basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

# INFORMATION TECHNOLOGY

## GUIDELINES FOR
## PRACTICAL ASSESSMENT TASK

## 2018

**These guidelines consist of 31 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 an opportunity to demonstrate your programming skills and understanding of the connections between the different areas of solution development.

You will also be required to demonstrate your 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.

You need to provide the following outputs:

- Phase 1:

  A report where you
  - Provide a brief description of the problem to be solved
  - Discuss the research/investigation/analysis done regarding the project
  - Provide a brief description of the purpose and scope of your project
  - Do a detailed design of your solution
- Phase 2:

  A working Delphi program that implements the planned solution

**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** | Analysis and Design | **60** | **40** |
| **Phase 2** | Coding and Testing | **72** | **48** |
| **General** | Final product and impression | **18** | **12** |
| | **Total:** | **150** | **100** |

**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.

You need to complete your PAT before you start the Grade 12 end-of-year examination. Not submitting your PAT or any part of the PAT, will mean that you will be awarded a zero ("0") for the PAT component of the examination or the parts not submitted.

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

Phase 1: Not later than the end of Term 1
Phase 2: Not later than the first week of Term 3.

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

# The Topic

**The food, beverage and confectionary industry**

**NOTE**: ***The topic EXCLUDES any form or reference to alcoholic beverages. In the discussion and examples below, the term 'beverage' refers to cool drink.***

The food, beverage and confectionary industry is a significant, global collective of diverse businesses that supplies the food, beverages and confectionary consumed by the world's population. This industry ranges from simple home industries to modern, highly technological companies manufacturing and transporting food supplies. Some of the activities are also directed at the preparation, preservation, serving, packaging and the retailing of food supplies.

You are required to plan and develop **a software program** that is associated with any aspect related to the food industry.

Examples of topics are:

- Farming and agriculture: vegetables, dairy, chicken, cattle, etc.
- Industrial fishing
- Companies manufacturing packaging required for food, beverages and confectionary
- Home industries/Service providers used for baking and the hosting of functions
- Modern technology and equipment used/supplied for manufacturing food
- Transporting food and beverage supplies
- Supermarkets that retail food and beverages
- Stock control at warehouses that distribute food and beverages
- Restaurants and food outlets
- Magazines and recipe books used for the preparation of food
- Jobs, careers related to this industry
- Health companies that carry out hygiene tests on food
- Chemicals used in preserving food
- Import and export of food and beverages
- Advertising, marketing, television and radio shows

Choose an application/environment that relates to the food, beverage and confectionary industry and do research on the information system requirements.

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

 

**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 2 marking guidelines.

**NOTE:** Your final project must consist of **one** single program, with logically related parts.

# Overview

## PHASE 1 – ANALYSIS AND DESIGN

The purpose of this phase is to analyse the problem, i.e. to determine ***what*** should be done and ***what*** the user requirements are. This includes the following activities:

- Identify the problem
- Research/Investigate the topic/scenario/the (potential) users from whom facts about the nature/functionality of the program you intend to produce can be gathered
- Define the task by determining the following:
    - WHAT will the system do?
    - HOW will the system be used?
    - WHO will use the system?

This will be followed by determining ***how*** the program/system will meet the requirements and the planning and design of the solution to the problem:

- Clarify the requirements by using available software design tools to design the solution, clearly indicating the logical program flow and navigation between screens.
- Design the database and other data structures to be used.
- Design the GUI(s) and data capturing mechanisms.
- Identify what processing will need to be done and define how the processing will be done.
- Indicate what the output will be and design appropriate data output components/ mechanisms.

## PHASE 2 – CODING AND TESTING

The purpose of Phase 2 is to implement the design by writing the code and testing the program. The following must be done:

- Write the programming code to implement the design and to complete the program.
- Test and debug the program.
- Clarify sections of the code by adding comments.
- Write project notes for the program.
- Demonstrate your program and answer questions about the program and the code during a debriefing session.

