CAISL
Carlucci American International School of Lisbon

GRADE 12 ADVANCED COMPUTER SCIENCE LEVEL 2 - IB COMPUTER SCIENCE YEAR 2 FRAMEWORK

CONTENTS

OVERVIEW: 1
EXPECTATIONS: 1
ACTIVITIES/PROJECTS: 2
RESOURCES: 2
ASSESSMENTS: 2
STUDENTS ARE EXPECTED TO COME WITH THE BASIC KNOWLEDGE ON: 3
PERFORMANCE INDICATORS: 3
- SYSTEM FUNDAMENTALS 3
- COMPUTER ORGANIZATION 3
- NETWORKS 4
- COMPUTATIONAL THINKING, PROBLEM-SOLVING AND PROGRAMMING 4
- OBJECT-ORIENTED PROGRAMMING 4
- CONTROL 5

ADDITIONAL PERFORMANCE INDICATORS – HL 5
- ABSTRACT DATA STRUCTURES 5
- OBJECT-ORIENTED PROGRAMMING 5

OVERVIEW:

In Computer Science Year 2 students will develop a project that will be assessed internally. The project consists of an application that will solve a real-life problem. Students are encouraged to make a positive contribution to their environment creating a useful solution and learning in that process. Also, students will be prepared to answer questions about a complex context that changes yearly (e.g., Computer Science and Health Systems, Autonomous Vehicles and Artificial Intelligence).

EXPECTATIONS:

The student will be able to:

Develop computational solutions (develop simple programs)
Design, prototype and test a proposed solution
Understand and manipulate abstract concepts.
Develop logical and critical thinking as well as experimental, investigative and problem-solving skills
Know and understand relevant facts and concepts.
Know and understand appropriate methods and techniques.
Know and understand computer science terminology.

Know and understand methods of presenting information.
Apply and use relevant design methods and techniques.
Apply and use terminology to communicate effectively.
Apply and use appropriate communication methods to present information.
Construct, analyze, evaluate and formulate success criteria, solution specifications including task outlines, designs and test plans.
Construct, analyze, evaluate and formulate appropriate techniques within a specified solution.
Demonstrate the personal skills of cooperation and perseverance as well as appropriate technical skills for effective problem-solving in developing a specified product.

ACTIVITIES/PROJECTS:

Small applications solving specific problems.
Presentations to the class with results of a study/research about a topic.
Discussions about a relevant topic including ethical issues.

RESOURCES:

Moodle page
Computer Science Guide (First Examinations 2014)
Core Computer Science, Dimitriou K. & Hatzitaskos M., Express Publishing - 2015

ASSESSMENTS:

For students to receive a credit towards their High School Diploma, they must demonstrate proficiency on:

Summative assessments set by the class teacher which may take the form of:
Small descriptive and reflective essays about specific topics.
Small programs solving different problems.
Class presentations and discussions.
Written paper tests.
Develop and document a fully operational solution to a real problem.
Lab reports.
While the skills and activities of IB Computer Science are common to students at both SL and HL, students at HL are required to study additional topics in the core, a case study and also extension material of a more demanding nature in the option chosen. The distinction between SL and HL is therefore one of both breadth and depth.

The IB Computer Science HL course has some additional elements.

Assessments for **Standard Level**:

- External assessment Paper 1 – 45%
- External assessment Paper 2 – 25%
- Internal Assessment – 30%

Assessments for **Higher Level**:

- External assessment Paper 1 – 40%
- External assessment Paper 2 – 20%
- External assessment Paper 3 – 20%
- Internal Assessment – 20%

STUDENTS ARE EXPECTED TO COME WITH THE BASIC KNOWLEDGE ON:

Programming simple console applications with Java.
Developing simple graphical user interfaces with CSS templates.
Using an Integrated Development Environment to create applications.

PERFORMANCE INDICATORS:

SYSTEM FUNDAMENTALS

Identify the context for which a new system is planned. DOK 3
Describe the need for change management. DOK 3
Outline compatibility issues resulting from situations including legacy systems or business mergers. DOK 2
Know how to compare the implementation of systems using a client's hardware with hosting systems remotely for a given context. DOK 3
Know how to evaluate alternative installation processes. DOK 3
Prevent problems that may arise as a part of data migration on a given context. DOK 4
Describe different types of user documentation. DOK 2
Identify a range of causes of data loss. DOK 2
Describe a range of methods that can be used to prevent data loss. DOK 2
Describe the roles that a computer can take in a networked world. DOK 2
Identify the relevant stakeholders when planning a new system. DOK 2
Describe methods of obtaining requirements from stakeholders. DOK 2
Describe appropriate techniques for gathering the information needed to arrive at a workable solution. DOK 2
Construct suitable representations to illustrate system requirements. DOK 3
Describe the purpose of prototypes to demonstrate the proposed system to the client. DOK 3
Discuss the importance of iteration during the design process. DOK 3
Explain the possible consequences of failing to involve the end-user in the design process. DOK 4
Understand the social and ethical issues associated with the introduction of new IT systems. DOK 4
Define software usability and identify a range of usability problems with commonly used digital devices. DOK 2
Identify methods that can be used to improve the accessibility of systems. DOK 2

