basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

INFORMATION TECHNOLOGY

GUIDELINES FOR PRACTICAL ASSESSMENT TASK

Grade 12

2016

These guidelines consist of 38 pages.

INTRODUCTION

The 16 Curriculum and Assessment Policy Statements subjects which contain a practical component all include a Practical Assessment Task (PAT), for example a Practical or Performance Assessment Task. These subjects are:

- AGRICULTURE: Agricultural Management Practices, Agricultural Technology
- ARTS: Dance Studies, Design, Dramatic Arts, Music, Visual Arts
- SCIENCES: Computer Applications Technology, Information Technology
- 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-year examination mark. The PAT is implemented across the first three terms of the school year, which 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 also allow for the assessment of skills that cannot be assessed in a written format, for example test or examination. It is therefore important for schools to 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.

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
- To implement higher order and critical thinking skills, formulate strategies and to solve problems on different levels
- To develop good working practices to prepare you for the real world, like
  - Time management
  - Thorough research and planning
  - Perseverence 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.

In order to achieve good results for your final product, you would have to

- Provide a comprehensive and methodical report that clearly outlines the research and investigation done on the purpose and scope of the project
- Provide a possible solution for the task that was identified
- Provide a suggested design of what the product will look like

The PAT is divided into **three phases**, as explained below:

- Phase 1: Outlines the project task and solution and provides the relevant research methods
- Phase 2: A possible design of the project
- Phase 3: A working, fully documented Delphi/Java program that implements the planned solution

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

Phase 1: Not later than one week before the end of Term 1
Phase 2: Not later than one week before the end of Term 2
Phase 3: Not later than three weeks before the start of the Grade 12 final examination

LEARNERS NEED TO STRICTLY ADHERE TO THE DUE DATES FOR EACH PHASE.

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

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	Research and Analysis	30
Phase 2	Design	40
Phase 3	Level 1: Coding and Testing (80) Level 2: Higher Order Solutions (20)	100
General	Final product and impression	10
	Total:	**180**

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.

The Topic

Health

Good health is important to everyone. Computer systems are required by all organisations and institutions that take care of people's health.

You are required to plan and develop **a computer program** for any organisation, institution or company related to people's health.

Examples and ideas related to the topic on health:

- Gymnasiums, for example registration of members, exercise programmes suited to members, fitness tests and results, et cetera
- Dietary institutions, for example registration of members, analysing their current diet, dietary programmes, record of progression, et cetera
- Pharmacy/Stock-taking systems of medicine, for example a database of medicines, doctors, patients and medical aids, purchasing and selling of stock, prescription medicine, et cetera
- Hospital systems, for example casualties – registration of patients, diagnosis, doctor who treated the patient, ward and room the patient was admitted to, et cetera
- Physiotherapists, for example registration of patients, evaluation, exercise program, progression reports, et cetera

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

You are not limited to the list of ideas above, but you need to choose data and functionalities (services) in such a way so as to develop a usable and well-rounded application related to the topic of health.

Specific requirements:

- The project must include the major development tools, for example a database and programming language constructs, appropriately integrated.
- Include all the aspects mentioned in the PAT requirements (see page 8) and Phase 3 marking guidelines **(Annexure A)**.

General programming aspects that will be assessed in your project:

- Programming style
- Graphical user interface (GUI)
- Use of sound human-computer interaction (HCI) and software engineering principles
- Input/Output management
- 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

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

Overview

PHASE 1 – ANALYSIS

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

- Use the scenario to identify the task, and to outline the purpose, content scope of the task.
- Determine the clients/users and their requirements.
- Perform the necessary research, for example interviews, surveys, photographic material et cetera.
- Provide a methodical report based on the research/investigation carried out.
- Provide a list of references and the evidence of research.
- Compile detailed requirements and specify what your solution should provide for in order to meet these requirements.

NOTE: The ***user*** is the target audience and will thus determine the needs and requirements of the program.

PHASE 2 – DESIGN

The purpose of phase 2 is to determine ***how*** the program/system will meet the requirements. It needs to provide a well-planned solution.

The following must be done:

- Clarify the user requirements, indicating specifically how each requirement will be met in your program/solution.
- Provide a clear IPO chart to indicate the input, processing and output requirements of the system for at least THREE of the main interfaces.
- Use at least ONE more relevant software planning tool to clarify the requirement.

NOTE: The following tools can be used in the design process:
  - TOE charts, ERD diagrams.
  - Flow charts, UML diagrams (class diagrams and use case diagrams).

- Show the design of the database, including the tables, relationships, field names, field types and field sizes
- Clearly indicate the logical program flow and navigation between screens.
- Show the GUI design following HCI principles of at least THREE different screens, excluding the introductory screens.
- Give at least TWO examples of how you would ensure the entry of valid data for the system using the GUI interfaces that you have provided.

