Learning Objectives for COP 2220 – C Programming

In *Computer Science Curriculum 2008, Report from the Interim Task Force, ACM, IEEE*, the authors describe Learning Objectives like this:

Learning Objectives are central components of any body of knowledge; basically they capture important elements that are typically absent from a mere list of knowledge topics. They are intended to capture what students are able to *do* with knowledge. This typically varies from simple recall to include, for instance, more sophisticated process-oriented skills. Given the rate of change of knowledge, it is often argued with considerable justification that these skills ought to form the basis of curriculum development as they are fundamental to life-long learning. If properly selected, they encourage the student to make effective use of new knowledge as it becomes available.

I've looked through the complete list of Learning Objectives from the IEEE and ACM, selected the minority that are relevant and divided them into three categories, based on the approach to learning C that we'll be taking in the class.

1. Objectives numbered with an *L* are language skills and achievements for mastering the C programming language.

2. Objectives numbered with an *S* are outcomes and skills related to C's main application area – Systems Programming, broadly construed. (This encompasses Operating System modules and utilities, as well as embedded and real-time applications).

3. Objectives numbered with an *O* pertain to the application of C to the development and extension of other programming languages, especially scripting languages, such as Lua, Perl, TCL, Python and so on.

While it is very unlikely that anyone will master all of these learning objectives in our one class, we will cover topics that link to many of them. I'll reference these objectives as a part of each major topic we cover.

I've provided a few links for each of these objectives. These aren't the only links, nor necessarily the best ones. They are the ones I liked, or that came up on the first page of the Google search, or both. You're welcome to find your own in addition.

Please read through these objectives and follow some of the links– perhaps they'll stimulate your thinking about what you would like to take away from our class.

Language Related Objectives:

L1: Analyze and explain the behavior of simple programs involving the fundamental programming constructs covered by this unit. (PIC 4, 5, 6, 7, 8, 10, 12, 17, 18)
(http://www.cprogramming.com/tutorial.html#ctutorial,
http://warrior-101.tripod.com/dstut/cbasics.htm,
http://en.wikipedia.org/wiki/C_%28programming_language%29,
http://icecube.wisc.edu/~dglo/c_class/index.html,http://arduino.cc/en/Reference/HomePage,
http://www.arduino.cc/playground/uploads/Main/arduino_notebook_v1-1.pdf)

L2: Modify short programs that use standard conditional and iterative control structures and functions. (PIC 5, 6) (http://www.cprogramming.com/tutorial/c/lesson2.html,

http://www.cprogramming.com/tutorial/c/lesson3.html,
http://en.wikipedia.org/wiki/Iteration#Computing,
http://en.wikipedia.org/wiki/Conditional_%28programming%29)

L3: Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, and the definition of functions. (PIC 3, 4, 5, 6, 8)
(http://www.cprogramming.com/tutorial.html#ctutorial)

L4: Choose appropriate conditional and iteration constructs for a given programming task. (PIC 5, 6) (http://www.cprogramming.com/tutorial/c/lesson2.html,
http://www.cprogramming.com/tutorial/c/lesson3.html,
http://en.wikipedia.org/wiki/Iteration#Computing,
http://en.wikipedia.org/wiki/Conditional_%28programming%29)

L5: Apply the techniques of structured (functional) decomposition to break a program into smaller pieces. (PIC 8) (http://en.wikipedia.org/wiki/Decomposition_%28computer_science%29,
http://www.cprogramming.com/tutorial/c/lesson4.html)

L6: Describe the mechanics of parameter passing. (PIC 8)
(http://www.cs.usfca.edu/~wolber/SoftwareDev/C/paramPassing.htm,
http://www.cs.fsu.edu/~engelen/courses/COP402001/notes7_4.pdf,
http://homepages.ius.edu/jfdoyle/c335/Html/CNotes.htm#Call-by-Value%20versus%20Call-by-Reference,
http://en.wikipedia.org/wiki/Parameter_%28computer_science%29)

L7: Describe strategies that are useful in debugging. (PIC 18)
(http://www.cprogramming.com/debugging/debugging_strategy.html,
http://cprogramming.com/beginner_programming_mistakes.html)

