CLA: PROGRAMMING ESSENTIALS IN C

OVERVIEW

DEVELOPED BY

C++ INSTITUTE

CONTENTS

OVERVIEW

CLA: PROGRAMMING ESSENTIALS IN C

- For beginners with little or no prior knowledge of programming.
- Designed to be a full semester course: 9 chapters, 16 quizzes and chapter assessments, 80+ lab exercises, pre-final and final tests.
- Accessed online with no special equipment or system requirements.
- Aligns to C++ Institute *CLA – C Programming Language Certified Associate Certification.*
- Instructor-Led Training offered at no cost.

TARGET AUDIENCE

The *CLA: Programming Essentials in C* curriculum is designed for students who want to learn the fundamentals of programming through the C language.

CURRICULUM DESCRIPTION

This course covers the universal concepts of computer programming, as well as the syntax, semantics and data types of the C language.

TARGET CERTIFICATION

The *CLA: Programming Essentials in C* curriculum helps students prepare for the *CLA – C Programming Language Certified Associate certification* exam. *C Programming Language Certified Associate (CLA)* is a professional certification that measures the ability to accomplish coding tasks related to the basics of programming in the C language, as well as fundamental programming techniques, customs and vocabulary, including the most common library functions and the usage of the preprocessor.

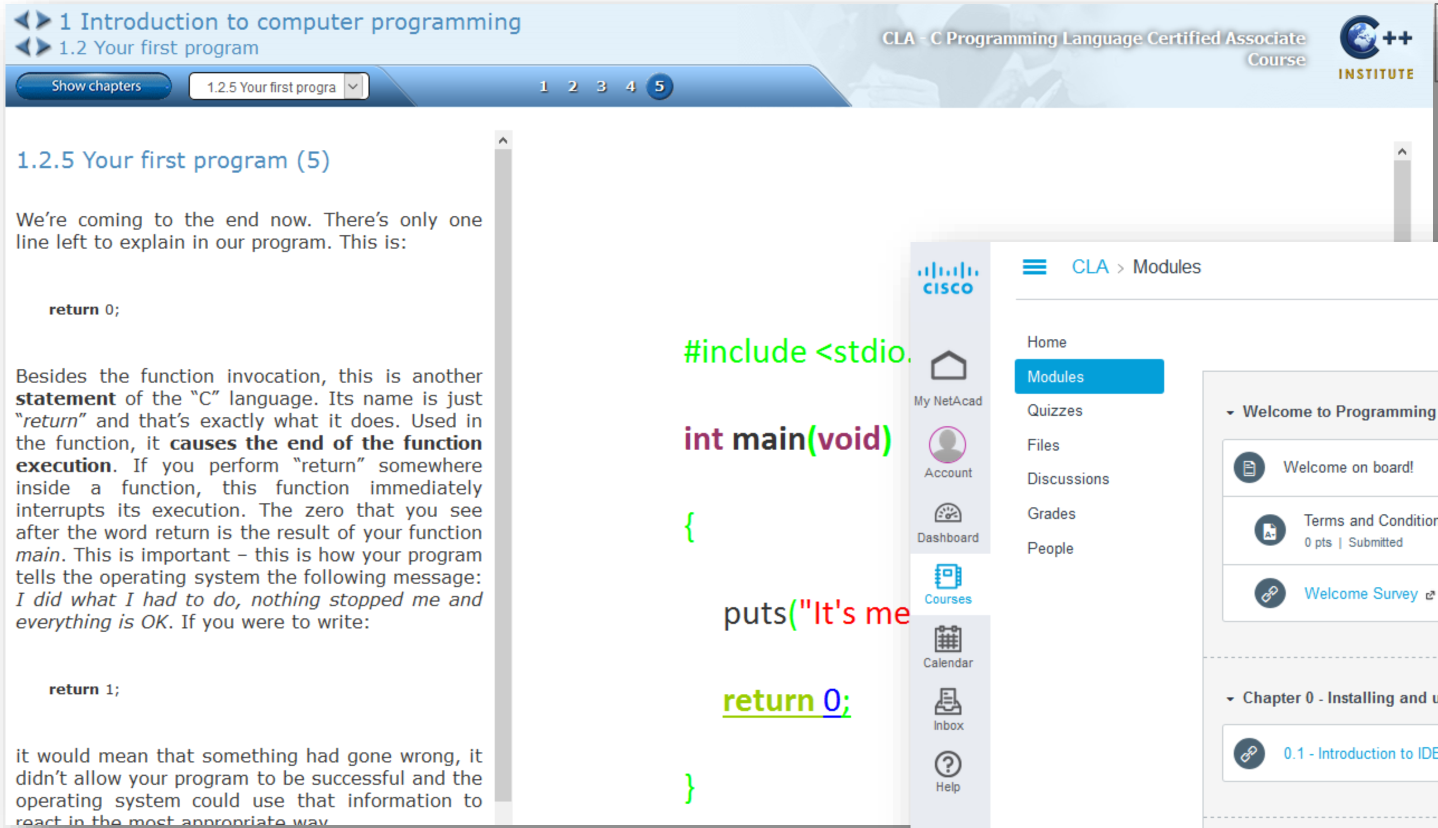
COURSE DESIGN

The course is broken down into 9 modules:

- Module 0: explains the process of installing and using the programming environment.
- Module 1: introduces common computer programming concepts, e.g. integers and variables.
- Module 2: introduces the concepts of data types and flow control.
- Module 3: further discusses the concepts of flow control, introducing more data types and logic in computer science.
- Module 4: touches on the subject of aggregating data into arrays.
- Module 5: explains the differences between arrays and structures.
- Module 6: introduces the subject of functions.
- Module 7: discusses files and streams.
- Module 8: elaborates on the subject of the pre-processor and declarations.