PHASE 3 – CODING AND TESTING

The purpose of Phase 3 is to implement the design by writing the code and testing the program. The following must be done:

Level 1 requirements: Coding and Testing

- Write the programming code to implement the design and complete the program.
- Use appropriate structures to satisfy the requirements of the algorithms.
- Use multinested loops and conditional structures.
- The use of at least ONE text file is compulsory. Use other data structures where applicable.
- Use local and global variables.
- Use OOP principles, re-use code, use functions, procedures, methods and objects.
- Use relevant validation procedures and components.
- Clarify sections of the code by adding comments.
- Write project notes to help navigate through the program, provide tool tip texts. These project notes must describe how to interact with the program. The notes must also describe any known bugs or problems. Project notes can be written as part of the help function of the program.
- Develop a good GUI and rename relevant components.
- Input data using the most effective method, for example a text file, database, keyboard, components
- Process using the most appropriate methods.
- Generate output of data using the correct components and structures, with formatting where needed.
- Compile a complex code/string using string-handling techniques.
- Ensure smooth interaction between frames/forms/tabs.
- Use at least TWO different dynamic components.
- Correctly manipulate and query the database.
- Demonstrate your program and answer questions about the program and the code during an interview session.

Level 2 requirements: Higher-order solutions

- Display a drawing/animation/graph/map or timer.
- Play a video/thread/networking mobile app or a time-based simulation.
- Ensure there are at least THREE calculations, ONE of which must be complex.
- Include a variety of at least THREE SQL statements and programming code for file querying and maintenance, such as using aggregate functions, calculated fields, inserting of data, deleting of data and updating of data.
- Include complex programming code such as dynamically parameterised queries, that is assigning parameters at runtime.

PAT requirements

The project must include the following:

- Database manipulation through programming language constructs
- A multi-form/multi-screen GUI with good functionality and usability, based on sound HCI principles
- The use of a text file for input/output purposes, for example to populate data structures and to provide reports.

Database

The database must:

- Have at least TWO linked tables (relational tables)
- Involve sufficient data volumes and use a variety of field types (approximately 10 records)
- Must be accessed and manipulated by the program using code constructs
- Must be accessed and manipulated by the program using SQL statements

Text files

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

The data from the text file could be used to:

- Do calculations and manipulations in combination with data from the database
- Update existing data
- Insert/Update records in the database

AND/OR

- Create report(s) by writing to a text file.
- Could be used as a help file

GUI

The graphical user interface (GUI) must have at least:

- THREE forms/screens and
- TWO components (different types) that are created dynamically

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.*

What you need to be able to do the PAT

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

- Java/Delphi programming software, including a GUI IDE (Integrated Development Environment)
- An office suite with the following software:
  - Word processing software
  - Database software
- Internet access to find data and information
- Access to other sources, such as printed media (for example magazines, newspapers, brochures, textbooks) or other electronic material (for example e-books, e-articles).
- Access to facilities to convert hard copies to electronic documents, for example a scanner, digital camera
- Storage media to save and backup your work electronically, for example a flash drive, rewritable CD/DVD

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 30% of the work that you submit.

Non-compliance

You will be given up to three weeks before the commencement of the final end-of-year examination to submit outstanding work or present yourself for the PAT.

Should you fail to fulfil the Practical Assessment Task requirements, you will be awarded a zero ('0') for the PAT component of IT.

Instructions for Phase 1

You need to clearly understand the problem/task at hand. Research/Investigate the topic well so that you have a good idea of what is expected of the software you need to develop.

Select a specific application to apply the specifications described in this document. Use relevant research strategies; for example interviews, surveys, photographic material and the Internet. Carefully analyse the needs or application requirements to determine ***what*** the programming solution should do and provide.

The deliverable of this phase is a clear explanation of what the problem/task is and what the solution should be capable of doing in terms of the needs of the user (company)/task and their stated objectives.

DEFINE THE TASK

Write a brief description (approximately 150 words) in your own words to describe in general terms the problem/task and how the project will solve the problem. In other words, the description should be written to explain your choice of scenario within the topic based on health.

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

The description must give the overall picture of what the purpose and scope of the project is, but *not the details*.

DO RESEARCH

The research stage is where you gather facts about the nature of the program you are developing.

There must be a minimum of TWO different research methods.

Your research should help you clarify the type of program, provide some useful examples that can guide you and should provide a clear understanding of why the particular type of program is suitable for this project.

The outcome/summary of the research is a report (approximately 600 words).

The report should be written in clear, unambiguous language and should include the following aspects:

- Key areas pertaining to the topic
- A summary of results derived from the research
- the scope and limitations

Provide evidence of research:

All/most material used in the research process must be submitted

References:

A list of references must be provided to authenticate the research carried out.

User requirements:

The aim is to derive and establish the user, data and processing requirements of the system, including a consideration of the human aspects (for example a suitable user interface) and the physical environment (for example hardware requirements). Identify the prospective user(s) and identify the user needs and acceptable limitations.

- Use a table or a 'use case diagram' to explain the role, activity, requirements and limitations of each user of the program.
- Provide evidence of a user acceptance test for users of the program.