L8: Describe the representation of numeric and character data. (PIC 4)
(http://www.willamette.edu/~gorr/classes/cs130/lectures/data_rep.htm,
http://www.cs.fsu.edu/~engelen/courses/COP402001/notes7_4.pdf,
http://www.cprogramming.com/tutorial/c/lesson9.html,
http://www.cprogramming.com/tutorial/string.html,
http://www.cprogramming.com/tutorial/unicode.html)

L9: Understand how precision and round-off can affect numeric calculations. (PIC 14)
(http://www.cprogramming.com/tutorial/floating_point/understanding_floating_point.html,
http://www.dspguide.com/ch4/1.htm)

L10: Discuss the use of primitive data types and built-in data structures. (PIC 4)
(http://www.cppreference.com/wiki/data_types,
http://en.wikipedia.org/wiki/C_syntax#Primitive_data_types,
http://www.cprogramming.com/tutorial/c/lesson11.html)

L11: Implement the user-defined data structures in a high-level language.(PIC 9)
(http://www.cprogramming.com/tutorial/c/lesson7.html,
http://c-faq.com/~scs/cclass/int/sx1.html)

L12: Choose the appropriate data structure for modeling a given problem. (PIC 9) (http://books.google.com/books?id=yy2qKCf2_UYC&lpg=RA3-PA472&ots=xfyc-cRRfN&dq=choosing%20data%20structures%20c&pg=RA3-PA472#v=onepage&q=choosing%20data%20structures%20c&f=false, http://scienceblogs.com/goodmath/goodmath/programming/data_structures/, http://www.codeguru.com/forum/showthread.php?t=479994)

L13: Identify and describe the properties of a variable such as its associated address, value, scope, persistence, and size. (PIC 4) (http://irc.essex.ac.uk/www.iota-six.co.uk/c/d5_variable_scope.asp, http://icecube.wisc.edu/~dglo/c_class/scope.html, http://icecube.wisc.edu/~dglo/c_class/vstorage.html, http://www.cprogramming.com/tutorial/statickeyword.html)

L14: Discuss type incompatibility. (PIC 4) (http://icecube.wisc.edu/~dglo/c_class/promo_conv.html, http://www.cprogramming.com/tutorial/c/lesson11.html, http://en.wikipedia.org/wiki/Compatibility_of_C_and_C%2B%2B)

L15: Demonstrate the difference between call-by-value and call-by-reference parameter passing. (PIC 8) ((http://www.cs.usfca.edu/~wolber/SoftwareDev/C/paramPassing.htm, http://www.cs.fsu.edu/~engelen/courses/COP402001/notes7_4.pdf, http://homepages.ius.edu/jfdoyle/c335/Html/CNotes.htm#Call-by-Value%20versus%20Call-by-Reference, http://en.wikipedia.org/wiki/Parameter_%28computer_science%29)

L16: Select, with justification, an appropriate set of tools to support the development of a range of software products. (http://www.cprogramming.com/tools.html, http://en.wikipedia.org/wiki/Comparison_of_Java_and_C%2B%2B, http://www.freewebs.com/godaves/javabench_revisited/,
http://arduino.cc,
http://www.bloodshed.net/devcpp.html,
http://www.atmel.com/dyn/Products/tools_card.asp?tool_id=2725,
http://www.microsoft.com/downloads/details.aspx?displaylang=en&FamilyID=a22341ee-21db-43aa-8431-40be78461ee0)

System programming related Objectives:

S1: Appreciate the effect of AND, OR, NOT and EOR operations on binary data (PIC 12)
(http://www.cprogramming.com/tutorial/bitwise_operators.html,
http://www.arduino.cc/en/Reference/Bitwise,
http://www.arduino.cc/en/Reference/BitwiseAnd,
Google for “bebop boolean boogie” and read the book,
http://www.hackersdelight.org/basics.pdf)

S2: Be aware of the various classes of instruction: data movement, arithmetic, logical, and flow control.
(http://www.avrbeginners.net/,
http://www.geocities.com/siliconvalley/park/3230/pas/lowlevel.html,

S3: Appreciate how conditional operations are implemented at the machine level.
(http://www.pcengines.ch/tp3.htm,
http://homepages.cwi.nl/~steven/pascal/book/7statements.html,
http://www.avrbeginners.net/)