Each student has access to hands-on practice materials, quizzes and assessments to learn how to utilize the skills and knowledge gained on the course and interact with some real-life programming tasks and situations.

COURSEWARE



1 Introduction to computer programming
1.2 Your first program

CLA - C Programming Language Certified Associate Course

Show chapters | 1.2.5 Your first progra | 1 2 3 4 5

1.2.5 Your first program (5)

We're coming to the end now. There's only one line left to explain in our program. This is:

```
return 0;
```

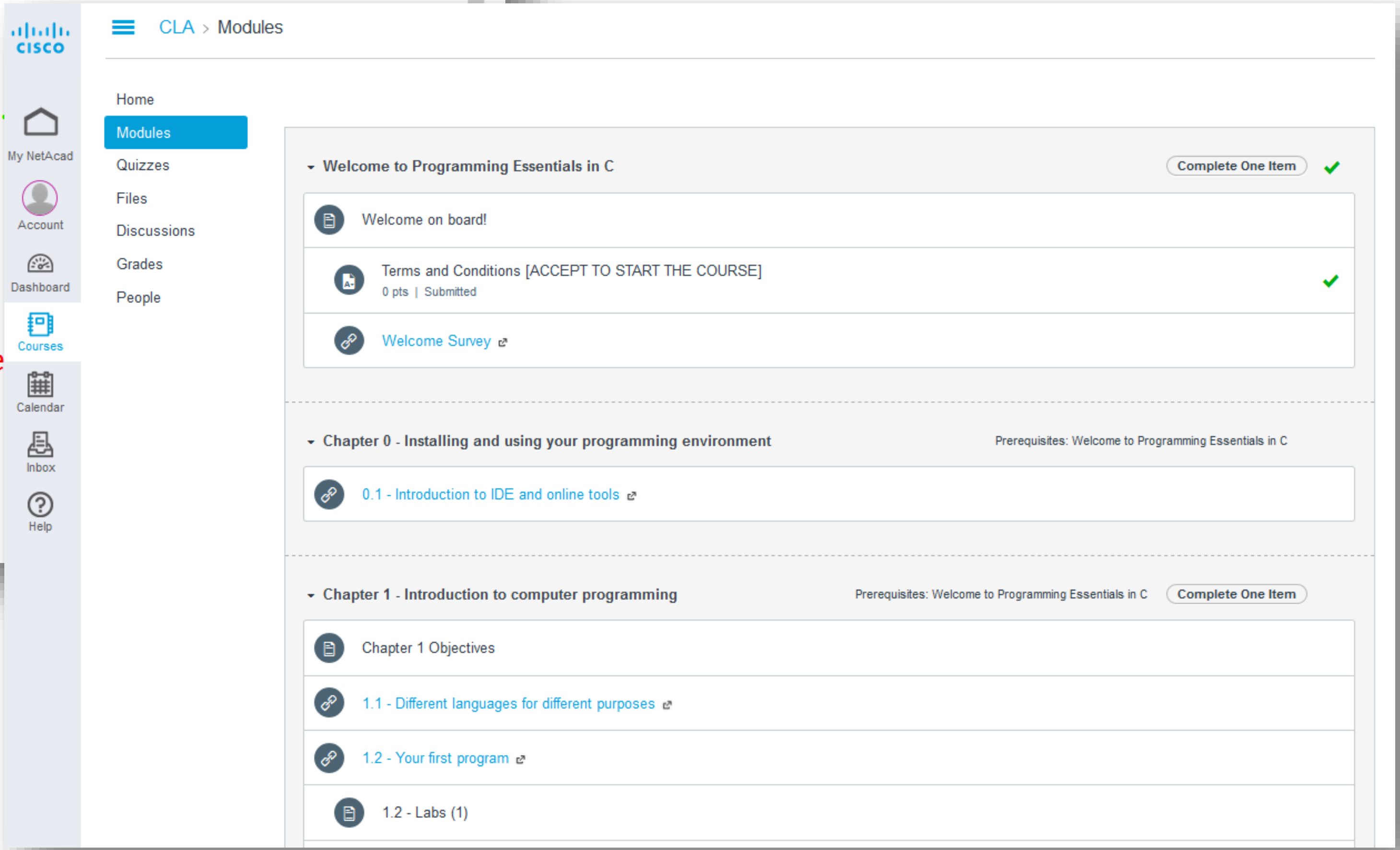
Besides the function invocation, this is another **statement** of the "C" language. Its name is just "*return*" and that's exactly what it does. Used in the function, it **causes the end of the function execution**. If you perform "return" somewhere inside a function, this function immediately interrupts its execution. The zero that you see after the word return is the result of your function *main*. This is important – this is how your program tells the operating system the following message: *I did what I had to do, nothing stopped me and everything is OK*. If you were to write:

```
return 1;
```