 

# PAT requirements

The project should include the following, appropriately integrated:

- Database manipulation through programming language constructs and SQL statements
- Text file(s) for input-output purposes, e.g. to populate data structures, provide reports
- A multi-form/multi-screen GUI with good functionality and usability, based on sound HCI principles
- Manipulation/Transformation of data through:
    - Mathematical/Statistical processes
    - String/Text processes

**Database**

The database component:

- Must be accessed and manipulated using your own programming code as well as using SQL statements
- Should not only involve data storage/retrieval and possibly trivial processing by the package with virtually no code of your own (resulting in a simple solution)
- Must include at least TWO relational tables
- Must involve sufficient data volumes and use a variety of field types

**Text file**

Your program must use input from a text file to manage data that could have been captured elsewhere. This must include mechanisms to transfer and process data between the database and text file/s.

The data from the text file could be used to:

- Do calculations and manipulations in combination with data from the database
- Update existing data (add records, delete records, update records, etc.)

At least ***one*** report should be provided in a text file format.

**GUI**

The graphical user interface (GUI ) must be functional and based on sound HCI principles.

The GUI must have at least:

- Three forms/screens
- One component created dynamically

**Variables and data structures**

- Use appropriate variables and data structures
- Carefully consider the scope of variables (local vs. global)

**Modular Programming**

Incorporate your own methods/procedures/functions, e.g. to validate data or to transform/ manipulate data

**Further requirements**

*Apply good programming principles and techniques*

- Descriptive names for variables, data structures, fields, components, etc.
- Well-structured, readable code
- Write comments/annotations to explain pieces of code, especially on the manner in which variables/data structures and output variables/components are used.

*Write project notes*

- Describe how to use/interact with the program
- Describe any known bugs or problems

Project notes can be written as a help function that is part of the program

**General programming aspects that will be assessed:**

- Programming style
- Graphical user interface (GUI)
- Use of human-computer interaction (HCI) and software engineering principles
- Use of OOP principles
- Functionality of the program
- Level of expert programming
- Robustness of the program, including the use of defensive programming techniques
- Whether the project matches the original aims and goals
- Internal documentation to explain sections of the program

# What you need to be able to do the PAT

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

- 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, etc.

# 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
- Submit work which is not your own e.g. programming code that was written by someone else
- Share your PAT work with other learners
- Allow other learners to obtain or use your research/code (this does not mean that you may not exchange books/links to informative websites)
- Use research or code that other learners obtained and used in their PAT projects
- 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 20% 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

The aim of Phase 1 is to:

- Define the task in your own words
- Do research in order to get a clear understanding of the problem/task at hand
- Analyse the problem/task at hand by doing research to identify what the system should do, how the system will be used and who the users would be. If possible, consult with them to determine the requirements/functionality of the program.
- Determine ***HOW*** you will go about solving the problem and plan the detail.
- Produce the plan showing the high-level overview of how the solution will be constructed using explained/annotated pseudocode/diagrams (or a suitable alternative). The plan must show the main blocks within the proposed solution.
- Specify and document an overall design that meets the requirements using an IPO table as a software design tool.

## DEFINE THE TASK

The aim is to provide an overall picture of the purpose and scope of the project, but *not the details*.

Write a brief description (approximately 100 words) in your own words to describe the problem/task in general terms and how the project will solve the problem. In other words, the description should clearly indicate that:

- You understand the requirements of the project
- Your solution will address the requirements
- Your project adheres to the PAT requirements/specification.

## DO RESEARCH

The aim is to gather basic background knowledge and facts about the identified task and the nature of the program you are developing.

Your research should help you to

- Understand the topic/scenario
- Clarify the nature/type of program you are required to develop
- Look at some existing solutions, where possible, to gather ideas
- Understand what particular type of program(s) is suitable for this project

The outcome is a report (± 150 words/1 page) that indicates, for example, what a software solution for the problem may include, similarities, missing features and general working/flow of existing solutions. The report should be written in clear unambiguous language.

 

## DO THE ANALYSIS

The aim is to

- Present 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.
- Provide an analysis of **how** the system will be used
- Determine **who** will use the system, identify the prospective user(s) and identify the user needs and acceptable limitations.

## BASIC SOLUTION DESIGN

Use a diagram/illustration to show the flow of the PAT from the start to the end. The diagram must illustrate the use of tabs/forms, navigation and any other significant aspects of the design.

Use an IPO-diagram as a software design tool to clearly design/indicate the logical program flow and navigation between screens.

The aim is to:

- Determine how you will go about solving the problem and plan the detail.
- Produce the plan, showing a high-level overview of how the solution will be constructed and using explained/annotated pseudocode/diagrams (or suitable alternatives). The plan must show the main blocks within the proposed solution.
- Specify and document an overall design that meets the requirements, using an IPO-diagram.

