documented, but still acceptable	• Major shortcomings in the discussion of role, activity and limitations of users • Only ONE user of the system discussed • Poorly documented – not acceptable	• Not done or incorrect or irrelevant	**4**	
Navigation/Description of flow diagram	**4**	**3**	**2**	**1**	**0**		
A diagrammatical representation of the design and flow of events when the program is used	An excellent attempt to show the sequence of all steps and flow of events when the program is executed with no shortcomings	A good attempt to show the sequence of all steps and flow of events when the program is executed with minor shortcomings	A satisfactory attempt to show the sequence of steps and flow of events when the program is executed with significant shortcomings	A poor attempt to show the sequence of steps and flow of events when the program is executed with major shortcomings	No diagram OR Incorrect, irrelevant or unsuitable for the application	**4**	

DATA STRUCTURES	4	3	2	1	0		
Database design • All fields relevant • Type and size of fields well chosen • Relational • Normalised	All database design requirements met A well-designed relational database normalised correctly	Good database design with minor shortcomings A relational database normalised with minor shortcomings	Acceptable database design with several shortcomings A relational database normalised with major shortcomings	Database design done, but with limited value A poor attempt to normalise a relational database	No database or incorrect or irrelevant No relational database Database not normalised	**4**	
Class description and class diagram	Class well defined with attributes and methods that serve a definite purpose in the context of the application Class diagram included; illustrates an appropriate design in terms of the attributes and the proposed methods with no errors	One incorrect/irrelevant aspect identified in the class diagram/description, e.g. Scope Return type Method type Parameters	Two incorrect/irrelevant aspects identified in the class diagram/description, e.g. Scope Return type Method type Parameters	Three or more incorrect/irrelevant aspects identified in the class diagram/description, e.g. Scope Return type Method type Parameters	No class diagram or totally incorrect	**4**	
Text file(s) and array/advanced programming concepts	Excellent and relevant description of use of text file(s) AND a good application of an array/ advanced programming concepts described	Acceptable and relevant description of use of text file(s) AND an acceptable application of an array/ advanced programming concepts described	Description of use of text file(s) with some shortcomings AND the application of an array/ advanced programming concepts is described with shortcomings	An attempt to describe the use of a text file with major shortcomings OR an array/ advanced programming concepts with major shortcomings is described	Not done or incorrect or irrelevant	**4**	
GUI DESIGN	**4**	**3**	**2**	**1**	**0**		
• Design fits to program's intended use • Appropriate components • Ease of use, logical flow • Clearly marked navigation • Friendly dialogue/Help	Good GUI design All of the listed principles applied throughout the system, e.g. with data capturing, output, navigation	Satisfactory GUI design Most (at least 4) of the principles applied throughout the system, e.g. with data capturing, output, navigation	Limited GUI design Most (at least 3) of the principles applied throughout the system, e.g. with data capturing, output, navigation	Poor GUI design Applied less than 50% (less than 2) of the principles	GUI design not functional or does not support the intended use at all	**4**	

IPO – Software design tool							
DATA INPUT	**4**	**3**	**2**	**1**	**0**		
Input interfaces (at least TWO) • Source of input, such as keyboard, text file, array or database • Data type • Format of input, e.g. date, gender (M/F) • GUI component used	Clearly describes all inputs according to all FOUR points listed	Minor shortcomings in describing all inputs according to all FOUR points listed	Clear description according to THREE points listed OR Major shortcomings in describing all inputs according to all FOUR points listed	Poor attempt to describe input values	No inputs described or incorrect	**4**	
Input validation • At least FOUR different data types validated • At least FOUR inputs validated including: ○ Validate for NULL/empty field AND ○ Test if value was selected in a selection component • Associated error messages	Clearly describes all points listed	Clearly describes TWO points listed OR Minor shortcomings in describing all points listed	Clearly describes ONE points listed OR Major shortcomings in describing all points listed	Poor attempt to describe validation	No validation described or incorrect	**4**	
DATA PROCESSING	**4**	**3**	**2**	**1**	**0**		
WHAT processing will need to be done	Lists at least EIGHT processes to be done	One or two processes not listed	About 50% of the processes listed	Only one or two processes listed	No processes listed	**4**	
HOW processing will be done – supply algorithms, formulas, etc.	Clearly describes how at least FOUR processes will be done	Clearly describes how THREE processes will be done	Clearly describes how TWO processes will be done OR An attempt to describe how FOUR processes will be done	Clearly describes how ONE process will be done OR A poor attempt to describe TWO or THREE processes	Processes not described or incorrect or irrelevant	**4**	