S4: Understand the way in which subroutines are called and returns made. (PIC 8)
(http://homepages.ius.edu/jfdoyle/c335/Html/CNotes.htm#Call-by-Value%20versus%20Call-by-Reference,
http://www.ethernut.de/en/documents/arm-inline-asm.html)

S5: Explain how interrupts are used to implement I/O control and data transfers. (
http://www.best-microcontroller-projects.com/hardware-interrupt.html,
http://www.arduino.cc/playground/Code/Interrupts,
http://www.embedded.com/story/OEG20010725S0105)

S6: Understand how analog quantities such as pressure can be represented in digital form and how the use of a finite representation leads to quantization errors.
(http://www.dspguide.com/ch3.htm,
http://itp.nyu.edu/physcomp/sensors/Reports/Reports,
http://www.seattlerobotics.org/encoder/jul97/basics.html,
http://sensorworkshops2008.blogspot.com/,
http://itp.nyu.edu/physcomp/sensors/Class/Manufacturers)

S7: Appreciate how typical input devices operate
(http://www.arduino.cc/playground/Main/InterfacingWithHardware,
http://www.freeduino.org/,
http://itp.nyu.edu/physcomp/sensors/Main/HomePage)

S8: Describe what makes a system a real-time system.
(http://en.wikipedia.org/wiki/Real-time_system,
http://www.linuxfordevices.com/c/a/Linux-For-Devices-Articles/RTAI-RealTime-Application-Interface/,
http://www.dedicated-systems.com/encyc/publications/faq/rtfaq.htm)

S9: Explain the presence of and describe the characteristics of latency in real-time systems.
(http://www-03.ibm.com/linux/realtime.html,

http://rt.wiki.kernel.org/index.php/HOWTO:_Build_an_RT-application,
http://billgrundmann.wordpress.com/2009/03/02/the-overhead-of-arduino-interrupts/)

S10: Summarize special concerns that real-time systems present and how these concerns are addressed. (http://www.mindspring.com/~tcoonan/design.html,
http://www.best-microcontroller-projects.com/switch-debounce.html,
ftp://ftp.seu.edu.cn/Pub2/EBooks/Books_from_EngnetBase/pdf/7001/7001_PDF_C73.pdf)

S11: Compare and contrast compiled and interpreted execution models, outlining the relative merits of each. (PIC 2) (http://6004.csail.mit.edu/Fall01/handouts/L13-4up.pdf,
http://www.perl.com/doc/FMTEYEWTK/comp-vs-interp.html)

S12: Explain the differences between machine-dependent and machine-independent translation and where these differences are evident in the translation process.
(http://6004.csail.mit.edu/Fall01/handouts/L13-4up.pdf,
http://www.cs.utsa.edu/~qingyi/cs5363/slides/optimization.pdf,
http://www.velocityreviews.com/forums/t139141-jre-is-machine-dependent-but-compiler-is-machine-independent.html)

S13: Explain the uncertainties associated with sensors and how to deal with those uncertainties.
(http://www.dspguide.com/ch2/7.htm,
http://www.dspguide.com/ch15/2.htm,
http://itp.nyu.edu/physcomp/sensors/Class/SensorsAndTime,
http://colorforms.vox.com/library/post/sensor-workshop-actions-events-and-filtering.html)

S14: Design a simple control architecture.
(http://en.wikipedia.org/wiki/Control_system,
http://www.cds.caltech.edu/~murray/courses/cds101/fa02/caltech/astrom-ch4.pdf,

Other Objectives:

O1: Discuss the challenges of maintaining legacy software.

O2: Demonstrate the typical functionality of a scripting language, and interpret the implications for programming.

O3: Implement a simple script that exhibits parameter passing

O4: To be aware of the range of possibilities for games engines, including their potential and their limitations

O5: To use a games engine to construct a simple game

References

I've abstracted these from the (much longer) Computer Science curriculum by the ACM and IEEE described in: *Computer Science Curriculum 2008, Report from the Interim Task Force, ACM, IEEE*.

http://www.computer.org/portal/cms_docs_ieeecs/ieeecs/education/cc2001/ComputerScience2008.pdf