Qualification Accredited

A LEVEL

Programming Languages Guide

COMPUTER SCIENCE

H446

For first teaching in 2015

Programming Languages Guide

Version 2

www.ocr.org.uk/computerscience

Contents

Introduction

This guide compliments the Appendix in the A Level Computer Science specification which gives the scope of the specific languages learners could expect to encounter in the written examinations. Below is an overview of languages that might be used in the classroom for teaching and project work.

The choice of a programming language for A Level Computer Science is no simple matter. On one hand, console mode procedural languages appear better for learning theoretical concepts, while the Object-Oriented drag-n-drop GUI languages seem to simplify the coursework process. The ultimate choice will depend on the preferences of the teacher and their experience.

The hardest language to learn is the first one - all other languages will be easier to learn once this has been achieved. The OCR A Level specification will allow any appropriate language to be used as the most important concepts are common to all languages: sequencing, logic, and experience in troubleshooting.

The following are popular and appropriate languages.

Part 1 Languages Overview

PYTHON

Python is an incredibly powerful and versatile language that has quickly become the de-facto programming language in education due to its simple syntax, easy readability and reduced typing stemming from the fact that Python was designed for teaching.

It has a vast array of libraries that allow Python to work on almost all platforms and that extend Python's functionality to sophisticated scientific modelling, matrix and optimisation problems. It is used widely in universities for programming simulations in maths and the sciences. Python is also used by many large companies such as Google, and powers sites such as Youtube and Reddit.

Python is pre-installed on most Linux distributions and comes as standard on the popular Raspberry Pi. It also has an excellent implementation of lists (including some functions that won't look out of place in a spreadsheet, eg sum, min, max) and always a multitude of ways to perform a task. It has a variety of free resources, including free Apps learners can access on their mobile devices, reducing the need for purchasing textbooks. There is a wealth of free resources online specifically aimed at Key Stage 3 to Key Stage 5 which introduce Python in a manageable fashion which is sadly absent from other languages. Python lends itself to exploration in programming with a mantra of "unless it's wrong it's ok", where as long as it runs in the interpreter then its fine. Python is not strictly typed, like other languages, so the syntax is much more forgiving for novice programmers. This also means that data types do not need to be declared unless necessary which makes it much easier to use and learn. It also has a sophisticated OOP implementation that is optional, as Python is a "multi-paradigm" language - you can do procedural, functional or OOP depending on the level of learners/teachers which allows great flexibility when learning as skills can be re-visited later and refined to be more efficient. Everything in Python is treated as an object so teaching OOP is simpler.

There are several libraries such as PyQt and QtDesigner that allow a Visual Basic-style WYSIWYG GUI design for rapid creation of app interfaces. Python also has the ability to run on a mobile device and even in a web browser window and there are numerous apps, plugins and extensions that allow Python to be run on practically any device including a USB memory stick using PortablePython.

There are also many free IDEs available such as Pyscriptor and Visual Studio, which has an excellent Python plugin that makes coding Python really easy as it has code complete and many other useful features including Python for Kinect, where students can access the Kinect library of features to manipulate. Another popular IDE for Python is Eclipse, which also works with many other languages and is very well supported. All these IDEs incorporate and extend the Python interpreted and add extra functionality that makes it even easier to troubleshoot.

Despite the many advantages of Python compared to the other languages there are a few minor drawbacks, such as it is not the fastest language when compared to C++, especially when used in gaming. This is mainly due it being an interpreted language and not a compiler language like C++. Python is very different from many traditional languages to the point where it contradicts some theoretical questions! The major difference is that Python does not have a CASE statement or arrays as these are dealt with in lists. Python is not as accepted as Java and C++ in some industries. The default IDE is not very well specified, missing a lot of common IDE features like report generation, trace tables etc, but there are many open source IDEs that have these features, such as Pyscriptor or Eclipse which do all of this and are free.

C

In the ideal world, C would be the language to teach: it is accepted by almost all platforms and has plenty of free and paid resources available. It requires limited hardware, can be programmed in any text editor (then the code can be compiled with a small compiler) and is very fast. C has the ability to incorporate Assembler into its code allowing learners to find out a lot about their operating system and how it allocates RAM. It also allows learners to write software drivers, ie for a Linux environment which doesn't always have the same level of driver support as Windows or Mac. Data structures are handled very well through pointers and stacks. In fact, most C++ IDEs (and Delphi) are backward compatible and can compile C. Some syntax features are similar to a whole of family of languages dominant in design: C++, C#, Java, Javascript, Objective-C, etc.

C is popular as a robotics language, such as the popular Arduino platform. While less powerful than Raspberry Pi, Arduino is used in the industry and has a great number of resources and hardware modules available (http://www.arduino.cc/).

Unlike interpreted languages, C will not give feedback as it runs through the code line by line, instead, if errors are present, it will create an "unsuccessful compilation" report indicating the line numbers where the mistakes were found, which might not be ideal for those learning to program.