DATA OUTPUT	4	3	2	1	0		
Output interfaces (at least TWO) • Data to output • Format of the output, e.g. currency, date • Output component, such as dbGrid, rich edit, label, etc.	Clearly describes all outputs by addressing all THREE points listed	Minor shortcomings in describing all outputs by addressing all THREE points listed	Clear description of all outputs by addressing TWO points listed OR Limited outputs described	Poor attempt to describe outputs	No output described or incorrect	**4**	
					TOTAL	**48**	

Comments/Feedback:

____

____

Teacher name:________________________ **Teacher signature:** ________________________ **Date:**____________________

Phase 2:	**Name of learner:**						
PROJECT ASSESSMENT							
DATABASE DESIGN	**4**	**3**	**2**	**1**	**0**		
Implementation of database design	Database design correctly implemented, with at least 2 relational tables, suitable fields, data types and sizes Large/Adequate data volume	Database design correctly implemented, with at least 2 relational tables, suitable fields, data types and sizes Limited volume of data used	Database design using at least 2 relational tables, but not properly implemented Errors in fields, data types and sizes	Database design not relational One table with suitable fields, data types and sizes	Totally inappropriate or incorrect or not used	**4**	
GUI DESIGN	**4**	**3**	**2**	**1**	**0**		
Ease of use/HCI principles • Excellent layout and communication (screen tips, feedback, help, etc.) • Most appropriate components • Readable/Relevant input/output • Excellent use of effects/colour/ icons/shortcuts/tool tip text, etc.	Excellent – all four aspects applied correctly throughout the program	Good – one aspect omitted or not applied well	Satisfactory – two aspects omitted or not applied well	Limited – more than two aspects omitted or not applied well	Poor GUI design Little/No thought given to HCI principles	**4**	
DATA DICTIONARY	**4**	**3**	**2**	**1**	**0**		
Variables and components • Variety of appropriate variable types • Correct use of local and global variables • Proper naming convention of variables, e.g. iNumber, sName • Correct prefix for components, e.g. edt, red, cmb	Excellent – all four aspects applied correctly in all instances	Good – one aspect omitted or not used well	Satisfactory – two aspects omitted or not used well	Limited – more than two aspects omitted or not used well	Totally inappropriate or incorrectly applied	**4**	

DATA STRUCTURES (excl. Database)	**4**	**3**	**2**	**1**	**0**		
Text file(s)	Excellent and relevant use of one or more text file(s)	Good use of a text file	Limited use of a text file	An attempt to use a text file with shortcomings	Not done or incorrect or irrelevant	**4**	
Class(es) and object(s)	Applicable class, correctly compiled with applicable attributes and methods Object(s) correctly instantiated Methods correctly defined and called Object(s) integrated well with the application	One minor shortcoming in the compilation of the class/definition of a method/call of a method Object(s) integrated satisfactory with the application	Two minor shortcomings in the compilation of the class/definition of a method/call of a method Object(s) integrated with the application with shortcomings, e.g. limited method calls	More than two shortcomings in the compilation of the class/ definition of a method/call of a method Object(s) not integrated well with the application	Class not implemented/poorly defined OR not relevant to the application OR duplication of a table in the database	**4**	
Array OR advanced programming concepts	Excellent and relevant use of array(s) Could include: Sensible use of array of objects or parallel arrays or two-dimensional array OR Excellent application of inheritance or polymorphism or method binding or effective use of overloaded methods	Limited use of array(s) Could include: Array of objects or parallel arrays or two-dimensional array OR A basic application of either inheritance, polymorphism, method binding or effective use of overloaded methods	Limited use of array(s) with minor shortcomings OR A basic application of inheritance or polymorphism or method binding or the use of overloaded methods with minor shortcomings	An attempt to use an array Shows potential but not used for a suitable purpose or does not work correctly OR An attempt to apply inheritance or polymorphism or method binding or the use of overloaded methods with major shortcomings	Not done or incorrect or irrelevant	**4**	
INPUT	**4**	**3**	**2**	**1**	**0**		
Input data • Variety of sources of input, such as from the keyboard, text file, array or database • Correct data types • Appropriate format used, e.g. date, gender (M/F) • GUI component used	Excellent application of all FOUR aspects listed	Minor shortcomings in the application of all FOUR aspects listed	Approximately 50% of the aspects listed correctly applied	Limited application of the aspects listed	No application of the aspects listed	**4**	

