1)

a)

i) B) ✓

ii) B) ✓

iii] B) 2

iv] Requirement analysis 2

v] Decision 2.

b]

	Verification	Validation.
i]	It is a static process	It is a dynamic process of checking.
ii]	Code analysis and execution is not required	It involves the execution of code.
iii]	It is a human centric approach.	It is a computer based approach.
iv]	Target - requirement analysis, logic design, database design, code design.	Target - Final product is validate - module or unit
v]	It consist of methods like Desk checking, code walkthrough, code inspection.	It consist of methods like white box testing, black box testing and gray box testing

vi]	Verification is the first step to be taken place.	Validation is done after the verification process.
vii]	It answers the question- Are we building the product right?	It answers the question- Are we building the right product?
viii]	It finds out the errors that are not done by validation.	The errors missed by verification are identified by validation.
ix]	Example: Checking the position and spelling of content in a button is verification	Example: If the button navigates to the correct place after clicking is validation

a) Validation as we are more concerned with building the right product by checking fund transfer, generating receipts etc.

b] Verification is done as we check whether submit button appears after filling out a contact form.

ii] Fundamental Principles of Software Testing:

The fundamental Principles of Software Testing are discussed as follows:

i] Exhaustive testing is not always possible:

* It is not possible to test each and every line of the software that is written.
* Only the functionalities and the module workflows can be tested effectively
* Ex: In a railway reservation system, each time of testing consumes a large amount of time and thus majority of the functions are tested.

ii] Defect Clustering:

* A defect can lead to another defect causing a large number of defects.
* Pareto Principle: In most of the situations only 20% of the code will cause the 80% of the issue and thus that particular code can be resolved
* Example: In amazon website, if the payment service has a defect and then the delivery service also fails and thus payment service has to be corrected

iii] Pesticide Proximity:

* Testing the software with the same test cases will not be effective as it does not lead to find new errors present in the code.

* Example: Testing the local software with same test cases again and again will not introduce new error.

iv] Testing shows the presence of error:

* the testing just shows the presence of error and it does not conclude that there is no error in the software.

* Example:

Testing the facebook site just tells the error if present not that facebook is error free.

v] Early Testing:

* The testing of the software/user requirements can be done in the earlier stage as it reduces the cost of recovery from error while in the development stage.

* Example:

The requirements from the client must be tested and verified before developing

vi] Testing is context dependent:

* The softwares are independent from each other.

* They are not identical.

* Thus testing is context dependent and the same testing cannot be carried for all softwares

* Example: There are different testing methods and test cases used for a gaming platform from a website.

vii] Absence of error: Fallacy:

* If all the test cases has been passed, the software would run successfully in almost 99% of the cases but it is not sure that there is a absence of error in the software.

* Example: Even all the test cases for a project is passed, there may be some errors that are not tested

Thus these are the principles that contributes to effective testing practices.

c] V-Model:

Introduction:

* The V mode is a variant of waterfall model.
* The testcases are executed in all the stages and the testing is done in a parallel manner.

When to use V-model:

* When the requirements are frozen.
* Technology and solution are known.
* Verification and validation is carried through all the stages.