HAND IN

Hand in a document that contains the following:

- The project description (approximately 150 words)
- A report (approximately 600 words)which contains the following
  - A summary outlining the research/investigation/analysis
  - A conclusion
- References
- Detailed information on user requirements stating the role, activities and limitations of each user
- Evidence of research

Instructions for Phase 2

This is where you plan the detail and provide information on ***HOW*** you will go about solving the problem.

The aim is to specify and document an overall design that meets the requirements, using software design tools such as IPO diagrams, TOE charts, flow charts, ER-diagrams, UML diagrams (use case and class diagrams), clearly annotated.

The outcome is a plan showing a high-level overview of ***how*** the solution will be constructed using explained/annotated diagrams (or suitable alternative. The plan must show the main blocks/aspects within the proposed solution.

Specify and document:

- The method of solving the problem
- The functions of the constituent parts of the program/system
- The interrelationships between the various parts of the program/system
- The algorithms, data types and data structures as well as any other requirements of the solution
- The effectiveness of the proposed solution in meeting the requirements of the problem

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 instructions.

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

The Delphi/Java program must be able to manipulate the content of database tables, for example update/edit/delete/add data, provide results of queries, provide reports, et cetera.

DESIGN THE GRAPHICAL USER INTERFACE (GUI)

The aim is to produce a GUI design that considers good human-computer-interface (HCI) principles that prevents errors occurring due to invalid input and that minimises 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 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 of 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 designs.

DESIGN THE SOLUTION (FLOW, ALGORITHMS, DATA TYPES/STRUCTURES, ET CETERA)

Use appropriate design tools to design the overall solution, considering all constituent parts and the interrelationships between the various parts of the program/system:

- Description of modular structure of the program/system
- Definition of data requirements
  - Structures
  - File organisation and processing (for example text files)
  - Validation required
- Identification of processes and suitable algorithms for data transformation
- Diagrams/Definitions and details of classes, objects, their attributes and methods
- Description of measures to ensure security and integrity of data.

HAND IN

Hand in a document that provides the following:

- A planned database design
- A GUI design
- A list of data structures and its use in the program
- Software tools used. The IPO design is compulsory PLUS ONE other tool from the given list
- Validation techniques
- Overall solution design

Instructions for Phase 3

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

After completing your project, you will also demonstrate the program and answer questions about your program, the process and the code.

DEVELOP THE DATABASE

Implement the design and construct the database applying appropriate techniques.

Ensure that the database connects correctly to the program and interacts with the program in a meaningful and effective way that supports the solution.

DEVELOP THE GUI

Implement the design by developing the GUI and using appropriate components to ensure easy use and navigation. The user should have a good experience when using the program.

WRITE THE CODE

Use the planning documents of Phase 1 and Phase 2 to write the code for all units/parts.

Use good programming techniques and structures.

Implement effective algorithms and sound defensive programming techniques to produce a robust program.

Document the code so that other people will be able to interpret the program and understand what individual pieces of code do.

The database must be embedded in the program.

DOCUMENT THE PROGRAM

Use any appropriate facility of the programming language to write project notes that a user can access to describe how to interact with the program.

The notes must also describe any known bugs or problems.

Add comments to explain sections of 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/Java project, including the comments and project notes
- The declaration of help received **(Annexure C)**
- The declaration of authenticity **(Annexure D)**

INTERVIEW

Demonstrate the program for evaluation.

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 the 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 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 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 in order to finalise the mark.

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. The PAT furthermore develops learner's life skills and provides opportunities for learners to engage in their own learning

Annexure A: Assessment Tools

Phase 1:	**Name of learner:**						
Scenario (approximately 150 words)	**4**	**3**	**2**	**1**	**0**		
Scenario (Short description)	• Task is clearly stated (purpose and audience) • Thorough understanding of what the problem/task involves. • Explains a possible solution for the problem/task.	• Task is clearly stated (purpose and audience) • Good understanding of what the problem/task involves. • Covers almost all aspects.	• Purpose and audience not that clear. • Shortcomings in understanding and coverage of required aspects.	• Vague, unsure of the purpose of the program. • Minimal understanding of what the problem involves and minimal coverage of aspects.	• Totally inadequate or not applicable • Poor or no coverage of aspects.	**4**	
Evidence of research	**6**	**5**	**4**	**2**	**0**		
Examples: • Documents (pricelist/invoice /reports, et cetera) • Screenhots • Interview • Questionnaires	• Two or more relevant examples given. • Excellent structure. • Supports the summary. • Excellent presentation.	• Two or more relevant examples given. • Good structure. • Supports the summary. • Well presented.	• Two or more relevant examples given • Satisfactory structure. • Supports the summary. • Presentation satisfactory.	• One relevant example given. • Acceptable structure. • No screenshots, printouts, et cetera. • Includes only a few aspects outlined in investigation section	• No evidence of research provided.	**6**	
Report on research (approximately 600 words)	**6**	**4**	**2**	**1**	**0**		
Report: Summary of research	• Extensive research done. • Explains deductions from each research method used. • Clearly explains all key areas. • Excellent, clear direction for the project, for example clearly defines the scope of the program • Conclusion	• A large amount of research done. • Clearly explains all key areas. • Well explained summary of what the program should do. • Shows thorough understanding. • Conclusion.	• Acceptable amount of research done. • Discussed most key areas. • Acceptable summary. • Shows reasonable understanding.	• Limited research done. • Vague, too little coverage of key areas. • Brief, incomplete summary. • Minimal understanding.	• No evidence of research. • No key areas discussed or incorrect and irrelevant or not done.	**6**	

