basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

INFORMATION TECHNOLOGY

GUIDELINES FOR PRACTICAL ASSESSMENT TASKS

2014

These guidelines consist of 35 pages.

What is the PAT?

The PAT is a software development project in which you will have an opportunity to demonstrate your software development and programming skills.

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.

After completion of the project you will have to submit the following as proof:

- a report where you (phase 1)
  - provide a brief description of the purpose and scope of your project
  - discuss the research/investigation done regarding the project
  - provide analysis of a possible solution
- a document outlining the system design (phase 2)
- a working Delphi/Java program, fully documented, that implements the planned solution (phase 3)

NOTE: You will be required to demonstrate and discuss your program during a debriefing session.

Mark allocation

The PAT counts 25% of your final mark for IT, therefore it is crucial that you strive to produce work of a high standard.

Phase	Development Phase	Maximum Mark
Phase 1	Analysis	30
Phase 2	Design	50
Phase 3	Coding and testing (50) Complexity level (30)	80
General	**Final product and impression**	20
	Total:	**180**

NOTE:

- The PAT mark is a compulsory component of the final certification mark for all candidates registered for Information Technology.
- **Submittion date:** Your completed PAT needs to be presented and submitted not later than three weeks before the start of the Grade 12 final exam.
- Your PAT will be moderated by subject experts and quality assured by Umalusi.

The Problem

TOPIC – Software system for a non-profit/charity organisation

As part of an effort to contribute towards the good work non-profit and charity organisations such as CANSA (support cancer patients), The Red Cross, children's homes, SPCA, organisations caring for the deaf, blind, etc. do, you have to develop an information system to assist anyone of these types of organisations in your area.

Choose a specific non-profit or charity organisation and do research on their information system requirements.

The system you develop could support your organisation of choice in activities such as:

- Keeping track of their staff, activities and members
- Capturing information on fundraising events
- Keeping track of their different branches, donations and supporters
- Capturing information on awareness campaigns
- Organising the order, distribution and income generated from marketing material.

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

Specific requirements:

- The project must include the major development tools, i.e. a database and programming language constructs, appropriately integrated.
- Include all the aspects mentioned in the PAT requirements (see page 5).and phase 3 marking guidelines (Annexure A).

General programming aspects that will 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 program with logically related parts. Projects which consist of two or more unrelated programs will only obtain marks for one of the programs since only one of the programs will be regarded as the actual project.

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:

- Determine the requirements of the client/user – do some research if needed and/or conduct interviews.
- Research possible solutions by looking at existing solutions.
- Report on your research/investigation – providing clear references/evidence.
- 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 or requirements.

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 with detailed specifics.

The following must be done:

- Clarify the requirements, indicating specifically how each requirement will be met in your program/solution.
- Using the necessary tools, design the Database, GUI and program – providing exact specifications for each.
- Clearly indicate the logical program flow and navigation between screens.

NOTE:

The following tools should/could be used in the design process:

- IPO diagrams, TOE charts, ERD diagrams.
- Flow charts, UML diagrams (class diagrams, activity diagrams and use case diagrams).
- System and database design tools available on the Internet.

PHASE 3 – CODING AND TESTING

The purpose of phase 3 is to implement the design by writing the code and testing the program:

- 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 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, e.g. to populate data structures and provide reports.

Database

The database component must:

- use more than one linked table (relational tables)
- involve sufficient data volumes and use a variety of field types
- be accessed and manupilated using your own programming code (combined with SQL)

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 will not replace variety, effectiveness and quality.

Text file

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

The data from the text file could be used to:

- do calculations and manipulations in combination with data form the database
- update excisting data
- insert records into the database.

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

GUI

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

- three forms/screens and
- two components (different types) that are created dynamically.

Data structures (programming)

You are expected to design one or more simple classes which contain attributes, constructors and methods with parameters and return types.

Also, incorporate your own methods, e.g. to validate data or to transform/manipulate data stored in the database.

Further requirements:

Apply good programming practices such as the following:

- Use descriptive names for variables, data structures, fields, components, etc.
- Construct well-structured and readable programming code.
- Apply modular programming (design methods to perform tasks and group methods into classes).
- Write comments/annotations to explain pieces of code. Give more detailed comments on the way in which input variables/data structures and output variables/components are used and interpreted.