There are very few cases of C being used by beginner programmers. It is also a procedural language, which doesn't use Object Oriented programming. This means it won't cover the related syllabus topic in 1.2.4. There is also no native integration of SQL, however it is possible through ODBC drivers (www.easysoft.com/developer/languages/c/odbc_tutorial.html).

C#

C# is very similar to C++. The differences are mainly in their implementation of the Object Oriented paradigm. For example, strings are objects in C# but not in C++ (http://msdn.microsoft.com/en-us/library/yyaad03b(v=vs.90).aspx). C# is much more platform-dependent, as it is a part of Microsoft .NET and must be compatible with other .NET languages, such as VB.NET. It is Windows-specific but will run on Linux with the third-party Mono framework. This might degrade performance on a Raspberry Pi. In exchange for easier coding and assistance of templates, some of the freedom and creativity of C++ are gone (http://msdn.microsoft.com/en-us/magazine/cc301520.aspx), however, we do get drag and drop GUI design in the style of VB.

C++

C++ shares a lot of features with C. It is also aimed at desktop development rather than web and mobile, it is a compiled language with cryptic error messages and it also excels in giving learners knowledge of the operating systems. Drivers and utility software can also be effectively written in C++ using one of the many available industrial strength IDEs. Like C, C++ gives graduates some very employable skills and will provide continuity should they choose to study Computer Science at University. Unlike C, however, it supports Object-Oriented programming which is excellent for minimising repetitive code and has better GUI generation facilities. There are plenty of resources available for learning C++ but they are aimed at university level and will not provide an inclusive experience for learners at A level.

The downsides of C++ are similar to C, the learning curve is steep and it is very verbose (http://screenshots.en.sftcdn.net/en/scrn/50000/50804/c-code-export-10.jpg). Due to its OOP nature, the names of classes and methods can get quite complex, especially when using API calls (interfacing with operating systems' built-in functions). There is also incompatibility between different dialects of C++ while C has long been standardised.

JAVA

Java is from the same family of languages as C++, but its strength lies in its multiplatform capability. It is possible to develop Java programs for every platform: desktop or mobile, and therefore is much more popular than C++ or C#. There are a lot of resources and free IDEs available for Java. Alternatively, it can be developed in a Notepad window, as long as the Java SE Development Kit (JDK) is installed on a computer. Java is the most popular university teaching language, there are plenty of resources (but mostly university-oriented) and Java skills make graduates very employable.

The downsides of Java include the requirement to have Java plugins on user computers (except Android where a different Dalwik library is installed by default). The plugins also have to be regularly updated, as every few months there seems to be another bug or security breach found in Java. Another issue is the GUI design. Java supports "Swing" widgets for most platforms but doesn't have the flexibility of Microsoft Visual Studio unless an extension to a major IDE is used (for example Window builder in Eclipse). Java is a verbose language and it requires memorising quite a few commands and APIs which might deter some learners, and worsens the learning curve for the language.

Java runs on all platforms (except IOS and needs to be compiled in a special Darwik way on Android) and is standard across many industries as it shares a lot of conventions with other ECMA-type languages like C++, C+, Javascript and ActionScript. There is a great demand in the industry for Java programmers and it is a very valuable skill as so many applications are written in it that require maintaining and updating. Just like Python there is a variety of IDEs available, including learning-oriented platforms like BlueJay. There are a few drawbacks to learning Java first, such as that it forces learners to use OOP even if they are not ready to do so. It is very verbose and susceptible to typos by learners especially when you start out. There is no denying that it is slow and a resource hog. It also has many flaws and security bugs that are occasionally identified which forces the requirement for updates. This can break some programs which then need to be registered with the new Java. It also requires a sophisticated structure made of many files and subfolders which, along with the other areas of concern can mean that teachers without Computer Science backgrounds may be overwhelmed by the level of knowledge needed. Java apps might also be filtered as a security risk by some schools' security systems due to the insecure nature of Java.

VISUAL BASIC

Visual Basic is the most popular member of the Basic family. It exists in a variety of applications: the most professional and full-featured of VBs is VB.NET. It can develop desktop and web/mobile applications in a fully-featured free IDE (Visual Studio Express). Unlike other VBs, it supports Object-Oriented paradigm and can integrate with API calls, however, it only runs on Windows and the Internet Explorer/Silverlight platforms (Silverlight is similar to Flash but is promoted by Microsoft). However, the long-time users of VB have noted that .NET is harder to program, is a more verbose language than classic VB and requires the learner to remember more specific terms and class names. VB source code is compiled into .NET bytecode where it can interact with C++ and other languages from Visual Studio.NET. Visual Studio is a big package and can be slow on older (or virtualised) computers. Some schools will also have a problem allowing it to run from learner accounts (this applies to all other Visual Studio languages like C# and C++).