## DATA DICTIONARY

Design a database to serve as data source that can provide data to manipulate/transform, e.g. through mathematical/statistical processes using programming instructions.

Clearly indicate the fields, the field types and the primary key.

Describe the role of the database in the program.

The Delphi program must also be able to manipulate the contents of the database tables, e.g. update/edit/delete/add records, provide results of queries, provide reports as a result of processed/manipulated data, etc.

Plan and design other data structures that will be used. These include objects, arrays and text files. Clearly indicate how and where these data structures will be used.

 

## DATA CAPTURING (INPUT)

Design a GUI that considers good human-computer-interface (HCI) principles, that prevents errors occurring from invalid input and that minimises the amount of information a user has to input.

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

- The user – type and context
- User requirements/needs, usability
- Dialogue – must be relevant, simple and clear
- Icon usage and presentation – well selected and relevant, well placed with a clear purpose
- Colour – use and combination of colour
- 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 usage of space on the screen

Provide sample(s) of planned data capture and data entry designs (prototype screen dumps may be used but must be annotated) and of planned valid output designs.

Plan and design how validation and error catching will be handled for all relevant components.

## DATA PROCESSING

Identify and list the processing that will need to be done. Design the necessary algorithms to do the required processing.

Provide:

- A detailed list of all processing to be done
- Algorithms/Methods that will be used to do the required processing. This can include:
    - Specific formulas and/or functions that will be used
    - Pseudocode solutions and diagrams

## DATA OUTPUT

Identify and list the output that will be required. Select/Design the necessary output components and how the output will be formatted.

Provide:

- A detailed list of all required output
- Output components that will be used and how output will be formatted

## HAND IN

Once you have completed Phase 1 of the project, submit a detailed report that covers the following:

- Problem identification
- Research/Investigation on the topic/scenario/(potential) users to gather facts about the nature/functionality of the program you intend to produce.

  This should include proof of research with proper referencing.
- A detailed task definition/analysis indicating the following:
  - WHAT will the system do?
  - HOW will the system be used?
  - WHO will use the system?
- Basic design –flow diagram
- Data dictionary – including detailed database design
- GUI design
- IPO
  - Data input – what will be the input, validation and error catching techniques
  - Data processing – what will need to be processed, how will it be processed
  - Data output – required output, components and format of output

**NOTE:** Your report should NOT include that you must write a Delphi program that should be user-friendly and include a database. These are obvious requirements. You should focus on the ONE specific problem you have chosen.

Also include your declaration of authenticity for Phase 1 (**Annexure A**).

# Instructions for Phase 2

The aim is to implement your design by using appropriate software tools (programming language, database software, IDE, etc.) and to use techniques to code 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 by 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.

## TEST THE PROGRAM/SYSTEM

Perform tests to determine:

- The functionality of the program/system – to confirm that the program/system satisfies the system requirements (user acceptance testing)
- Whether units of code (single functions, procedures, interface(s), etc. – one feature at a time) are working correctly (unit testing).

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

Compare test input results with expected results to determine success or failure.

Debug where necessary.

## 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.

## HAND IN

On completion of Phase 2, hand in the following:

- The completed Delphi project, including the comments and project notes
- The declaration of authenticity **(Annexure A)**

## 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.

 