References		**3**	**2**	**1**	**0**		
All relevant research		• All references (at least 3) using Harvard/APA style	• All references (at least 3) but not using Harvard/APA style	• Some (only 2) references included or incorrect style	• No references included	**3**	
User requirements	**7**	**5**	**3**	**1**	**0**		
Role, activity, requirements and limitations of the user (In table format or a use case diagram)	• Role, activity, requirements and limitations of at least 2 different types of users of the system discussed. • Well documented, neat and to the point.	• Some shortcomings in discussion of role, activity, requirements and limitations of at least 2 different types of users of the system. • Documented well but can improve slightly.	• Shortcomings in discussion of role, activity, requirements and limitations of users, for example sections left out. • Only 1 user of the system discussed. • Not well documented but still acceptable.	• Serious shortcomings in discussion of role, activity, requirements and limitations of users for example most of the section were left out or incorrect information. • Only 1 user of the system discussed. • Poorly documented – not acceptable.	• Not done or incorrect or irrelevant.	**7**	
Presentation	**4**	**3**	**2**	**1**	**0**		
Time Management – Phase 1	• Met deadline, all work done.	• One day late but complete.	• One day late and almost complete.	• One day late and sections not done or incomplete.	• More than one day late, or not done	**4**	
					Total	**30**	

Comment/Feedback:

_____

_____

Teacher name: _____ **Teacher signature:** _____ **Date:** _____

Phase 2:	**Name of learner:**					
Database design	**3**	**2**	**1**	**0**		
Choice of fields	• Well-chosen fields. • All fields contribute to the solution. • No redundant fields.	• The majority of the fields contribute to the solution **OR** • One redundant field	• Most fields do not contribute to solution **OR** • More than one redundant field	• No database or incorrect or irrelevant	**3**	
Field types and size	• All fields well-chosen in type and size.	• Most fields well chosen, some poor choices in field types or sizes.	• Poor selection of fields, field types and sizes in most cases.	• No database.	**3**	
Appropriateness – Tables & relationships	• At least two normalized tables for solution. • Appropriate relationship/s between tables. • Keys correctly specified in all tables.	• At least two tables. • Correct use of fields for relationships between tables. • Keys incorrectly used in some instances.	• Only one table/only the fields listed – no relationships. • Correct primary key.	• Only one table/only the fields listed – no relationships. • No/incorrect primary key **OR** • No database	**3**	
Role of database (DB) Describes how the DB will be manipulated, for example within a dataset, access fields and records, navigate records et cetera. Role of manipulation in program described/motivated.	• Clearly describes how the database will be manipulated in the program and how it will contribute to the solution • All manipulation clearly motivated	• Manipulation and interaction mostly well described and reasonably motivated. • Mostly suitable to meet requirements.	• Manipulation not well motivated or mostly not suitable to meet requirements.	• No database or incorrect, irrelevant or unsuitable to the application.	**3**	
GUI design	**3**	**2**	**1**	**0**		
HCI principles Does GUI design consider: • Purpose of program & user • Standard GUI design principles • Ease of use, logical flow • Clearly marked navigation • Friendly dialogue • Help	• Well-designed GUI considering all 6 aspects for at least three of the main interfaces, excluding the introductory screens.	• Satisfactory. • GUI design considers/ meets at least four requirements for at least three of the main interfaces, excluding the introductory screens.	• Limited consideration of requirements • At least three requirements for at least three of the main interfaces, excluding the introductory screens	• Poor GUI design • Less than three requirements considered.	**3**	