VB.NET is still a relatively simple and full featured programming language that can be comfortably delivered by teachers to most pupils and there are plenty of resources for it, as well as schools using it. It is obviously very visual, has GUI building facilities which are second to none and comes with many wizards, for example for integrating with Access or compatible databases. Some of the coding can be done by these wizards. On one hand, this help pupils get the projects done faster, on the other hand, they could create a big part of their data handling from wizards which will obscure the real working from them.

Visual Basic for Applications is an interesting and a simple language built on the foundations of classic pre-.NET VB. Running on top of Microsoft Office products, it is available on all but Starter editions by pressing Alt+F11. Interestingly, it also runs on a Macintosh version of MS Office, so it is possible to do programming cross-platform. Using VBA in Access allows for simple and integrated data-handling projects, while using it in Excel allows leveraging of all of Excel's graphics and functions. Using it in Powerpoint gives a great multimedia interface.

Even when Microsoft Office is not available, it is possible to program in a simple text editor. As Vbscript is Microsoft's answer to Javascript, it is possible to create a web application in VB

with GUI and proper programming using nothing but Notepad. However, the best features will only work with Internet Explorer, leaving out users with Chrome, Safari or Firefox. Another use of Vbscript is for system administration, together with Powershell (an obscure and hard-to-learn language that imposes Windows group policies, etc). Among other things, it can be used to bulk move folders and set up user accounts.

DELPHI/OBJECT-PASCAL

Delphi is an excellent and efficient language, based upon a one-time standard teaching language, Pascal. It has all the strengths of VB without its weaknesses but the industry adoption rate

has been poor. Its current distributor has rolled out versions for cross-platform capabilities.

Part 2 Project Type Conisderations

Learners' practical coursework projects are necessarily limited in scope by their lack of experience and time management. It is also to be assumed that not everybody taking A level Computer Science will go on to be a programmer or systems analyst.

Most programs are data oriented: stock control, scheduling, booking, testing. The necessary features include multiple levels of access, login verification, GUI/menus, file/database input/output, printed output. They follow **create, read, update and delete (CRUD)** (http://en.wikipedia.org/wiki/Create,_read,_update_and_delete).

Apart from CRUD, pupils sometimes opt to do games, simulations and apps.

The discussion of computer languages could be organised by their merits in these categories:

DATA HANDLING DESKTOP

These are relatively simple computation-wise, often not conceptually harder than traditional ICT Access Databases. This is compensated by a lot of validation checks and sophisticated data structure. There are lots of resources for it, but this type of software is going extinct in this connected world.

Languages: VB, Python, C#, C++, C, Objective-C/Cocoa and

Delphi (VB and Python seem to be more popular)

DATA HANDLING WEB-BASED

This is very employment-oriented with medium computational complexity (WYSIWIG (what you see is what you get) is often not possible). Web standards change all the time, and these projects often require knowledge of two languages (eg Javascript for client-side and PHP/ASP for the server side, plus CSS for the interface). Another consideration is the requirement for paid hosting services or getting the school's IT departments to run a server which they are usually not very keen on.

Languages: HTML5/Javascript/PHP/jquery, VB/ASP, Java, SQL

MOBILE APP

Usually easier to do computationally than desktop applications but require the knowledge of mobile API and server connections.

Languages: Objective-C (IOS), Java, Python (via SL4A) for Android, Java for Blackberry, HTML5 for Windows Phones, Django (based on Python)

DESKTOP GAME

Computationally complex projects that nevertheless, can really motivate learners.

Languages: C#, pygame, C++, Objective-C (MacOS), Java, ActionScript

VIDEO CONSOLE GAME

Language: X#

WEB GAME

Languages: HTML5, ActionScript, Visual Studio Dot Net via

Silverlight (http://en.wikipedia.org/wiki/Silverlight)

MATHS/SCIENTIFIC

Languages: Python via numpy, Java

3D SCRIPTING

Language: Blender with Python (http://en.wikipedia.org/wiki/Blender_(software))

SOCIAL MEDIA EXTENSION

Language: Ruby on Rails

ROBOTICS

Languages: C, C++, Python

COMBINATIONS

Languages: Cython – Python but with all the features and libraries of C, Iron Python – Python but with all the .NET libraries and features baked in

The small print

We'd like to know your view on the resources we produce. By clicking on the 'Like' or 'Dislike' button you can help us to ensure that our resources work for you. When the email template pops up please add additional comments if you wish and then just click 'Send'. Thank you.

If you do not currently offer this OCR qualification but would like to do so, please complete the Expression of Interest Form which can be found here: www.ocr.org.uk/expression-of-interest

ocr.org.uk/alevelreform
OCR customer contact centre

General qualifications
Telephone 01223 553998
Facsimile 01223 552627
Email general.qualifications@ocr.org.uk

A DIVISION OF CAMBRIDGE ASSESSMENT

LLOYD'S REGISTER · LRQA
ISO 9001

UKAS MANAGEMENT SYSTEMS
001