# Assessment Tools

| **Phase 1 - Analysis and Design** | | | **Learner name:** | | | | |
|---|---|---|---|---|---|---|---|
| **Investigation:** | **4** | **3** | **2** | **1** | **0** | | |
| **Problem identification** | Clearly outlines the current problem/task. | Acceptable outline of the current problem/task. | Limited outline of the current problem/task. | Weak outline of the current problem/task. | No clear problem/task defined. | **4** | |
| **Report on research** | Extensive research done. Research is relevant. Proof of own research and/or at least 3 references. | Large amount of research done, but with limited relevance. Proof of own research and/or at least 2 references. | Research done, but not always relevant. Limited references/proof | Limited research and/or references | No research/references included | **4** | |
| **Problem analysis/definition** | **4** | **3** | **2** | **1** | **0** | | |
| **WHAT will the system do** | Clear description covering all detail. | Clear description with minor detail missing | Complete description, but lacks clarity. | Incomplete description, with limited use. | Incorrect, irrelevant or not described at all. | **4** | |
| **HOW will the system be used** | Clear description covering all detail. | Clear description with minor detail missing | Complete description, but lacks clarity. | Incomplete description, with limited use. | Incorrect, irrelevant or not described at all. | **4** | |
| **WHO will use the system** | Clear description covering all detail. | Clear description with minor detail missing | Complete description, but lacks clarity. | Incomplete description, with limited use. | Incorrect, irrelevant or not described at all. | **4** | |
| **BASIC DESIGN** | **4** | **3** | **2** | **1** | **0** | | |
| **Navigation/Description** **A diagrammatical representation of the design and flow of the PAT** | An excellent attempt to show the sequence of all steps and execution of processes for the solution with no shortcomings. | A good attempt to show the sequence of all steps and execution of processes for the solution with no shortcomings. | A satisfactory attempt to show the sequence of all steps and execution of processes for the solution with shortcomings. | A poor attempt to show the sequence of all steps and execution of processes for the solution. | No diagram OR incorrect, irrelevant or unsuitable for the application. | | |
| **DATA DICTIONARY** | **4** | **3** | **2** | **1** | **0** | | |
| **Choice of data structures**, Arrays, text files, variable, etc. (***How*** data will be stored) | A variety of data structures that all contribute to the solution and are clearly defined. | A variety of data structures included, but is not always the best option for solution or not clearly defined. | Data structures include not only appropriate variables but also other structures like text files or arrays. | Data structure definition covers basic variables, with appropriate types that are needed for solution. | No or poor definition of data structures used. | **4** | |

 

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Database basic design**<br>• All fields are relevant<br>• Type and size of fields well-chosen | All database design requirements met. | Good database design with minor shortcomings. | Average design with several shortcomings. | Database design done, but with limited value. | No database or incorrect or irrelevant. | **4** | |
| **Database advanced design**<br>• Relational<br>• Normalised | A well-designed relational database normalised appropriately. | A relational database normalised with minor shortcomings. | A relational database with more than one inappropriate field. | A poor attempt to normalise a relational database. | No relational database<br>Database is not normalised | **4** | |
| **GUI DESIGN** | **4** | **3** | **2** | **1** | **0** | | |
| • Design fits to program's intended use<br>• Appropriate components<br>• Ease of use, logical flow<br>• Clearly marked navigation<br>• Friendly dialogue / Help | Good GUI design, considering all of the principles in a variety of applications eg. Data capturing, output, navigation | Satisfactory GUI design, considering most (at least 4) of the principles in a variety of applications eg. Data capturing, output, navigation | Limited GUI design, considering (at least 3) of the principles in a variety of applications eg. Data capturing, output, navigation | Poor GUI design considering less than 50% (less than 2) of the principles. | GUI design is not functional or does not support the intended use at all. | **4** | |
| **Software Design Tool: IPO** | | | | | | | |
| **DATA INPUT** | **4** | **3** | **2** | **1** | **0** | | |
| **Input data**<br>(**What** input will be provided)<br><br>**Input Format and source**<br>(***How*** input will be stored) | Clearly describes all inputs in terms of the data that will be stored, sources and format of input. | Minor shortcomings in describing all inputs in terms of the data that will be stored, sources and format of input. | Limited description. More than two inputs not clearly described. | Poor description. Most inputs not clearly described. | Input requirements not described, incorrect or irrelevant | **4** | |
| **Validation/Error catching**<br>• Validation/error catching for relevant input<br>• Associated error messages | Clearly describes appropriate and effective techniques, to ensure only valid data is entered. | Techniques, to ensure only valid data is entered are mostly appropriate and effective | Validation/error-catching limited and sometimes inappropriate/not meaningful. | Validation/error catching poorly described or inappropriate/not meaningful. | No effort at validation | **4** | |
| **DATA PROCESSING** | **4** | **3** | **2** | **1** | **0** | | |
| **WHAT** processing will need to be done | Clearly lists all processing/ manipulation that will need to be done. | One or two processing/manipulations not correctly listed. | Most processing/manipulations correctly listed. Limited detail. | Many processing/ manipulations not clearly listed. | Most Processing/manipulations not listed. | **4** | |

 