Write project notes

- These notes must describe on how to interact with the program. The notes must also describe any known bugs or problems.
- Project notes can be written as a part of the help function of the program.

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 (e.g. magazines, newspapers, brochures, textbooks) or other electronic material (e.g. e-books, e-articles).
- Access to facilities to convert hard copies to electronic documents, e.g. scanner, digital camera
- Storage media to save and backup your work electronically, e.g. 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 acknowledgement..

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 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 you have a good idea of what is expected of the software you need to develop.

Select a company or application to apply the specifications described in this document. Interview or carefully analyse the company's 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 (±150 words) in your own words to describe the problem/task and how the project will solve the problem in general terms. In other words the description should be written to convince the manager of the company to hire you to develop the required software. The manager should realise that

- you understand the problem and
- your solution will solve the problem

It has to also describe how your program satisfies the PAT requirements/specification.

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.

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 of the research is a report (±1200 words). The report should be written in clear unambiguous language and can include explanatory diagrams.

DO THE ANALYSIS

The aim is to derive and establish the user, data and processing requirements of the system, including a consideration of the human aspects (e.g. suitable user interface) and physical environment (e.g. hardware requirements). It should specify ***WHAT*** is needed (not HOW).

Specify and document the data and processing requirements for the solution to the problem:

- Identify the prospective user(s)
- Identify user needs and acceptable limitations
- Data source(s) and destination(s)
- Justification of proposed solution

HAND IN

Hand in a document that contains the following:

- The project description (±150 words)
- A report which contains the following
    - outlining the research/investigation (±1200 words)
    - the analysis of user requirements
    - the acceptance tests

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 pseudocode/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 inter-relationships 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 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 provide reports.

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

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 from 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
- User requirements, usability
- Dialogues – must be relevant, simple and clear
- Icon usage and presentation – well selected and relevant, well placed and purpose clear.
- Colour – use of 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 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, ETC.)

Use appropriate design tools to design the overall solution considering all constituent parts and the inter-relationships between the various parts of the program/system.

- Description of modular structure of program/system
- Definition of data requirements
    - Structures
    - File organization and processing (e.g. 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.

DEVISE A TEST STRATEGY

Select and document suitable test data with expected results for normal (typical) data, erroneous data and boundary (extreme) data.

HAND IN

Hand in a document that provides the following:

- A planned database design
- GUI design
- Class design
- Overall solution design
- Test strategy

Instructions for Phase 3

This is where you implement your design by making use of appropriate software tools (programming language, database software, IDE, etc.) 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 making use of 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 structure.

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.

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 authenticity (See **Annexure E**)

DEBRIEFING

Demonstrate the program for evaluation and debriefing.

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 of the randomly selected code. 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.

GOOD LUCK!

Annexure A: Assessment Tool

Phase 1:	**Name of learner:**						
Scenario	**4**	**3**	**2**	**1**	**0**		
Scenario (Short description ±150 words)	Task is clearly stated (purpose and audience) Thorough understanding of what the problem/task involves. Cover all aspects.	Task is clearly stated (purpose and audience) Understanding the problem/task involves. Cover 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**	
Investigation/ Research		**3**	**2**	**1**	**0**		
Report on key areas the program will address		Extensive research done Clearly explains all key areas Well explained summary of what the program should do. Show thorough understanding	Acceptable amount of research done Discuss most key areas Acceptable summary Show 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	**3**	
Conclusion		Excellent, clear direction for the project, e.g. clearly defines the scope of the program Clear overview of a very appropriate possible solution. Thorough insight and understanding	Adequate direction – not always drawn from investigation, scope and purpose not clear in some respects. Adequate insight	Vague, direction unclear - little relevance to investigation or scope and possible solution not appropriate Minimal insight	Not drawn from the investigation or not relevant topic No direction for the project or no conclusion	**3**	
Report Structure		Well-structured report. Provides, e.g. relevant screen shots, printouts, etc. Includes all aspects outlined in investigation section	Acceptable structure Few screen shots, printout, etc. not always relevant Includes almost all aspects outlined in investigation section	Weak structure No screen shots, printout, etc. not always relevant Includes only a few aspects outlined in investigation section	No report or not relevant to the aspects outlined in investigation section	**3**	
References			All references (at least 3) using Harvard/APA style	Some (only 2) references included or incorrect style	No references included	**2**	