it would mean that something had gone wrong, it didn't allow your program to be successful and the operating system could use that information to react in the most appropriate way.

```
#include <stdio.
int main(void)
{
    puts("It's me
    return 0;
}
```

Course Page in Netacad.com



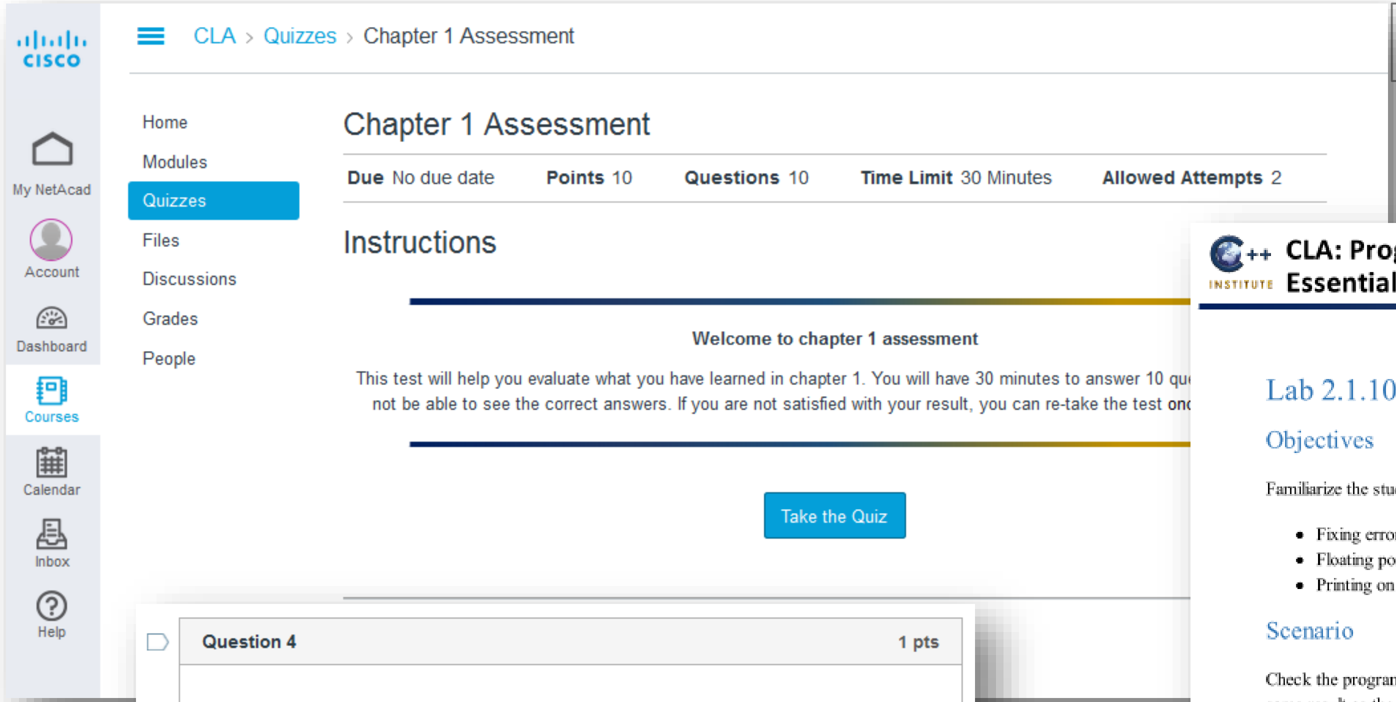
CLA > Modules

- Welcome to Programming Essentials in C — Complete One Item ✓
  - Welcome on board!
  - Terms and Conditions [ACCEPT TO START THE COURSE] — 0 pts | Submitted ✓
  - Welcome Survey
- Chapter 0 - Installing and using your programming environment — Prerequisites: Welcome to Programming Essentials in C
  - 0.1 - Introduction to IDE and online tools
- Chapter 1 - Introduction to computer programming — Prerequisites: Welcome to Programming Essentials in C — Complete One Item
  - Chapter 1 Objectives
  - 1.1 - Different languages for different purposes
  - 1.2 - Your first program
  - 1.2 - Labs (1)

Course Content

COURSEWARE

Chapter Assessments and Quizzes



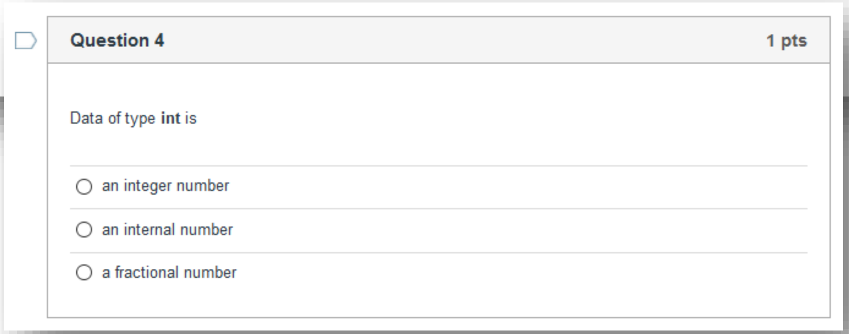
Chapter 1 Assessment

Due No due date **Points** 10 **Questions** 10 **Time Limit** 30 Minutes **Allowed Attempts** 2