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **HOW** processing will be managed - algorithms, formulas, etc.) | Clearly describes how all processing/manipulation will be done. | One or two processing/manipulations not clearly described. | Some processing/ manipulations not clearly described. Mostly correct. | Many processing/ manipulations not clearly described. Many errors. | Processing/manipulation not described, incorrect or irrelevant | **4** | |
| **DATA OUTPUT** | **4** | **3** | **2** | **1** | **0** | | |
| **Output Data** (**What** output will be provided)<br><br>**Output Format** (***How*** output will be presented) | Clearly describes all outputs in terms of the data that will be provided and how sources of output will be used and formatted. | Minor shortcomings in descriptions. One or two outputs not clearly described. | Limited descriptions. More than two outputs not clearly described. | Poor descriptions. Most outputs not clearly described. | Output requirements not described, incorrect or irrelevant | **4** | |
| | | | | | **Total:** | **60** | |

| **Phase 2 Coding:** | **Learner name:** | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| **DATABASE DESIGN** | **4** | **3** | **2** | **1** | **0** | | |
| **Implementation of Database design** | Database design correctly implemented, with at least two relational tables, suitable fields, data types and sizes.<br>Large/Adequate data volume used. | Database design correctly implemented, with at least two relational tables, suitable fields, data types and sizes.<br>Limited volume of data used. | Uses at least two tables, but database design not properly implemented. Errors in fields, data types and sizes OR<br>Small amounts of data used. | Use of a relevant database.<br>One table.<br>Few 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**<br>• Excellent layout and communication (screen tips, feedback, help et cetera)<br>• Most appropriate components<br>• Readable/Relevant input/output<br>• Excellent use of effects/ colour/ icons/shortcuts/ tool tip text, et cetera | Excellent – all four aspects have been met and used correctly. | 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** | |
| **DATA STRUCTURES** | **4** | **3** | **2** | **1** | **0** | | |
| **Variables used** | Variety of appropriate variable types, correct use of local and global variables, proper naming convention of variables and components | Variety of variable types, use of local and global variables, Choice, use and naming of variables not always suited | Limited variety of variable types and use of local and global variables AND/OR poor naming of variables. | Poor variety of variable types.<br>Incorrect use of local and global variables. Poor naming of variables. | Totally inappropriate or incorrectly used | **4** | |
| **Data structures used**<br>(User defined, excl. DB) | Used at least two data structures (e.g. array and text file) correctly and appropriately | Used at least two data structures (e.g. array and text file) with minor shortcomings | Used at least one data structure (e.g. array or text file) correctly and appropriately | Used at least one data structure (e.g. array or text file) but not always correctly and appropriately | Totally inappropriate or incorrect or not used | **4** | |

 

| INPUT | 4 | 3 | 2 | 1 | 0 | | |
|---|---|---|---|---|---|---|---|
| **Input data**<br>(Mechanisms/components, type, format etc.) | Variety of input mechanisms used.<br>Most appropriate type, format and components used. | Different input mechanisms used, not always the most appropriate type, format or components. | Limited variety of input mechanisms used, often not the most appropriate type, format or component. | Limited variety of input mechanisms used, mostly not the most appropriate type, format or component. | Totally inappropriate or ineffective | **4** | |
| **Validation/Error catching** | Extensive validation/error catching for relevant input<br>Clear and appropriate error messages and exception-handling mechanisms. | Variety of validation/error catching for relevant input<br>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 described or inappropriate/not meaningful. | No effort at validation | **4** | |
| **PROCESSING** | **4** | **3** | **2** | **1** | **0** | | |
| **Algorithm correctness/ Processing** | All solution algorithms used are appropriate, work correctly and meet all processing requirements. | Appropriate solution algorithms that work correctly but one processing requirement not met. | Most solution algorithms used are appropriate and work correctly and meet most processing requirements. | Solution algorithms are mostly inadequate or mostly not working correctly, processing requirements not all met. | Totally inadequate or not working correctly | **4** | |
| **Algorithm efficiency** | All algorithms used are the most efficient solution.<br>Algorithms use good programming techniques.<br>Effective modular design with correct use of own functions and procedures | Most algorithms used are the most efficient solution.<br>Algorithms use acceptable programming techniques.<br>Limited modular design with correct use of own functions and procedures | Limited efficiency of algorithms used.<br>Few algorithms use good programming techniques.<br>Poor modularity with limited use of own functions and procedures | Poor efficiency of algorithms used.<br>Algorithms do not use good programming techniques.<br>Attempted use of own functions and procedures | Totally inadequate or not working correctly | **4** | |
| **Use of creative/complex code to create a graph/map or timer** | • Non-trivial, excellent layout and placements.<br>• Works correctly, adds value to the solution.<br>• Relevant and appropriate. | • Good layout and appropriate placement.<br>• Works but could be improved.<br>• Relevant and appropriate. | • Satisfactory layout and placements.<br>• Works with minor shortcomings.<br>• Relevant and appropriate | • An attempt has been made, with major shortcomings. | • No attempt has been made. | **4** | |