Components	**3**	**2**	**1**	**0**		
Components	• Most appropriate components used in all cases. • Excellent layout. • All choices clearly substantiated.	• Appropriate components used in most cases • Satisfactory layout. • Some choices substantiated.	• Appropriate components used in a few cases. • Layout not satisfactory • Mostly not substantiated.	• Mostly inappropriate components used **OR** • Choices not substantiated	**3**	
Data structures used (excluding database)	**3**	**2**	**1**	**0**		
Choice of data structures, for example arrays, text files et cetera (***How*** data will be stored)	• Excellent variety of the most appropriate data structures • All choices clearly contributes to solution, are clearly substantiated and used in a relevant manner.	• Variety of appropriate data structures. • One or two data structures could have been replaced with more appropriate structures. • Most choices substantiated.	• Variety of appropriate data structures. • More than two data structures could have been replaced with more appropriate ones • Some choices substantiated.	• Not described or not substantiated **OR** • No variety	**3**	
Software Tools: **IPO design**	**3**	**2**	**1**	**0**		
Input (***How*** input will be obtained and managed)	• Clearly describes all input: o Format of the input, for example type, size. o Source of input, such as from the keyboard, text file, array or the database. o GUI component used.	• Most input described. • Minor shortcomings in descriptions.	• Only some input described. • Limited description.	• No input requirements described **OR** Incorrect	**3**	
Processing (***How*** processing will be managed – including database manipulation)	• Clearly describes all processing/ manipulation in terms of how data has to be processed/ manipulated (algorithms, formulas, validated et cetera).	• Most processing/ manipulation described. • Minor shortcomings.	• Only some processing/ manipulation described. • Mostly limited, incomplete or incorrect descriptions.	• Processing/manipulation not described. • Incorrect or irrelevant.	**3**	
Output (***How*** output will be managed)	• Clearly describes all outputs o Format of the output, for example type, size o Source of output, for example if the values will be written to a text file, array, database or displayed in the GUI o GUI component used	• Most outputs clearly described. • Minor shortcomings.	• Only some outputs described. • Mostly limited, incomplete or incorrect descriptions.	• Output requirements not described. • Incorrect or irrelevant.	**3**	

(Any ONE other software development tool)	**3**	**2**	**1**	**0**		
Task-object-event chart (TOE), Use case diagrams, class diagrams or entity-relationship diagrams (ERD)	• Excellent use of tool that clearly defines the program.	• Tool constructed with shortcomings.	• Tool not clearly constructed.	• No tool, incorrect use of tool, irrelevant to application.	**3**	
Validation/Error catching	**3**	**2**	**1**	**0**		
(***How*** integrity of input, processing and output will be managed)	• Clearly describes appropriate, meaningful, effective ○ Techniques to ensure IPO integrity ○ Validation/Error catching for relevant IPO ○ Error messages associated with validation/error catching	• Techniques, validation/error catching mostly appropriate and meaningful for IPO	• Techniques, validation/error catching only in some instances appropriate and meaningful for IPO. • Only described once or twice.	• Validation/error catching not described or totally inappropriate.	**3**	
Overall	**4**	**3**	**2**	**0**		
Presentation and time management	• Met deadline. • Excellent presentation. • All work completed.	• Good presentation. • All work completed. • One day late.	• Good presentation. • Some aspects incomplete. • Did not meet deadline. • Not entirely co-operative.	• Did not hand in a phase 2.	**4**	
				Total	**40**	

Comment/feedback:

_____

_____

Teacher name: _____ **Teacher signature:** _____ **Date:** _____

Phase 3: Level 1 – Implementation Name of learner:							
Program aspects	**4**	**3**	**2**	**1**	**0**		
Algorithms What does it do? Does the algorithm solve the task?	• All solution algorithms used in solving the problem are the most appropriate and effective, for example nested If Else-statement used effectively instead of multiple If-statements and works correctly. It enhances the project.	• Appropriate solution algorithms used are effective, with one or two showing minor shortcomings.	• Most solution algorithms used are appropriate and effective	• Mostly inadequate solution algorithms or not effective.	• Totally inadequate solution algorithms. Solution not effective.	**4**	
Control structures (Conditions, iterations, et cetera)	• Used appropriate and most effective control to solve the problem in all instances.	• Used appropriate and most effective control structures to solve the problem in all instances, with minor shortcomings	• Appropriate and effective use of control structures in most instances.	• Inappropriate or ineffective use of control structures in some instances.	• Totally inappropriate or ineffective.	**4**	
Data structures (User-defined, excluding DB)	• Most appropriate and effective data structures, correctly used (for example arrays, text files, et cetera) to solve problem in all instances. • At least one text file.	• Most appropriate and effective data structures, correctly used to solve problem all instances – minor shortcomings	• Appropriate and effective data structures in most instances.	• Appropriate and effective data structures in some instances.	• Totally inappropriate or ineffective or not used.	**4**	
Interactivity/Data flow (User-defined parameter passing)	• Excellent/Proficient interaction between modules with parameter passing in at least two cases.	• Mostly good, proficient interaction between modules with parameter passing in at least two places.	• Some, interaction between modules with parameter passing in one place.	• Limited communication between modules/units/parts.	• No interactivity.	**4**	
Input	• Excellent variety, most appropriate, effective input strategies (database, text files, user input) used in all instances.	• Good variety, appropriately and effectively used in all instances, but with minor shortcomings.	• Appropriately and effectively used in most instances. Limited variety (for example only two types)	• Appropriately and effectively used in some instances • Very limited input (no variety/only one type)	• Totally inappropriate or ineffective	**4**	
Processing	• No logical errors. • All the results of processing are correct.	• One minor logical error • One result problematic.	• Two logical errors. • Some of the results are not correct.	• Many logical errors. • Many results incorrect.	• Many logical errors. • All the results are incorrect.	**4**	