Instructions

Welcome to chapter 1 assessment

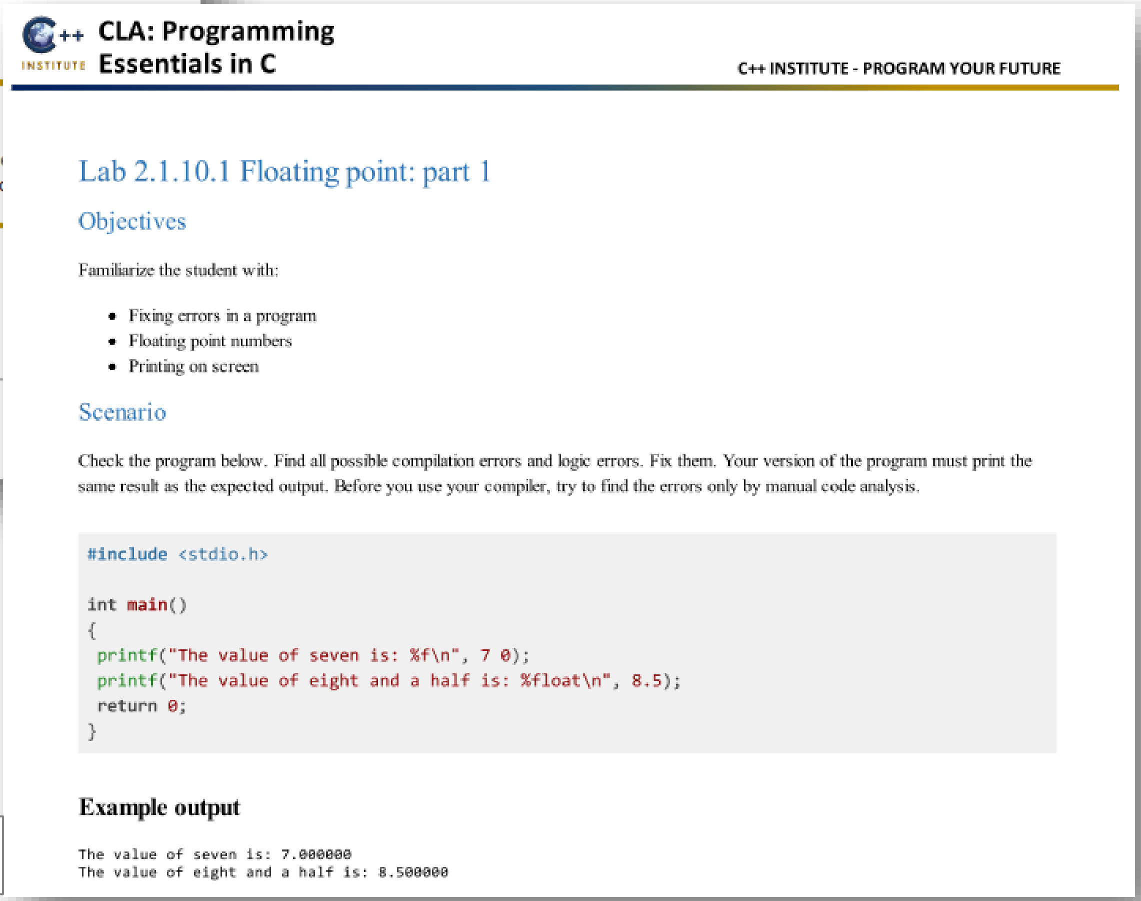
This test will help you evaluate what you have learned in chapter 1. You will have 30 minutes to answer 10 qu
not be able to see the correct answers. If you are not satisfied with your result, you can re-take the test onc



Question 4 1 pts

Data of type **int** is

- ○ an integer number
- ○ an internal number
- ○ a fractional number


C++ INSTITUTE
CLA: Programming Essentials in C
C++ INSTITUTE - PROGRAM YOUR FUTURE

Lab 2.1.10.1 Floating point: part 1

Objectives

Familiarize the student with:

- Fixing errors in a program
- Floating point numbers
- Printing on screen

Scenario

Check the program below. Find all possible compilation errors and logic errors. Fix them. Your version of the program must print the same result as the expected output. Before you use your compiler, try to find the errors only by manual code analysis.

```
#include <stdio.h>

int main()
{
 printf("The value of seven is: %f\n", 7 0);
 printf("The value of eight and a half is: %float\n", 8.5);
 return 0;
}
```

Example output

```
The value of seven is: 7.000000
The value of eight and a half is: 8.500000
```

Lab exercises

SCOPE AND SEQUENCE

CURRICULUM OBJECTIVES

The aim of the course is to:

- familiarize the student with the universal concepts of computer programming,
- present the syntax, semantics and basic data types of the C language,
- discuss the customs and vocabulary of the C language, including the most common library functions and the usage of the pre-processor
- align the course to the C++ Institute *CLA – C Programming Language Certified Associate certification*.

COURSE OBJECTIVES

During the course, students will study the following objectives:

- Introduction to compiling and software development,
- Basic scalar data types and their operators,
- Flow control,
- Complex data types: arrays, structures and pointers,

- Memory management,
- Functions,
- Files and streams,
- Structuring the code: functions and modules,
- Preprocessor directives and complex declarations.

COURSE OUTLINE

0 – Installing and using your programming environment	• introduction to compiling and software development.
1 – Introduction to computer programming	• languages: natural and artificial, • machine languages, • high-level programming languages, • obtaining the machine code: compilation process, • writing simple programs, • variables, • integer values in real life and in C, • integer literals.

COURSE OUTLINE (cont.)