Validation/Error catching	A variety of validation/error catching for relevant input Clear and appropriate error messages and exception handling mechanisms	A variety of validation/error catching for relevant input Mostly clear and appropriate error messages and exception handling mechanisms	Limited validation/error catching Error messages and exception handling sometimes inappropriate/ not meaningful	Validation/error catching poorly done or inappropriate/not meaningful	No effort at validation/ error catching	**4**	
PROCESSING	**4**	**3**	**2**	**1**	**0**		
Algorithm correctness/ Processing	All algorithms used are appropriate, work correctly and meet all processing requirements	Appropriate algorithms that work correctly but ONE processing requirement not met	50% of the algorithms used are appropriate, work correctly and meets most processing requirements	Algorithms are mostly inadequate/not working correctly, processing requirements not all met	Totally inadequate or not working correctly	**4**	
Algorithm efficiency	All algorithms provide the most efficient solutions Good programming techniques used Effective modular design with correct use of own functions and procedures	Most algorithms provide the most efficient solutions Acceptable programming techniques used Limited modular design with correct use of own functions and procedures	Limited efficiency of algorithms used Few algorithms use good programming techniques Poor modularity with limited use of own functions and procedures	Poor efficiency of algorithms used Algorithms do not use good programming techniques Attempted use of own functions and procedures	Totally inadequate or not working correctly	**4**	
Relevant and appropriate use of complex code, e.g. Dynamic component	Excellent use of complex code that works correctly Adds value to the system	Works correctly Adds value to the system	Works correctly with minor shortcomings	An attempt has been made with major shortcomings	No attempt has been made	**4**	
OUTPUT	**4**	**3**	**2**	**1**	**0**		
• Layout • Readability/Clarity, e.g. columns, headings • Formatted, e.g. currency • Most appropriate component/data structure used for output	Excellent application of all FOUR aspects listed	Minor shortcomings in the application of all FOUR aspects listed	Approximately 50% of the aspects listed applied correctly	Limited amount of aspects listed applied correctly	None of the aspects listed applied correctly	**4**	

Database manipulation (Delphi code AND SQL)	**2**	**1**	**0**		
Sort records in a table	Works correctly and is applicable to task	Works, but poorly constructed/ not applicable to task	Not done or does not work or incorrect	**2**	
Search for data in a table				**2**	
Insert a new record to a table				**2**	
Delete a record from a table				**2**	
Edit selected fields in a record				**2**	
SQL using string manipulation, e.g. using Left, Mid and Right				**2**	
Complex selection SQL query, e.g. using HAVING/combination of AND and OR in a single query				**2**	
At least two SQL queries using different calculations, such as minimum, maximum, sum and average				**2**	
At least one query involving two tables using an SQL statement				**2**	
At least one query involving two tables using Delphi code				**2**	
At least one dynamic SQL using a variable				**2**	