Output	• In all cases: ○ Most appropriate display, well formatted/readable/ structured, for example headings repeated on next page/screen where applicable. ○ Excellent layout	• In all cases: ○ Most appropriate display, well formatted/readable/structured/ understandable, but with minor shortcomings in one instance ○ Good layout	• In most cases ○ Appropriate display. ○ Satisfactory layout.	• In some cases ○ Ouput difficult to read ○ Inconsistent layout	• No output or unreadable	**4**	
Defensive programming; Data validation	• Every effort made to produce a robust program using appropriate defensive programming techniques correctly where necessary.	• Good use of defensive programming where necessary but there are minor aspects that could be improved on.	• Reasonable degree of error checking with a few obvious bugs still present.	• Minimal amount of error checking or defensive programming visible.	• No attempt at validation.	**4**	
Database design and HLL interaction	**4**	**3**	**2**	**1**	**0**		
Database design and manipulation within HLL	• Use of a relevant database with at least two tables and one relationship. • Correct fields, types and sizes. • Excellent, smooth interaction with HLL. • Well-chosen manipulation through HLL code constructs that all contribute to the solution.	• Use of a relevant database with at least two tables and one relationship. • Correct fields, types and sizes. • Good, smooth interaction with HLL. • Good manipulation through HLL code constructs that all contribute to the solution.	• Use of a relevant database with at least two tables and one relationship. • Acceptable fields, types and sizes. • Satisfactory interaction with HLL. • Satisfactory manipulation through HLL code constructs that contribute to the solution.	• Use of a relevant database • One table • Little interaction with HLL.	• No interaction or manipulation or no database.	**4**	

Database manipulation using code constructs.			**2**	**1**	**0**		
The database must be embedded			Correct with no shortcomings	Works with minor shortcomings	Not done or incorrect	**2**	
Insert a new record to a table						**2**	
Delete a record from a table						**2**	
Edit selected fields in a record						**2**	
View all fields/records; View selected fields/records						**2**	
At least two aggregate functions such as minimum, maximum, sum and average						**2**	
At least one query involving two tables						**2**	
Navigate through records in a dataset, from first, to next, to previous records using methods						**2**	
Filter/sort/search (at least one)						**2**	
GUI	**4**	**3**	**2**	**1**	**0**		
Ease of use / HCI principles Screens well planned and designed • Excellent communication (screen tips, feedback, help et cetera) • Most appropriate components • Readable/understandable output • Excellent use of effects/ colour/icons/shortcuts, tool tip text, et cetera	Excellent – all four aspects have been met and correctly used.	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.	Poor GUI design. Little/no thought given to HCI principles.	**4**	
Dynamic components	At least two different dynamically instantiated components, both meaningfully and correctly used	At least one dynamically instantiated component, meaningfully and correctly used	Dynamic component(s) used meaningfully but does not work correctly	Dynamic component(s) used that work correctly but not meaningful	No dynamic component.	**4**	

Documentation	4	3	2	1	0		
Comments/Notes (Explanation of program and code)	• Code clearly annotated to explain all necessary parts. • Explanation shows excellent insight. • Extensive project notes present and of an excellent standard. • Clearly explains working of the program.	• Code clearly annotated to explain all necessary parts. • Explanation shows good insight. • Project notes present and of a very good quality.	• Code annotated to explain most necessary parts. • Explanation shows some insight. • Project notes present 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**	
Scope of variables		**3**	**2**	**1**	**0**		
		Use of local and global variables appropriately and effectively – enhances program.	Use of local and global variables; but not always appropriately used.	Limited number of variables – local only or global only	No variables used or inappropriate use of variables.	**3**	
Manipulating string processes		**3**	**2**	**1**	**0**		
		Combine multiple string methods to do complex manipulation, for example generate code/key, extracting parts from several DB fields/ variables using a combination of several string methods and/or calculations	Standard – Combine at least two string methods	Simple – single manipulation only (only use one string method)	No string methods used.	**3**	
Modular programming (Re-use of code)		**3**	**2**	**1**	**0**		
		Re-use of code – Good use of functions/ procedures/methods/ objects/parameter passing	Re-use of code – Good use of functions/procedures/ methods/objects/ parameters. Limited use of parameter passing.	Used – inappropriate or no parameter passing or not working correctly	No evidence of modular programming	**3**	

Overall	**4**	**3**	**2**	**1**	**0**		
Does the program meet the requirements?	• Well exceeds requirements stated in Phase 1 • Comprehensive program, all elements function as specified. • Shows insight in all aspects.	• Exceeds requirements stated in Phase 1. • Less comprehensive. all elements function as specified. • Shows insight in most aspects.	• Slightly exceeds requirements. • Some program elements function as specified in Phase 1. • Shows insight in one or two aspects.	• Meets minimum requirements. • Basic program. • Basic scope. • Very limited insight.	• Does not meet minimum requirements. • Less than basic. • Limited scope.	**4**	
Presentation	**1**			**0**			
Time Management – Phase 3	Met deadline, all work done			Did not meet deadline or not given in		**1**	
					Total (implementation):	**80**	

Higher-order Programming:

The table below determines the complexity level of the program to discriminate between different levels of programs.