COMPUTER ORGANIZATION

Outline the architecture of the central processing unit (CPU) and the functions of the arithmetic logic unit (ALU) and the control unit (CU) and the registers within the CPU. DOK 2
Describe primary memory, distinguishing between random access memory (RAM) and read-only memory (ROM), and their use in primary memory. DOK 2
Explain the use of cache memory and give some practical examples. DOK 3
Explain the machine instruction cycle. DOK 3
Identify the need for persistent storage. DOK 2
Describe the main functions of an operating system. DOK 2
Outline the use of a range of application software. DOK 2
Identify common features of applications. DOK 2
Define the terms: bit, byte, binary, denary/decimal, hexadecimal. DOK 2
Outline the way in which data is represented in the computer. DOK 2
Define the Boolean operators: AND, OR, NOT, NAND, NOR and XOR. DOK 2
Construct truth tables using all studied Boolean operators. DOK 2

NETWORKS

Identify different types of networks. DOK 2
Outline the importance of standards in the construction of networks. DOK 2
Describe how communication over networks is broken down into different layers. DOK 2
Identify the technologies required to provide a VPN. DOK 2
Define the terms: protocol, data packet. DOK 2
Explain why protocols are necessary. DOK 2
Explain why the speed of data transmission across a network can vary. DOK 2
Explain why compression of data is often necessary when transmitting across a network. DOK 2
Outline the characteristics of different transmission media. DOK 2
Explain how data is transmitted by packet switching. DOK 2
Outline the advantages and disadvantages of wireless networks. DOK 2
Describe the hardware and software components of a wireless network. DOK 2
Describe the characteristics of wireless networks. DOK 2
Describe the different methods of network security. DOK 2
Evaluate the advantages and disadvantages of each method of network security. DOK 2

COMPUTATIONAL THINKING, PROBLEM-SOLVING AND PROGRAMMING

Define modules when needed. DOK 3
Evaluate whether the order in which activities are undertaken will result in the required outcome. DOK 2
Use sub-procedures, when needed, to solve problems. DOK 3
Identify when decision-making is required in a specified situation. DOK 2
Identify the condition associated with a given decision in a specified problem. DOK 2
Describe how concurrent processing can be used to solve a problem. DOK 3
Identify examples of abstraction. DOK 2
Outline the standard operations of collections. DOK 2
Analyze an algorithm presented as a flow chart. DOK 4
Construct pseudocode to represent an algorithm. DOK 4
Explain the essential features of a computer language. DOK 3
Outline the need for a translation process from a higher-level language to machine executable code. DOK 2
Construct algorithms using loops, branching. DOK 3
Describe the characteristics and applications of a collection. DOK 3

OBJECT-ORIENTED PROGRAMMING

Outline the general nature of an object. DOK 3
Distinguish between an object (definition, template or class) and instantiation. DOK 2
Construct unified modelling language (UML) diagrams to represent object designs. DOK 4
Describe the process of decomposition into several related objects. DOK 3
Describe the relationships between objects for a given problem. DOK 3
Outline the need to reduce dependencies between objects in a given problem. DOK 4
Construct related objects for a given problem. DOK 3
Explain the need for different data types to represent data items. DOK 3
Describe how data items can be passed to and from actions as parameters. DOK 4
Define the term encapsulation. DOK 2
Define the term inheritance. DOK 2
Define the term polymorphism. DOK 2
Describe the advantages of libraries of objects. DOK 2
Describe the disadvantages of OOP. DOK 3

Define the terms and use them in code: private, protected, public, extends, static. DOK 1
Describe the uses of the primitive data types and the reference class string. DOK 3
Construct code examples related to selection statements. DOK 2
Construct code examples related to repetition statements. DOK 2
Construct code examples related to static arrays. DOK 2
Discuss the features of modern programming languages that enable internationalization. DOK 2

CONTROL

Discuss a range of control systems. DOK 3
Outline the uses of microprocessors and sensor input in control systems. DOK 3
Evaluate different input devices for the collection of data in specified situations. DOK 3
Explain the relationship between a sensor, the processor and an output transducer. DOK 4
Describe the role of feedback in a control system. DOK 2
Discuss the social impacts and ethical considerations associated with the use of embedded systems. DOK 3
Compare a centrally controlled system with a distributed system. DOK 2
Outline the role of autonomous agents acting within a larger system. DOK 4

ADDITIONAL PERFORMANCE INDICATORS – HL

ABSTRACT DATA STRUCTURES

Describe the characteristics of a two-dimensional array. DOK 2
Construct algorithms using two-dimensional arrays. DOK 3
Describe the characteristics and applications of a stack/queue. DOK 3
Construct algorithms using the access methods of a stack/queue. DOK 3
Describe how linked lists operate logically. DOK 3
Sketch linked lists (single, double and circular). DOK 2
Describe how trees operate logically (both binary and non-binary). DOK 2
Define the terms: parent, left-child, right-child, subtree, root and leaf. DOK 2
State the result of *inorder*, *postorder* and preorder tree traversal. DOK 3
Sketch binary trees. DOK 2
Define the term dynamic data structure. DOK 2
Compare the use of static and dynamic data structures. DOK 3

OBJECT-ORIENTED PROGRAMMING

Define the term recursion. DOK 2
Describe the application of recursive algorithms. DOK 2
Construct algorithms that use recursion. DOK 2
Trace recursive algorithms. DOK 3
Define the term *object reference*. DOK 3
Construct algorithms that use reference mechanisms. DOK 2
Identify the features of the abstract data type (ADT) list. DOK 2
Construct algorithms using a static implementation of a list. DOK 3
Construct list algorithms using object references. DOK 4
Explain the advantages of using library collections. DOK 4
Outline the features of ADT's stack, queue and binary tree. DOK 2
Explain the importance of style and naming conventions in code. DOK 2