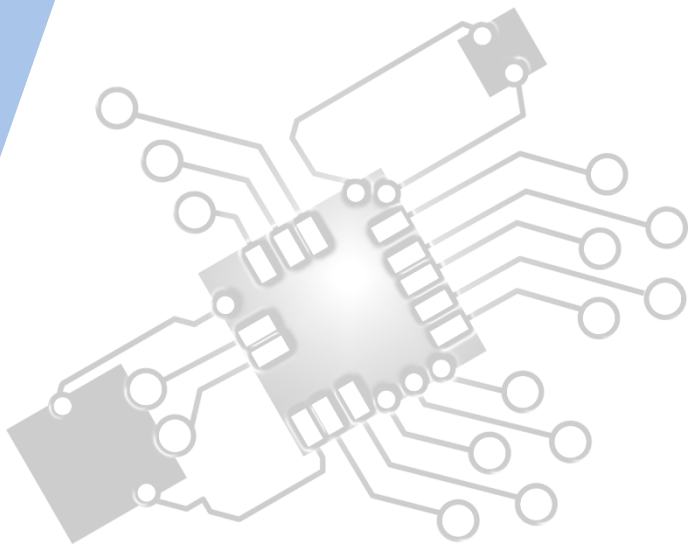
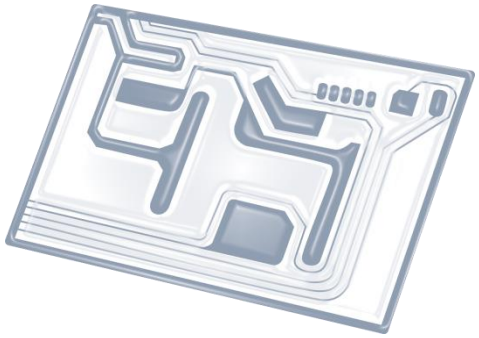
Planning & system installation

IB Computer Science



Content developed by
Dartford Grammar School
Computer Science Department

HL Topics 1-7, D1-4



1: System design

2: Computer Organisation

3: Networks

4: Computational thinking

5: Abstract data structures

6: Resource management

7: Control

D: OOP

HL & SL 1.1 Overview

Planning and system installation

1.1.1 Identify the context for which a new system is planned.

1.1.2 Describe the need for change management

1.1.3 Outline compatibility issues resulting from situations including legacy systems or business mergers.

1.1.4 Compare the implementation of systems using a client's hardware with hosting systems remotely

1.1.5 Evaluate alternative installation processes

1.1.6 Discuss problems that may arise as a part of data migration

1.1.7 Suggest various types of testing

User focus

1.1.8 Describe the importance of user documentation

1.1.9 Evaluate different methods of providing user documentation

1.1.10 Evaluate different methods of delivering user training

System backup

1.1.11 Identify a range of causes of data loss

1.1.12 Outline the consequences of data loss in a specified situation

1.1.13 Describe a range of methods that can be used to prevent data loss

Software deployment

1.1.14 Describe strategies for managing releases and updates

1: System design

2: Computer Organisation

3: Networks

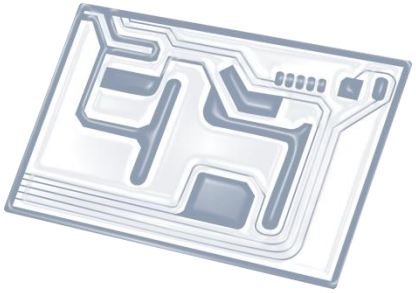
4: Computational thinking

5: Abstract data structures

6: Resource management

7: Control

D: OOP



Topic 1.1.7

Suggest various **types** of **testing**

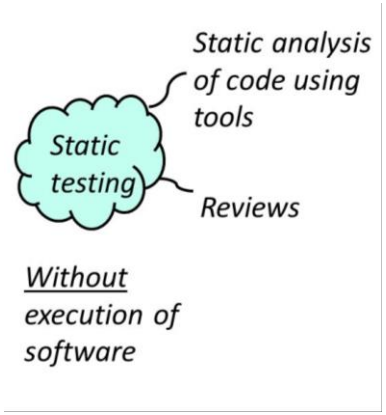
Testing is important!

- Testing is very important in developing a computerized system as it tries to ensure that the system **works as expected**.
- A system that does not work as expected (i.e. it has bugs) greatly reduces productivity and end user satisfaction.