Phase 3: Level 2 – Higher-order programming skills Name of learner:							
Aspects	**4**	**3**	**2**	**1**	**0**		
Play a video/thread/ networking mobile app/ a time-based simulation appropriately or any form of complex code. Borrowed code allowed.	• Works correctly, is relevant and appropriate. • Adds value to the solution.	• Works but could be improved. • Relevant and appropriate.	• Works with minor shortcomings. • Relevant and appropriate.	• Works with major shortcomings. • Relevant and appropriate.	• No attempt has been made.	**4**	
Display a drawing/ animation/graph/map or timer	• Non-trivial, excellent layout and placements. • Works correctly, adds value to the solution. • Relevant and appropriate.	• Good layout and appropriate placement. • Works but could be improved. • Relevant and appropriate.	• Satisfactory layout and placements. • Works with minor shortcomings. • Relevant and appropriate.	• An attempt has been made, with major shortcomings.	• No attempt has been made.	**4**	
Calculations	• Three correct calculations including one complex calculation.	• Three correct calculations.	• -Two correct calculations.	• One correct calculation.	• No calculations.	**4**	
SQL statements	• Three different correct SQL statements used, one of which must use data from two tables.	• Three different, correct SQL statements used, using one or two tables.	• Two different correct SQL statements using one or two tables.	• One correct SQL statement using one or two tables.	• No SQL statements or all statements incorrect.	**4**	
Database **Excluding CRUD operations** **Complex code and data volume**	• At least 10 records in a table. • Non-trivial input/output and processing, for example data extracted must be processed further (transformed) to obtain desired result, assigning query parameter values at runtime; modular approach, for example use of procedures/ functions/separate units. • Dynamically parameterised queries (parameters set at runtime, embedded in programming language statements by learner).	• Standard, significant data volume. • Complex, non-trivial programming code used but could be improved.	• Limited data , involve only data storage/retrieval and possibly trivial processing by the package virtually no own code – limited use of programming language • Complex code used with minor shortcomings	• Only one record in table • Complex code attempted with major shortcomings	• No data in tables • No complex code attempted	**4**	
					Total (Higher-order programming skills)	**20**	

General: Final product and impression	**Name of learner:**						
Aspect	**5**	**4**	**3**	**2**	**1**	**0**	**Mark**
Flow of development	• Each phase of development flows logically from the previous phase. • Did not deviate from original scope. The initial goal was reached and all the set requirements stated in Phase 1 were met.	• -Had to re-address minor issues and goals from previous phases. • -At least 80% of requirements were met. • -Some aspects initially planned were not accomplished.	• Had to re-address a number of issues and goals from previous phases. • More than 50% of requirements were met. • Some aspects had to be changed or scaled up or down.	• More than 50% of original requirements were not met. • Many of original aspects had to be changed or scaled up or down.	• A few of the original requirements were met.	No requirements met	
Professional project	• Useful and can be implemented as a real-life application. • Compares well with proprietary software. • Contains professional features, for example help functions, well-designed GUI layout, no unexpected errors, user-friendly feedback on all aspects and features.	• -Can be implemented as a real-life application with minor adjustments. • -Compares satisfactorily with proprietary software. • -Contains professional features for almost all aspects, for example help functions, well-designed GUI layout, no unexpected errors, user-friendly feedback on most aspects and features.	• Can be implemented as a real-life application with significant adjustments. • Compares to a lesser extent with proprietary software. • Contains professional features for a satisfactory number of aspects, for example help functions, well-designed GUI layout, no unexpected errors, user-friendly feedback on some aspects and features.	• Not ready to be implemented as a real-life application, but has some potential. • Does not compare with proprietary software. • Contains limited professional features, for example help functions, well-designed GUI layout, no unexpected errors. User-friendly feedback on limited aspects and features.	• Not ready to be implemented as a real-life application. • Contains limited professional features, for example help functions, well-designed GUI layout, no unexpected errors. User-friendly feedback on limited aspects and features.	No requirements met	
Completeness	• All phases of development were complete and well designed and executed. • All stages and phases well documented.	• Almost all phases of development were complete and well designed and executed. • Mostly all stages and phases well documented.	• At least two phases of development were complete and well designed and executed. • At least two stages and phases well documented.	• One phase of development was complete and well designed and executed. • One stage and phase well documented.	• None of the phases of development were complete and well designed. • None of the phases were well documented.	No requirements met	

Attitude and commitment	• All phases were handed in on time and were well designed. • Work was done regularly. Showed exceptional commitment and pride in work done during each stage. • Showed exceptional growth in knowledge and skills.	• Almost all phases were handed in on time and were well designed. • Work was done regularly. Showed commitment and pride in work done during each stage. • Showed definite growth in knowledge and skills.	• At least two phases were handed in on time and were well designed. • Work was done with intervals. Showed some commitment and pride in work done. • Showed some growth in knowledge and skills.	• At least one phase was handed in on time and was well designed. • Work was not done regularly. Showed limited commitment and pride in work done. • Showed limited growth in knowledge and skills.	• None of the phases handed in on time. Not well designed • Work was not done regularly. Showed no commitment and pride in work done. • Showed no growth in knowledge and skills.	No requirements met	
					Maximum: 20/2 = 10		