| **OUTPUT** | **4** | **3** | **2** | **1** | **0** | | |
|---|---|---|---|---|---|---|---|
| **Output** | In all cases:<br>• Excellent display, well formatted/readable/ structured, for example headings repeated on next page/screen where applicable.<br>• Excellent layout | In all cases:<br>• Most appropriate display, well formatted/readable/ structured, but with minor shortcomings in one instance<br>• Good layout. | In most cases<br>• Appropriate display.<br>• Satisfactory layout. | In some cases<br>○ Ouput difficult to read<br>○ Inconsistent layout | No output or unreadable | **4** | |
| **Database - High Level Language interaction** | **4** | **3** | **2** | **1** | **0** | | |
| **Effective use of data queries** | Excellent use of code constructs to view, navigate data.<br>Good use of code to filter/sort/search when selecting data. | Good use of code constructs to view, navigate data.<br><br>Adequate use of code to filter/sort/search when selecting data. | Adequate use of code constructs to view, navigate data.<br>Limited use of code to filter/sort/search when selecting data. | Limited use of code constructs to view, navigate data.<br>Few examples of using code to filter/sort/search when selecting data. | Totally inadequate or not working correctly | **4** | |
| **Complexity of data queries** | Excellent use of processing/data transformation.<br>Excellent use of calculated values. Variety of complex queries, including the use of relationships between tables. | Good use of processing/ data transformation.<br>Good use of calculated values.<br>Variety of complex queries. | Adequate use of processing/ data transformation.<br>Limited use of calculated values.<br>Few complex queries. | Limited use of processing/ data transformation.<br>No calculated values.<br>No complex queries. | Totally inadequate or not done. | **4** | |
| **Data maintenance - Insert/Update/Delete records** | Uses all data maintenance actions.<br>All actions correctly used.<br>All applications contribute to a functional system. | Uses all data maintenance actions.<br>All actions correctly used.<br>Some applications do not contribute to a functional system. | Uses all data maintenance actions.<br>All actions correctly used.<br>Most applications do not contribute to a functional system. | Uses some data maintenance actions.<br>All actions not correctly used.<br>Applications do not contribute to a functional system. | Totally inadequate or not done. | **4** | |
| **Database - SQL** | **4** | **3** | **2** | **1** | **0** | | |
| **Effective use of SQL in queries.** | Excellent use of a variety of SQL statements to filter/sort/search when selecting data. | Good use of SQL statements to filter/sort/search when selecting data. | Adequate use of statements to filter/sort/search when selecting data. | Limited use of filter/sort/search when selecting data. | Totally inadequate or not working correctly | **4** | |

 