Documentation	**4**	**3**	**2**	**1**	**0**		
Comments/Notes (Explanation of program and code)	Code clearly annotated to fully explain all necessary parts Explanation shows excellent insight Extensive project notes present and of an excellent standard Clearly explains working of program	Code clearly annotated to explain all necessary parts Explanation shows good insight Project notes present and of a good standard	Code annotated to explain most necessary parts Explanation shows some insight Project notes present and of a moderate standard	Code annotated to explain certain parts Explanation shows little insight Inadequate project notes present	No comments or no project notes	**4**	
Exceptional features	Contains feature(s) that are NOT part of the curriculum, e.g. connecting or running on a mobile device Feature(s) must show a high level of complexity to implement Learner must show knowledge and skills on how the feature(s) were coded	Contains eye-catching feature(s), e.g. animation, using fairly complex code in an original and sensible way to enhance the look and feel/functionality of the product Learner must show knowledge and skills on how the feature(s) were coded	Uses standard Delphi GUI features, e.g. eye-catching buttons and other GUI components in an original and sensible way to enhance the look and feel/functionality of the product Learner must show knowledge and skills on how the feature(s) were coded	At least one attempt to apply standard Delphi GUI features to enhance the look and feel/ functionality of the product Learner must show knowledge and skills on how the feature(s) were coded	No exceptional features	**4**	
					TOTAL (implementation):	**78**	

General: Final product and impression Name of learner:

Aspect	4	3	2	1	0		Mark
Completeness	Reached initial goal and met all stated requirements in Phase 1	Met at least 80% of the initial requirements	Met more than 50% of requirements	More than 50% of initial requirements not met	Almost none of the initial requirements met	**4**	
Professional product	Useful and can be implemented as a real-life application Well-designed and user-friendly Contains no errors	Useful as a real-life application with minor adjustments Good design and user-friendly Contains minimal errors	Useful as a real-life application with major adjustments Good design and user-friendly Contains several errors	Not ready to be implemented as a real-life application, but has some potential	Not ready to be implemented as a real-life application Poor design	**4**	
Ability to explain code	Explained all selected code clearly and with confidence Shows excellent insight	Explained the selected code with minor shortcomings Shows insight	Unable to explain some of the selected code adequately Shows some insight	Unable to explain most of the selected code Limited insight	Unable to explain any selected code No insight	**4**	
Attitude and commitment			Kept to due dates Well-designed phases Showed exceptional commitment and pride in work done	One of the phases was not on time Showed some commitment and pride in work done	None of the phases on time Showed no commitment	**2**	
					TOTAL:	**14**	

Assessment Summary

PHASE	FOCUS	MAXIMUM MARK	MARK OBTAINED
Phase 1	Analysis and Design	48	
Phase 2	Coding and Implementation	78	
General	Final Product and Impression	14	
TOTAL		**140**	
Adjustment %			
Final mark (Total x Adjustment %)			

DECLARATION OF AUTHENTICITY

I hereby declare that the work assessed is solely that of the learner concerned (except where there is clear acknowledgement and a record of any substantive advice/assistance given to the learner) and was conducted under supervised/controlled conditions to ensure that the work has not been plagiarised, copied from someone else or previously submitted for assessment by anyone.

Comment/Feedback:

____

____

Teacher name: ____ **Teacher signature:**____________________________________ **Date:**_________________________

2.11 ANNEXURE B: LEARNER DECLARATION

Learner declaration – Phase ____

I understand that work submitted for assessment must be my own.

Have you received help/information from any person to produce this work?

□No □Yes (Provide details below.)

Help/Information received from (person):	Nature of the help/information (Provide evidence):
___________________________ SIGNATURE OF LEARNER	___ / ___ / 2019 DATE

2.12 ANNEXURE C: DECLARATION OF AUTHENTICITY

Declaration of authenticity

Learner name		**ID number**	
Grade	12	**Year**	2020
Subject	INFORMATION TECHNOLOGY		
Practical Assessment Task (PAT)		**Teacher**	

I hereby declare that the contents of this assessment task are my own original work (except where there is clear acknowledgement and appropriate reference to the work of others) and have not been plagiarised, copied from someone else or previously submitted for assessment by any person.