User requirements	**4**	**3**	**2**	**1**	**0**		
Role, activity, requirements and limitations	Role, activity, requirements and limitations of at least 3 different types of users of the system discussed Well documented, neat and to the point.	Some short comings in discussion of role, activity, requirements and limitations of at least 3 different types of users of the system discussed. Documented well but can improve slightly	Short comings in discussion of role, activity, requirements and limitations of users e.g. sections left out. Only 2 different types of users of the system discussed. Not well documented but still acceptable.	Serious short comings in discussion of role, activity, requirements and limitations of users e.g. 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	**4**	
Acceptance Tests	Tests for all requirements are clearly defined	Tests for almost all requirements are defined.	Tests for most requirements are defined. Not always well defined	Tests for a limited number of requirements are defined. Poorly defined.	No tests defined or incorrect	**4**	
Use Case diagram		Clearly describes what the system does to accommodate needs of all users Clearly shows requirements and test cases in all instances	Clearly describes what the system does for all users but minor shortcomings Clearly shows requirements and test cases but minor shortcomings	Describes what the system does in some instances Shows requirements and test use cases in some instances	No use case diagrams or incorrect	**3**	
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:	**Learner Name:**					
Database design	**3**	**2**	**1**	**0**		
Choice of fields	Well-chosen fields. All fields contribute to solution No redundant fields	Majority of fields contribute to 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	Appropriate number of tables for solution Appropriate relationships between tables Keys correctly specified in all tables Good collection of sample data	At least two tables No poor or incorrect use of relationship between tables Keys incorrectly used in some instances Limited collection of sample data	Only one table/only the fields listed – no relationships Correct primary key Poor collection of sample data	Only one table/only the fields listed – no relationships No/incorrect primary key Or No database	**3**	
Role of DB (no SQL) Describes how DB will be manipulated, e.g. within a dataset, access fields and records, navigate records etc. Role of manipulation in program described/motivated	Clearly describes how DB will be manipulated in 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 or irrelevant	**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	Satisfactory. GUI design considder/meets at least 4 requirements	Limited consideration of requirements – at least 3 requirements	Poor GUI design considering less than 3 requirements considered	**3**	

Components	Most appropriate 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 few cases, layout not satisfactory Mostly not substantiated	Inappropriate components used in most cases or choices not substantiated	**3**	
Data structures used (excl. Database)	**3**	**2**	**1**	**0**		
Choice of data structures, e.g. arrays, text files, variables, etc. (***How*** data will be stored)	Excellent variety of most appropriate data structures All choices clearly contributes to solution and are clearly substantiated	Variety of appropriate data structures. One or two data structures could have been replaced with more appropriate ones Most choices substantiated	Variety of appropriate data structures. More than two data structures could have been replaced with more appropriate ones View choices substantiated	Not described or substantiated or no variety	**3**	
OOP	**3**	**2**	**1**	**0**		
Classes and objects	Extensive number of classes. Well defined including – attributes and methods. Potential volume of data is significant	An acceptable number of classes. Fairly well defined, minor short comings in definition of entities	Limited number of classes defined. Serious short comings in definition of classes	No classes defined or totally incorrect or irrelevant	**3**	
Class diagrams	Clearly describes classes, their attributes, operations (methods) and the relationship between the classes in all instances	Clearly describes classes, their attributes, operations (methods) and the relationship between the classes in most instances but minor shortcomings	Describes classes, their attributes, operations (methods) and the relationship between the classes in few instances – major shortcomings	No classes or totally incorrect	**3**	
IPO design	**3**	**2**	**1**	**0**		
Input (***How*** input will be obtained and managed)	Clearly describes all inputs in terms of how input has to be obtained, how sources of input will be used and the format of the input, e.g. type, size	Most inputs described. Minor short comings in descriptions	Only some inputs described. Limited description	No input requirements described or incorrect	**3**	
Processing (***How*** processing will be managed)	Clearly describes all processing/ manipulation in terms of how data has to be processed/ manipulated (algorithms, formulas, etc.)	Most processing/ manipulation described. Minor short comings.	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 in term s of how it will be displayed, how sources of output will be used, as well as the format, type and size of the output	Most outputs clearly described. Minor short comings.	Only some outputs described. Mostly limited, incomplete or incorrect descriptions	Output requirements not described. Incorrect or irrelevant.	**3**	
Error catching	**3**	**2**	**1**	**0**		
Validation (***How*** integrity of input, processing and output will be managed)	Clearly describes appropriate, meaningful, effective • techniques to ensure IPO integrity • validation/error catching for all 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**	
Testing		**2**	**1**	**0**		
Test data and expected results		Complete set of most appropriate test data to be used that includes the expected results	Minimal test data or mostly not appropriate or minimal expected results included	No test data or test data totally inappropriate or no expected results	**2**	
Overall planning	**3**	**2**	**1**	**0**		
Planning	Produced a thorough high-level overview plan that clearly shows how all aspects of the problem will be solved Plan includes well-annotated pseudo-code/diagrams/charts (or suitable alternative) showing all of the main blocks within the proposed solution	Produced an acceptable high-level overview plan that contains a reasonable attempt to show how most aspects of the problem will be solved Plan includes annotated pseudo-code/ diagrams/charts (or suitable alternative), showing most of the main blocks within the proposed solution	Produced a limited high-level overview plan, minimal attempt , many shortcomings Plan includes pseudo-code/ diagrams/charts (or suitable alternative), but not always well-annotated, or showing only some of the main blocks within the proposed solution	No plan or plan very vague and confusing No annotations or main blocks not showed	**3**	