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Complexity of SQL interaction** | Excellent use of processing/data transformation. Excellent use of formatting for calculated fields.<br>Variety of complex queries, including the use of relationships between tables. | Good use of processing/data transformation.<br>Good use of formatting for calculated fields.<br>Variety of complex queries. | Adequate use of processing/ data transformation.<br>Limited use of formatting for calculated fields.<br>Few complex queries. | Limited use of processing/ data transformation.<br>Poor formatting of calculated fields.<br>No complex queries. | Totally inadequate or not done. | **4** | |
| **Data maintenance - Insert/ Update/Delete records using SQL** | Uses all data maintenance actions.<br>All actions correctly used.<br>All applications contribute to a functional system. | Uses all data maintenance actions.<br>All actions correctly used.<br>Some applications do not contribute to a functional system. | Uses all data maintenance actions.<br>All actions correctly used.<br>Most applications do not contribute to a functional system. | Uses some data maintenance actions.<br>All actions not correctly used.<br>Applications do not contribute to a functional system. | Totally inadequate or not done. | **4** | |
| **Documentation** | **4** | **3** | **2** | **1** | **0** | | |
| **Comments/Notes** (Explanation of program and code) | Code clearly annotated to fully explain all necessary parts.<br>Explanation shows excellent insight.<br>Extensive project notes present and of an excellent standard. Clearly explains working of the program | Code clearly annotated to explain all necessary parts.<br>Explanation shows good insight.<br>Project notes present and of a very good quality | Code annotated to explain most necessary parts.<br>Explanation shows some insight.<br>Project notes present and of a moderate standard | Code annotated to explain certain parts.<br>Explanation shows little insight<br>Inadequate project notes present | No comments or no project notes | **4** | |
| **Overall** | **4** | **3** | **2** | **1** | **0** | | |
| **Does the program meet the requirements?** | Exceeds requirements stated in Phase 1.<br>Comprehensive program.<br>All elements function as specified.<br>Shows insight in all aspects. | Meets the requirements stated in Phase 1.<br>Less comprehensive.<br>All elements function as specified.<br>Shows insight in most aspects. | Meets most of the requirements, but some don't function well<br>Only some program elements function as specified in Phase 1.<br>Shows insight in one or two aspects. | Only meets some requirements, and some don't function well.<br>Basic program.<br>Basic scope.<br>Very limited insight. | Does not meet the requirements.<br>Less than basic.<br>Limited scope. | **4** | |
| | | | | | **Total (implementation):** | **72** | |

## Assessment Summary

| Phase | Focus | Maximum Mark | Mark Obtained |
|---|---|---|---|
| Phase 1 | Analysis and Design | **60** | |
| Phase 2 | Coding and Implementation | **72** | |
| General | Final product and impression | **18** | |
| | **Total** | **150** | |
| **Final mark** | | | |

## 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 A – Declaration of Authenticity

| | | | |
|---|---|---|---|
| **Learner's name** | | **ID Number** | |
| **Grade** | 12 | **Year** | 2018 |
| **Subject** | Information Technology | | |
| Practical Assessment Task (PAT) | | **Teacher** | |

I hereby declare that the contents of this assessment task are my own original work (except for items listed below or 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.

**List of assistance received:**

| Nature of assistance | Person who provided assistance |
|---|---|
| | |

_____ ___ / ___ / 2018

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
- Identify the focus area to be investigated/researched
- Plan, research and carry out the project
- Write a report to a specified audience
- Develop a software solution according to requirements
- 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?

Before embarking on the PAT the learner needs to be taught the following:

- Programming code, principles and techniques
- Software engineering principles and techniques including the use of software design
- Project management skills including time, resource and task management
- The format and structure of accepted forms of research report to include abstract, introduction, discussion with all sources cited, conclusion, references

## Malpractice

Learners may NOT:

- Get help/guidance from others without acknowledging this help (complete **Annexure A** for EACH phase)
- Allow other learners to do the programming code of their project
- Submit work which is not their own
- Lend their PAT work to other learners
- Allow other learners to access or use their own work (this does not mean that they may not lend or borrow books to or from another learner, but they may not plagiarise other learners' research or work)
- Include work directly copied from books, the internet or other sources without acknowledgement and recognition (this may not exceed 20% of the work you submit)

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 A**) 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.

# 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
- Facilitate pre-reading to gain background information about the topic/scenario
- Give regular feedback to learners, e.g. on their investigation and analysis.
- 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 general evaluation based on continuous observation and feedback to provide a final impression regarding independent work, planning, insight and problem-solving.
- Make the assessment of the work of the learners by following any standardising and internal moderation procedures required
- Conduct a debriefing session to determine the learner’s understanding and insight of his/her project

The teacher will assess the potential project (analysis and design) 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?
- Is the focus area 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 in 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.

**Option 2:**

- The teacher could dedicate a continuous period of time to the PAT, for example the week(s) at the end of the second term and the start of the third term.

It is suggested that the teacher records the learners' topics/focus when they start with Phase 1 to avoid 'instant projects' that might possibly not be the learner's own work.

# 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 the documentation submitted for each phase
- The completed learner assessment tool (for each phase)

# 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 until 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 incomplete PAT components.

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 of the tasks.

**NOTE:**

In the instance of IT, such a situation is not very likely (unless a learner’s illness spans a long period) as the PAT is not a once-off sitting/assessment.

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

In the event of a learner failing to comply with the Practical Assessment Task requirements of a particular subject, and where valid reasons are provided, the evidence of such valid reasons must be included with the evidence of learner performance.