Adjustment %

Interview	**100% of total mark**	**90% of total mark**	**75% of total mark**	**60% of total mark**	**50% of total mark**	
Explain selected code	• Explained all selected code clearly and with confidence • Shows excellent insight.	• Explained 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 • Lacks insight	• Unable to explain any selected code • No insight	**%**
					Final Project Mark:	

Assessment Summary

Phase	Focus	Maximum Mark	Mark Obtained
Phase 1	Analysis	30	
Phase 2	Design	40	
Phase 3	Coding and Implementation	80	
Phase 3	Higher order programming skills	20	
General	Final product and impression	10	
Total		180	
Adjustment %			
Final mark (Total x Adjustment %)			

Declaration of Authenticity

I hereby declare that the work assessed is solely that of the learner (except where there is clear acknowledgement and record of any substantive advice/assistance given to the learner) concerned 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:** _____

Annexure B

Anatomy of a report

This document informs you of the specific requirements of the report.

If you follow these guidelines you will be assured of a successful report. You are expected to use a style sheet so that your report is neat as well as correctly structured. See column 1 below for formatting suggestions. (You can generate a table of contents automatically if you use formatting as outlined in column 1).

TITLE PAGE <title>	Report title Your name and grade submission date
SUMMARY <heading>	Overview of the report's essential information and recommendations
TABLE OF CONTENTS <table of contents>	List of numbered sections in the report and their page numbers
INTRODUCTION <heading1>	Terms of reference, the scenario and outline of the report's structure
BODY **Headings** <heading1> ***Subheadings*** <heading2>	Headings and subheadings which reflect the contents of each section. Includes information on important ideas about the topic, and a discussion of programs relevant to the scenario.
CONCLUSION <heading1>	States the conclusions that can be drawn from the information found. Makes recommendations about the direction the learner will follow in the project.
REFERENCE LIST <heading1>	List of reference material consulted during research for report. Use the Simplified Harvard style/APA style
APPENDIX <heading1>	Pictures and information that support your research but is not essential to your explanation.

Annexure C

Learner declaration – Phase ____

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

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

☐No ☐Yes (provide details below)

Help/information received from (person):	Nature of the help/information (provide evidence):

_____ ___ / ___ / 2016

SIGNATURE OF LEARNER DATE

Annexure D

Declaration of authenticity

Learner name		**ID Number**	
Grade	12	**Year**	2016
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 anyone.

_____ ___ / ___ / 2016

SIGNATURE OF LEARNER DATE

Guidelines for teachers to provide guidance

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

How will learners go about it?

The learner 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 appropriate format.

Skills required

The learner must be able to:

- Do research on the topic and document research findings properly, including citations as specified in the Phase 1 marking grid.
- 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

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 research, analysis and execution
- Solution development content and skills, including the ability to define a task
- Project management skills, including time, resource and task management
- The format and structure of accepted forms of research report to include the abstract, introduction, discussion with all sources cited, conclusion, references

Malpractice

Learners may NOT:

- Get help/guidance from others without acknowledgment (complete **Annexure C** 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 candidates may not lend their books to another candidate, but candidates 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 a penalty will be applied.

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.

Learner declaration of authenticity of the PAT

For each phase, learners complete a declaration (**Annexure C**) 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 D**) to confirm that the work submitted is their own.

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, for example where appropriate, record their own initial question with clear evidence of the guidance and the final question)
- Give regular feedback to learners, for example 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 allow the learner to investigate and to access the higher-level concepts and skills in the assessment objectives, for example 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 which can be managed within the timeframe and available resources?
- Do the focus 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?

The teacher will authenticate the PAT:

- 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
- Teacher will sign the assessment tool of each phase

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.

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 chooses this option, he/she should start with the PAT process towards the end of the first term, completing one phase per term.

Option 2:

- The teacher could dedicate a continuous period of time to the PAT, for example the last week(s) of each term, also completing one phase per term.

Evidence of assessment

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

The evidence for assessment will include the following:

- The project product, including a written report of approximately 600 words (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)

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 3 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- or 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.

Requirements

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

Practical Assessment Task components must:

- Comprise assessment tasks that constitute the learners' PAT mark as contemplated in Chapter 4 of the Curriculum and Assessment Policy Statement for IT
- 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 Curriculum and Assessment Policy Statement for IT
- Be available for monitoring and moderation
- Be evaluated, checked and authenticated by the teacher before being presented as the learner's evidence of performance

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 three weeks before the commencement of the final end-of-year examination to submit outstanding work or present himself or 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 or 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 himself or 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 includes 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 as may be accepted as valid by the head of the assessment body or his or her representative