____________________________ ___ / ___ / 2019

SIGNATURE OF LEARNER DATE

2.13 Guidelines for teachers to provide guidance

2.13.1 What are the learners required to do and provide?

Learners are required, with appropriate supervision, to:

- Choose an area of interest within the topic/scenario provided
- Formulate a focus question that can be investigated/researched
- Plan, research and carry out the project
- Deliver a report to a specified audience
- Provide evidence of all stages of the project for assessment

2.13.2 How will learners go about it?

Learners will:

- Plan and complete an individual project, applying a range of programming and software engineering skills and strategies to meet the objectives as set out by the PAT requirements
- Identify questions to ask
- Obtain, critically select and use selected information from a range of sources, process and analyse data, apply it relevantly and demonstrate understanding of appropriate linkages, connections and complexities of the topic and focus question
- Select and use a range of skills, including design tools and algorithms, solve problems, take decisions critically, creatively and flexibly, to produce a software solution
- Evaluate outcomes, both in relation to PAT requirements and own learning and performance
- Use appropriate communication skills and media to present evidence in an appropriate format

2.13.3 Skills required

Learners must be able to:

- Do a complete user requirement analysis which includes a complete description of the role, activities, requirements and limitations of at least TWO different users of the planned system
- Bring together information to suit the content and purpose
- Apply decision-making and problem-solving skills
- Extend planning, research, critical thinking, analysis, synthesis, evaluation and presentation skills
- Develop confidence in applying the content, programming and software engineering principles and techniques they have studied
- Develop and apply skills creatively, demonstrating initiative and enterprise
- Seek advice and support when needed

2.13.4 What must the learners be taught beforehand?

The taught elements include:

- Application software and ICT skills that will enhance the production of the report and the development of the project covering analysis and execution
- Solution development content and skills, including the ability to define a task
- Project management skills, including time, resource and task management

2.14 Malpractice

Learners may NOT:

- Get help/guidance from other persons without acknowledgment (complete **ANNEXURE B** for EACH phase)
- Allow others to do the programming code for their project
- Submit work which is not their own
- Lend work to other learners
- Allow other learners access to, or the use of, their own independently-sourced source material (this does not mean that learners may not lend their books to another learner, but learners should be prevented from plagiarising other learners' research)
- Include work copied directly from books, the internet or other sources, without acknowledgement and attribution
- Submit work typed or word-processed by another person

These actions constitute malpractice, for which learners will be penalised.

If malpractice is identified, the assessment authorities must be notified and details of any work which is not the learner's own must be recorded.

2.15 Learner declaration of authenticity of the PAT

For each phase, learners complete a declaration (**ANNEXURE B**) for the work done during that specific phase. All substantive advice/help given to the learners should be recorded as part of the phase documents.

After completing the PAT, learners should sign the declaration of authenticity (**ANNEXURE C**) to confirm that the work submitted is their own.

2.16 Role of the teacher

The teacher will teach the information management content, skills and strategies prior to the project.

While managing the project and supervising the learners, the teacher will:

- Conduct an initial planning review to discuss the topic/scenario, requirements, objectives and development of the project
- Agree on the focus question (learners should record the guidance given as part of the Phase 1 documents, e.g. where appropriate, record their own initial question with clear evidence of the guidance and the final question)
- Give regular feedback to learners, e.g. to formulate a focus question that is suitable and manageable
- Assess the work of the learners at the end of each phase using the standardised assessment tool and record feedback given
- Endorse each learner's assessment by signing the assessment tools for each phase, including a final declaration that the evidence submitted for assessment is the unaided work of the learner
- Confirm their evaluation based on continuous observation and feedback, as well as an interview session to provide a final judgement regarding independent work, insight and problem-solving
- Make the assessment of the work of the learners following any standardising and internal moderation procedures required

The teacher will assess the potential project (task definition and scope) against the following checklist:

- Is the focus area suitable for the project?
- Does the focus question allow the learner to investigate and access the higher-level concepts and skills in the assessment objectives, e.g. to plan, research, analyse, evaluate and explain, rather than simply describe and narrate?
- Are the focus question and proposed action clear and focused on an issue that can be managed within the time frame and available resources?
- Do the focus question and proposed action indicate that the learner will be capable of investigating and researching the topic and carrying out the activity or task independently and within appropriate ethical or methodological guidelines?
- Is the learner likely to face difficulties understanding the task and issues associated with the focus question?

The teacher will authenticate the PAT:

- The teacher will confirm on the assessment tool that the work assessed is solely that of the learner concerned and was conducted under supervised/controlled conditions.
- The teacher will sign the assessment tool of each phase.