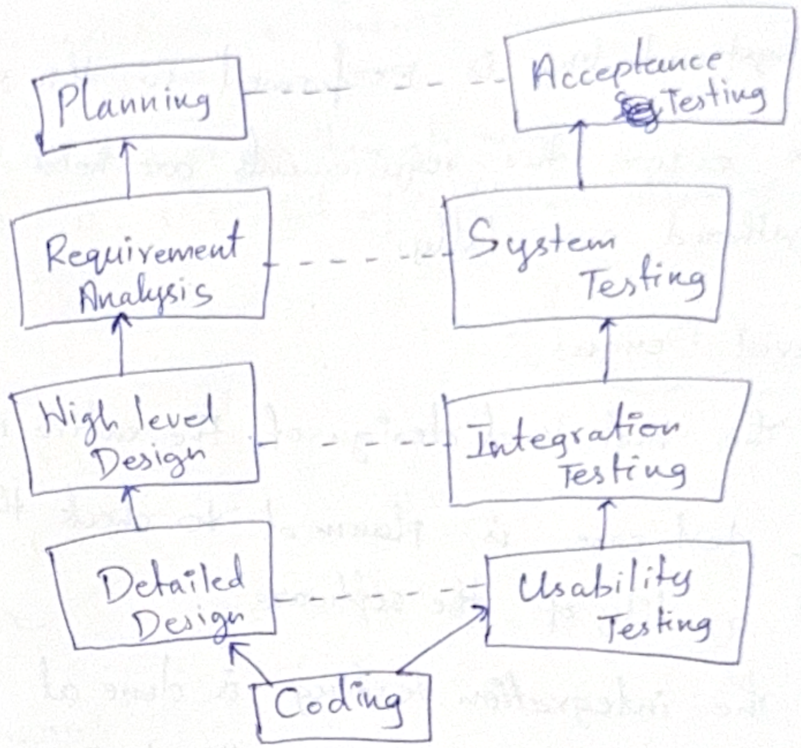
Strengths of V model:

* Easy to implement
* All requirements are tested effectively
* Parallel testing execution.

Weakness of V Model:

* No iteration process
* No change in the requirements of the software.

Diagram:



Planning:

* The software to be tested is analysed and the base plan for the testing is laid out in order to the test the software.
* It gets combined with the acceptance testing where the software scope is analysed based on the feasibility of its implementation.

Requirement Analysis:

* The various requirements of the product is analysed based on the user specification.
* All the requirements are specifically tested based on the analysis of the software.

* The system testing is performed in this stage so as to ensure the requirements are held and ~~maple~~ gathered successfully.

High level Design:

* The high level design of the entire implementation of the test case is planned. to check the overall quality of the software.

* The integration testing is done at this stage to analyse the errors in the high level design that is formulated.

Detailed Design:

* The detailed design of the software is prepared based on the requirements for the execution

* the usability testing ensures the design is subjected to the requirements of the user.

Maintanence:

* the regression testing is performed to make sure to correct the errors and it does not introduce any new errors in the software during

the maintanence phase.

This is the process through which each stage aligns with the V-model of software development.

2)

a] i] B) 6

ii] B) ✓

iii] D) ✓

iv] Hashmap ✓

v] Static ✓

c]

ii] Basis Path Testing:

* The Basis Path testing is used to test all the valid paths of the software.

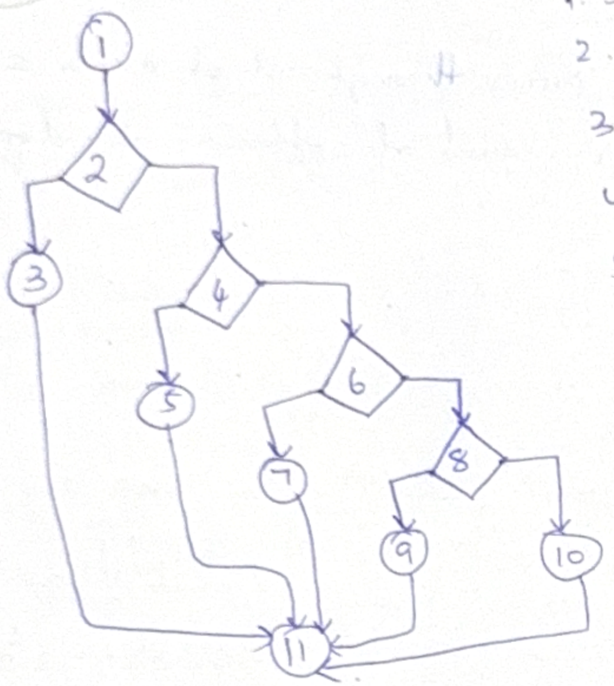
* This ~~and~~ involves the following process:

* Construction of control flow graph

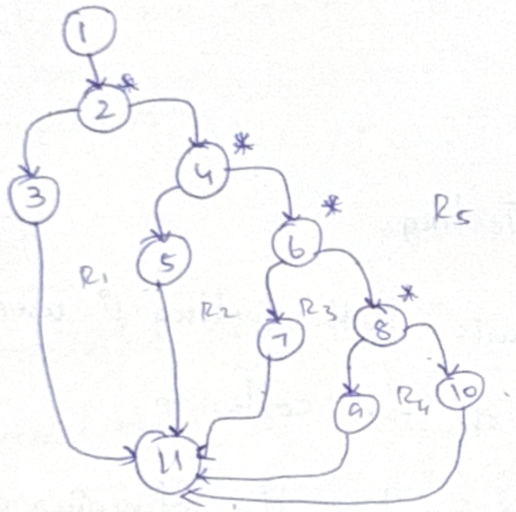
* Finding the independent paths of execution

* Deriving test case for each independent paths that are obtained.

Control Flow graph:




```
1. Start
2. if marks<0 or marks>
3.    return "Invalid"
4. If marks >=90:
5.    return 'A''
6. elif marks >=75:
7.    return "B"
8. elif marks >= 60:
9.    return "C"
10. else: return "D
11. End.
```



Cyclomatic Complexities:

V(G): Regions => 5.

V(G) => P+1 => 4+1 => 5.

V(G) = E-N+2 => 14-11+2

=> 3+2=5.

V(G)=5

Independent Paths:

1) 1 → 2 → 3 → 11

2) 1 → 2 → 4 → 5 → 11

3) 1 → 2 → 4 → 6 → 7 → 11

4) 1 → 2 → 4 → 6 → 8 → 9 → 11

5) 1 → 2 → 4 → 6 → 8 → 10 → 11

Test Cases:

Test Case - 1 Path - 1 :

Marks = -23.

Actual : Invalid Status: Pass

Expected : Invalid.

Test Case - 2 Path - 2 :

Marks = 94

Actual : A Expected: A Status : Pass

Test Case - 3 Path - 3:

Marks = 80

Actual : B Expected : B Status : Pass

Test Case - 4 Path - 4:

Marks = 62

Actual : C Expected : C Status : Pass.

Test Case - 5 Path - 5 :

Marks = 5 0

Actual : 5 Expected : 5 Status : Pass.

Thus the test cases are obtained for all the independent paths that are to be tested. and the Basis Path testing is applied successfully for the given Pseudo code.

b]

i] Regression Testing:

* the regression testing is mainly used in the maintanence phase of the software that is being developed.

* The Regression testing mainly takes care of the following two properties:

* The code changes made contains any error in it.

* the changes has introduced error in some other module as they are dependent on each other.

* When the software change is made on the existing product due to changes in the requirements the code changes has to be tested and also it

is necessary to ensure that the changes brought does not affect any other part of the software.

* The regression testing is used to check for these errors as the test cases are executed completely on the entire software after the code changes.

* Steps:

* Changes in code.

* Test cases for new code changes.

* Check the new functionality works properly.

* Check the existing functionality is not affected through the new functionality added.

* Through this way regression testing ensures quality and stability after changes are made.

ii] Usability Testing:

* The usability testing is done from the side of the users who are going to use the product.

* This testing ensures that the product is

designed as the user's requirements and are convinient for use by the user.

* This involves using

* Appropriate color combinations for colour blind people.

* Usage of voice system for the people who can't see.

* Making the website easy to navigate and providing accurate contents.

* Not changing the basic design for different versions of the product.

* Including suitable design for each component

* Avoid asking redundant information.

Thus, the usability testing improves the user experience of a software product through the above ways.