Use of software engineering tools	**3**	**2**	**1**	**0**		
Software Planning Tools	All tools, TOE chart, Class and Use Case diagrams, ERD and IPO charts appropriately used	Most of the tools (at least 3) used appropriately	Some tools used appropriately (at least 2)	No tools used or no tools used appropriately	**3**	
Presentation	**3**	**2**	**1**	**0**		
Time Management – Phase 2	Met deadline and all work done	Did not meet deadline, one day late, or not all work done	Did not meet deadline, two days late, or not all work done	Did not meet deadline, more than two days late, or most of the work done	**3**	
				Total	**50**	

Comment/feedback:

_____

_____

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

Implementation

Phase 3:	**Learner name:**						
Program aspects	**4**	**3**	**2**	**1**	**0**		
Algorithms What does it do? How well does it do it?	All solution algorithms used in solving the problem are most appropriate and effective, e.g. nested if else-statement used effectively instead of multiple if-statements and work correctly. Enhance the project	Appropriate solution algorithms used and 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, etc.)	Used appropriate and most effective control structures to solve the problem in all instances	Used appropriate and most effective control structures to solve the problem in all instances, but 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, excl. DB)	Most appropriate and effective data structures, correctly used (e.g. arrays, text files, etc.) to solve problem in all instances	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 all cases	Mostly good, proficient interaction between modules with parameter passing	Some, interaction between modules with parameter passing	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 minor shortcomings.	Appropriately and effectively used in most instances. Limited variety (e.g. only two types)	Appropriately and effectively used in some instances Very limited input (no variety/only one type)	Totally inappropriate or ineffective	**4**	

Output (coded)	In all cases: Most appropriate display, well formatted/readable/structured/ understandable, e.g. headings and repeated on following page/screen where applicable. No logical errors. All the results of processing are correct.	In all cases: Most appropriate display, well formatted/readable/structured/ understandable, but with minor shortcomings in one instance One minor logical error One result problematic	In most cases Appropriate display Some logical errors Some of the results are not correct.	In some cases Appropriate display Many logical errors Difficult to read output Many results incorrect	Many logical errors. Almost all the results are incorrect/few of the required results are delivered	**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	**4**	
Database **Use and interaction**	**4**	**3**	**2**	**1**	**0**		
Embedding of database manipulation within HLL • Access fields as part of a record for read/write purposes and save current record using applicable methods • Navigate records of dataset, e.g. move the current record pointer using a method or changing index value • Modify individual fields and records in dataset using applicable methods and apply all changes, e.g. save/edit/delete the current record or move to the next/previous record	Excellent, smooth interaction with HLL. Well-chosen manipulation through HLL code constructs that all contributes to the solution. Used at least 4 of the listed manipulations	Good, smooth interaction with HLL. Good manipulation through HLL code constructs that all contributes to the solution. Used at least 3 of the listed manipulations	Satisfactory interaction with HLL. Satisfactory manipulation through HLL code constructs that all contributes to the solution. Used at least 2 of the listed manipulations	Little interaction with HLL. Little manipulation through HLL code constructs that contributes to the solution. Used at least 1 of the listed manipulations	No interaction or manipulation or no database	**4**	