2.17 Supervised/Controlled conditions

The PAT must be managed in such a manner to be able to confirm that the work assessed is solely that of the learner concerned.

2.18 Managing the PAT

The teacher must plan his/her work schedule according to the time allocated for the PAT in the CAPS document for Information Technology (teaching plan for Grade 12).

There are different possible approaches to managing the PAT:

Option 1:

- The teacher could dedicate a portion of the time on a weekly basis to the PAT while simultaneously continuing with normal teaching to complete the Grade 12 curriculum in the rest of the week.
- If he/she choose this option, he/she should start with the PAT process towards the end of the first term, completing one phase per term.

Option 2:

- Teachers could dedicate a continuous period of time to the PAT, e.g. the last week(s) of each term, also completing one phase per term.

2.19 Evidence of assessment

Evidence presented for assessment must show how the individual learner has met the assessment objectives and criteria and must include the planning, feedback and progress of the project.

The evidence for assessment will include the following:

- The project product, including the documentation/report (content only, without the cover page, table of contents, references, graphics), design documents, final program (fully documented) and other evidence (for each phase)
- The completed learner assessment tool (for each phase)

2.20 Interview

Guidelines for the evaluation of the project:

- Schedule dates and times for demonstrations – allow about 15 minutes per project.
- Take in all the documentation before the demonstration takes place – at least one week in advance – and evaluate the documentation before the demonstration session.
- Learners should demonstrate their projects electronically on the computer.

- During the demonstration session learners should execute test procedures to show that the entire program is working correctly.
- Use the mark sheet for Phase 2 as a guideline and allocate marks accordingly during the demonstration.
- As part of the evaluation, identify random pieces of programming code in the project and ask the learner to explain the purpose and working of the randomly selected code. This is done to ensure that the learner did the coding him-/herself. A similar type of procedure will be followed during moderation. If a learner cannot explain the code used in the project, a mark of zero should be awarded for the project.
- Make sure that the learner hands in the electronic copy of the project that was demonstrated. Use this copy to allocate any outstanding marks in order to finalise the mark.

2.21 Requirements

(National Protocol for Assessment Grades R–12, Chapter 3)

Practical assessment task components must:

- Comprise assessment tasks that constitute the learner's PAT mark as contemplated in Chapter 4 of the IT CAPS
- Include a mark awarded for each assessment task (phase), as well as a consolidated mark
- Be guided by assessment components as specified in Chapter 4 of the IT CAPS
- Be available for monitoring and moderation
- Be evaluated, checked and authenticated by the teacher before being presented as the learner's evidence of performance

2.22 Non-compliance

(National Protocol for Assessment Grades R–12, Chapter 3)

The absence of a PAT mark in IT, without a valid reason, will result in the learner not being resulted for the subject.

The learner will be given up to the first week of Term 3 to submit outstanding work or present him-/herself for the PAT. Should the learner fail to fulfil the outstanding PAT requirements, such a learner will be awarded a zero ('0') for the PAT component for IT.

In the event of a learner not complying with the requirements of the PAT, but where a valid reason is provided:

- He/She may be granted another opportunity to be assessed in the assigned tasks, based on a decision by the head of the assessment body
- The learner must, within three weeks before the commencement of the final end-of-year examination, submit outstanding work or present him-/herself for the PAT
- Should the learner fail to fulfil the outstanding PAT requirements, the mark for the PAT component will be omitted and the final mark will be adjusted for promotion purposes in terms of the completed tasks

Valid reasons in this context include the following:

- Illness, supported by a valid medical certificate, issued by a registered medical practitioner
- Humanitarian reasons, which include the death of an immediate family member, supported by a death certificate
- The learner appearing in a court hearing, which must be supported by written evidence
- Any other reason that may be accepted as valid by the head of the assessment body or his/her representative

3. CONCLUSION

Upon completion of the practical assessment task, learners should be able to demonstrate their understanding of the industry, enhance their knowledge, skills, values and reasoning abilities, as well as establish connections to life outside the classroom and address real-world challenges. Furthermore, the PAT develops learners' life skills and provides opportunities for learners to engage in their own learning.