Many things need to be tested

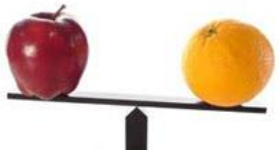
Static vs Dynamic testing

- Reviews, walkthroughs, or inspections are referred to as **static testing**, whereas actually executing programmed code with a given set of test cases is referred to as **dynamic testing**.
- Static testing is often implicit, like proofreading, such as when the IDEs check source code or syntax.
- Dynamic testing takes place when the program is run.
- Dynamic testing may begin before the program is 100% complete in order to test particular sections/modules of code.
- Static testing involves **verification**, whereas dynamic testing also involves **validation**. Together they help improve **software quality**.



With execution of software

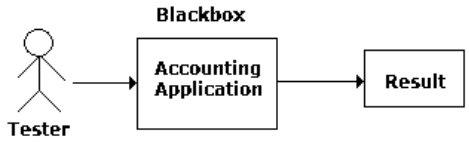
Alpha vs Beta testing

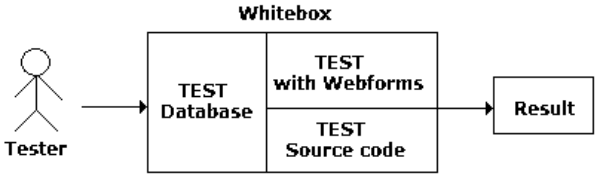


Alpha Testing	Beta Testing
Alpha testing performed by testers who are usually **internal employees** of the organization	Beta testing is performed by clients or end users who are **not employees of the organization**
Alpha testing performed at **developer's site**	Beta testing is performed at **client location** or end user of the product

Black-box vs White-box testing

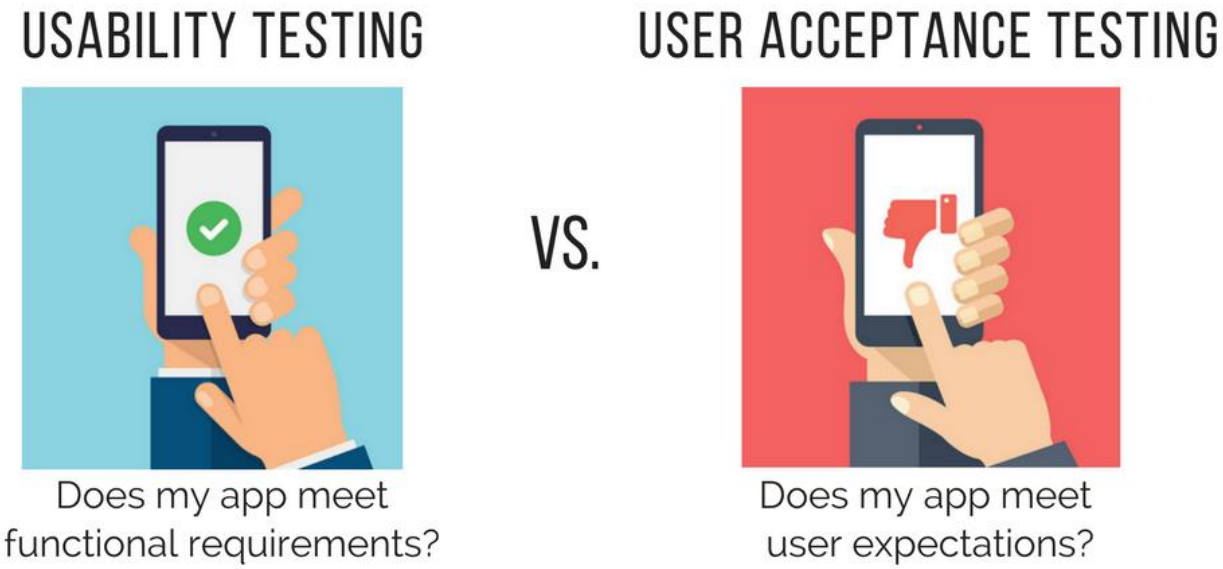
- **Black-box testing** (also known as functional testing) treats software under test as a black-box without knowing its internals. Tests are using software interfaces and trying to ensure that they work as expected.
- **White-box testing** (also known as structural testing) looks inside the software that is being tested and uses that knowledge as part of the testing process.





User acceptance testing

- Testing any new/updated system with its ultimate end users to see if it meets their expectation is very important.
- Happy users = more productive users = good for developers



Automated testing

- **Automated testing** is a method in software testing that makes use of special software tools to control the execution of tests and then compares actual test results with predicted or expected results.
- All of this is done automatically with little or no intervention from the test engineer.
- Automation is used to add additional testing that may be too difficult to perform manually of when the body of code that needs to tested is vast.