• Manipulate a dataset ohject and records and apply all changes, e.g. filter/Search/Sort a dataset and refresh the dataset • Use event/listener that triggers after record pointer has moved -for validation of fields -when a new record is to be created, inserted, modified -to perform calculations on non-table fields, e.g. calculated field							
GUI	**4**	**3**	**2**	**1**	**0**		
Ease of use/HCI principles	Excellent – all of the following: • Very intuitive • Excellent communication (screen tips, feedback, help etc.) • Most appropriate components • Readable/understandable output • Excellent use of effects/ colour/icons/shortcuts, etc.	Good – one aspect omitted or not good	Satisfactory – two aspects omitted or not good	Limited – more than two aspects omitted or not good	Poor GUI design. Little/no thought given to HCI principles	**4**	
Dynamic components	At least 2 dynamically instantiated components, both meaningfully and correctly used	At least 1 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**	

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	2	1	0		
Time Management – Phase 3	Met deadline, all work done	Did not meet deadline, 1 day late, all work done	Did not meet deadline, more than one day late, or not all work done	**2**	
			Total (implementation):	**50**	

The following table determines the complexity level of the program to discriminate between different levels of programs. Tick all features that are present in the program. **Complexity level** (Only one tick per line is allowed. Greyed blocks cannot be ticked. At the bottom, multiply the number of ticks in each column by the number at the top of the column)

Phase 3:	**Learner name:**					
Complexity level	**Complex (3)**	✓	**Adequate (2)**	✓	**Limited (1)**	✓
Algorithms	**Non-trivial algorithms**		**More advanced Grade 11 type**		**Trivial algorithms**	
User defined	Includes multi-nested loop/conditional constructs		Only double nested loops/conditionals		Only single loops/conditionals	
Database Exclude CRUD operations	High data volumes, many constraints, complex own code Programming language native operations to link, store/retrieve and manipulate data Non-trivial input/output, processing, e.g. data extracted must be processed further (transformed) to obtain desired result, assigning query parameter values at runtime; modular approach, e.g. use of procedures/functions/separate units Dynamically parameterised queries (parameters set at runtime, embedded in programming language statements written by learner		Standard, significant data volume Limited own code, Standard operations only Standard input/output, standard processing		Limited, involve only data storage/retrieval and possibly trivial processing by the package virtually no own code – limited use of programming language Limited input/output, no significant processing	
Standard	Complex, non-trivial or outside Grade 12 curriculum, e.g. recursive		Standard, several operations, within Grade 12 curriculum e.g. sorting, searching, etc.		Simple, one operation, covered in Grade 12 e.g. finding smallest of two values, even or odd, etc.	
Standard/user	Non-trivial drawings/animations/graphs/maps/timing		Standard drawings/ animation/graphs/maps/timing		None	

Utilizing sophisticated features of programming language						
Scope of variables	Use local and global variables appropriately and effectively – enhances program		Use local and global variables; but not always appropriately		Limited number of variables Local only	
Programming techniques	Outside curriculum, e.g. play video clips, threads, networking, mobile app., time based simulation – may be borrowed code (Must work correctly, be appropriate, add value to the solution)		None/not working correctly/not adding value		None	
Complexity of non-computing						
Manipulating math processes:	Involving mathematics beyond Grade 12-level, e.g. 3-D vector manipulation Non-trivial statistics/calculations provided		Grade 12 mathematics level Standard statistics provided, e.g. number above average, top 10%, etc.		Simple mathematical calculations, e.g. addition, subtraction, multiplication and division Statistics – only simple aggregates such as sum, average, minimum, maximum	
Manipulating string processes:	Combine multiple string methods to do complex manipulation, e.g. generate code/key extracting parts from several DB field/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)	
Modular aspects (Reuse of code)						
Re-use of code	Re-use of code – Good use of functions/procedures/methods/ objects/parameter passing		Good use of functions/procedures /methods/ objects/parameter passing		Used – inappropriate or no parameter passing or not working correctly	
Technical solution						
Exception /error catching/validation	Effective use of appropriate and efficient programming features and techniques' to produce a robust solution		Good use of programming features and techniques' to produce an adequate solution		Limited use of programming features and techniques to produce a simple solution	
	Total (Number of tics multiplied by 3)		**Total (Number of tics multiplied by 2)**		**Total (Number of tics multiplied by 1)**	
Column 1 + Column 2 + Column 3 (Maximum: 30) (Complexity):					**Total**	
Final phase 3 mark: Implementation + Complexity						

General: Final product and impression			**Learner name:**			
Aspect	**4**	**3**	**2**	**1**	**0**	**Mark**
Flow of development	Each phase of development logical flowed from previous phase. Did not deviate from original scope. The initial goal was reached and all the set requirements stated in Phase 1 was 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.	Almost none of original requirements were met.	
Professional project	Useful and can be implemented as a real life application. Compares well with proprietary software. Contains professional features such as 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 satisfactory with proprietary software. Contains professional features for almost all aspects such as 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 extend with proprietary software. Contains professional features for satisfactory number of aspects such as help-functions, well-designed GUI layout, no unexpected errors, user-friendly feedback on some but not all 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 such as 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. Contains limited professional features such as help-functions, well-designed GUI layout, no unexpected errors. User-friendly feedback on limited aspects and features.	
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	
Attitude and commitment	All phases were handed in on time and were well designed Worked 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	
					Maximum: 20	

Adjustment %

Debriefing	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	50	
Phase 3	Coding and Implementation	50	
Phase 3	Complexity	30	
General	Final product and impression	20	
Total		180	
Adjustment %			
Final mark (Total x Adjustment %)			

Authentication Declaration

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:

_____

_____

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

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 looks 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 report and their page numbers
INTRODUCTION <heading 1>	terms of reference, the scenario and outline of report's structure
BODY **Headings** <heading 1> ***Sub-headings*** <heading 2>	Headings and sub-headings which reflect the contents of each section. Includes information on important ideas about the topic, and Discussion of programs that is relevant to the scenario.
CONCLUSION <heading 1>	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 <heading 1>	list of reference material consulted during research for report use the Simplified Harvard style/APA style
APPENDIX <heading 1>	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):
_____ Signature of Learner	___/___/2014 Date

Annexure D

Declaration of authenticity

Learner name		**ID Number**	
Grade	12	**Year**	2014
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.

_____ ___/___/2014

SIGNATURE OF STUDENT DATE

Guidelines for Teachers

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, 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 atleast 3 different users of the planned system.
- bring together information to suit 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 abstract, introduction, discussion with all sources cited, conclusion, references

Malpractice

Learners must not:

- get help/guidance from others without acknowledgment (complete **Annexure B for each phase**)
- allowed 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 authentication of the PAT

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

After completing the PAT, learners to sign the declaration of authenticity (**Annexure C**) 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.

The teacher will manage the project and supervise the learners:

- conduct an initial planning review to discuss the topic/scenario, requirements, objectives and development of the project
- agree the focus question (learners should record the guidance given as part of the phase 1 document – Annexure B – e.g. where appropriate, record their own initial question with clear evidence of the guidance and the final question)
- give regular feedback to learners, e.g. to formulate a focus question that is suitable and manageable
- assess the work of the learners at the end of each phase using the standardised assessment tool and record feedback given
- endorse each learner's assessment by signing the assessment tools for each phase including a final declaration that the evidence submitted for assessment is the unaided work of the learner
- confirm their evaluation based on continuous observation and feedback as well as a debriefing 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, i.e. plan, research, analyse, evaluate and explain, rather than simply describe and narrate?
- Are the focus and proposed action clear and focused on an issue which can be managed within the timescale, 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 to confirm on the assessment tool that the work assessed is solely that of the learner concerned and was conducted under supervised/controlled conditions
- Teacher to 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 IT CAPS (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, e.g. the last week(s) of each term, also completing one phase per term.

Assessment Evidence

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 1600 words (content only, excluding 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)

Debriefing

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 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 marks 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) and 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.

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 reason in this context includes 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.