2 – Data types	• floating point values in real life and in C, • float literals, • arithmetic operators, • priority and binding, • post- and pre-incrementation and decrementation, • operators of type op=, • char type and ASCII code, • char literals, • equivalence of int and char data, • comparison operators, • conditional execution and if keyword, • printf() and scanf() functions

COURSE OUTLINE (cont.)

3 – Flow control	• conditional execution: the "else" branch, • integer and float types, • conversions, • typecast and its operators, • loops – while, do and for, • controlling the loop execution – break and continue, • logical and bitwise operators.
4 – Arrays	• switch: different faces of 'if', • arrays (vectors), • sorting in real life and in a computer memory, • initiators, • pointers,

COURSE OUTLINE (cont.)

4 – Arrays (cont.)	• an address, a reference, a dereference and the sizeof operator, • simple pointer and pointer to nothing (NULL), • & operator, • pointers arithmetic, • pointers vs. arrays: different forms of the same phenomenon, • using strings, • basic functions dedicated to string manipulation.

5 - Memory management and structures	• array indexing, • the usage of pointers: perils and disadvantages, • void type, • arrays of arrays and multidimensional arrays,

COURSE OUTLINE (cont.)

5 - Memory management and structures (cont.)	• memory allocation and deallocation: malloc() and free() functions, • arrays of pointers vs. multidimensional arrays, • structures, • declaring, using and initializing structures, • pointers to structures and arrays of structures, • basics of recursive data collections.

6 – Functions	• functions, • how to declare, define and invoke a function, • variables' scope, local variables and function parameters, • pointers, arrays and structures as function parameters,

COURSE OUTLINE (cont.)

6 – Functions (cont.)	• function result and return statement, • void as a parameter, pointer and result, • parameterizing the main function, • external function and the extern declarator, • header files and their role.
7 – Files and streams	• files vs. streams, • header files needed for stream operations, • FILE structure, • opening and closing a stream, open modes, errno variable, • reading and writing to/from a stream, • predefined streams: stdin, stdout and stderr,

COURSE OUTLINE (cont.)

7 – Files and streams (cont.)	• stream manipulation: fgetc(), fputc(), fgets() and fputs() functions, raw input/output: fread() and fwrite() functions.
8 – Preprocessor and complex declarations	• preprocessor, • #include: how to make use of a header file, • #define: simple and parameterized macros, #undef directive, • predefined preprocessor symbols, • macrooperators: # and ##, • conditional compilation: #if and #ifdef directives, • avoiding multiple compilations of the same header files, • scopes of declarations, storage classes, • user–defined types, pointers to functions, • analyzing and creating complex declarations.

HOW TO USE THE COURSE

ACADEMIC INSTITUTIONS

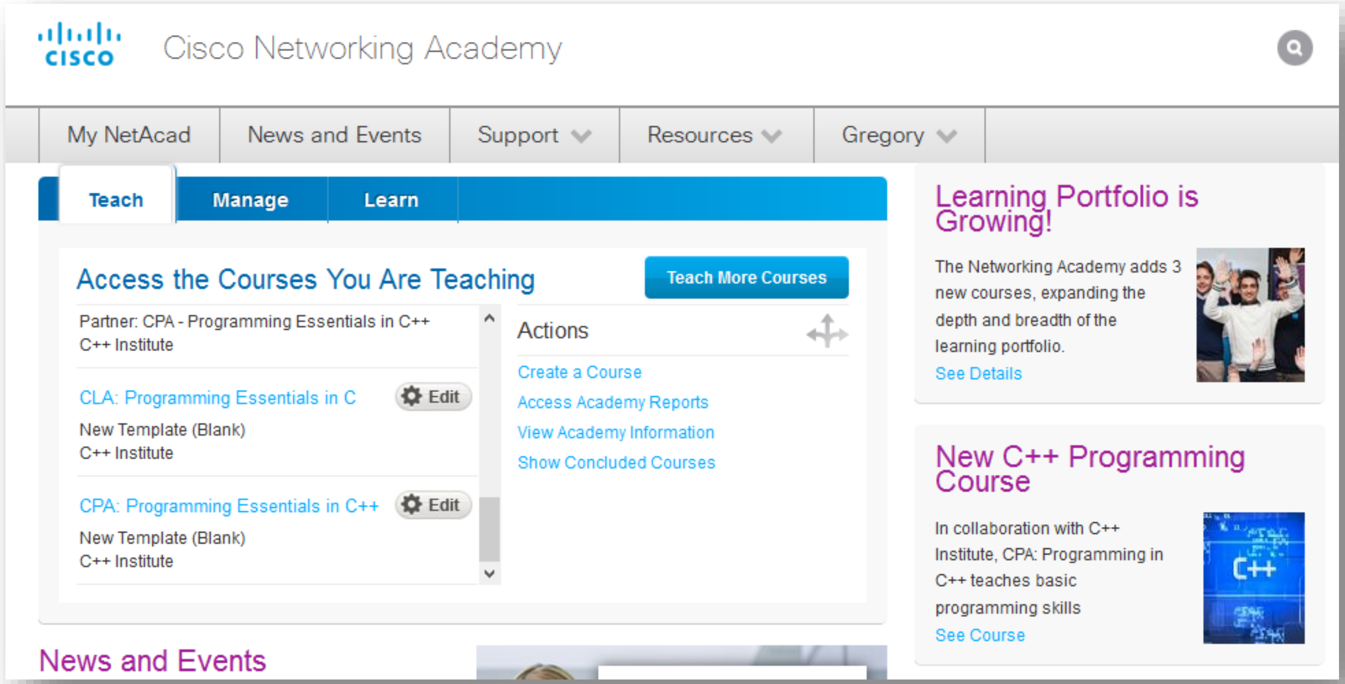
Academic institutions can use this course as follows:

- offer the course as a complete full semester course
- create interest and motivate new students to learn the fundamentals of computer programming
- motivate those students who already know another programming language to learn C
- supplement an existing C language course
- help students prepare for the *CLA – C Programming Language Certified Associate certification*
- introduce Netacad.com to your peers and colleagues

There are **no formal requirements for instructors** to teach *CLA: Programming Essentials in C*. However, the C++ Institute recommends that instructors earn a *CLA – C Programming Language Certified Associate Certification* prior to teaching the class.

HOW TO SET UP A CLASS IN NETACAD.COM

1. Go to the Netacad.com home page at www.netacad.com and sign in.
2. Select the **TEACH** tab.
3. Select the **CREATE A COURSE** link.
4. Enter the course information (select an **Academy**, enter a **Course Name** and **Course ID**. Then Select the course **Partner: CLA: Programming Essentials in C**, choose a language for the course, enter a **start** and **coclude date**, and select the **instructor**.
5. Click **Save** to set up your class.



Access the Courses You Are Teaching

Partner: CPA - Programming Essentials in C++
C++ Institute

CLA: Programming Essentials in C
New Template (Blank)
C++ Institute

CPA: Programming Essentials in C++
New Template (Blank)
C++ Institute

Create a Course
Access Academy Reports
View Academy Information
Show Concluded Courses

Learning Portfolio is Growing!

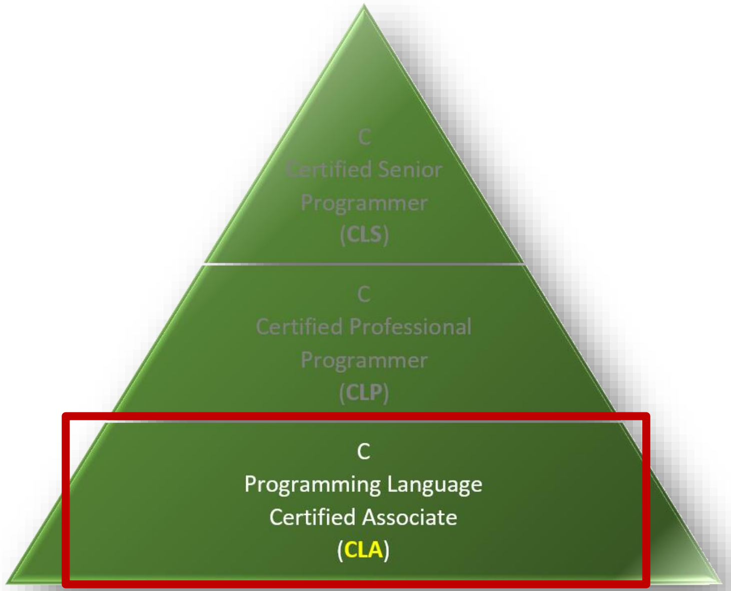
The Networking Academy adds 3 new courses, expanding the depth and breadth of the learning portfolio.

New C++ Programming Course

In collaboration with C++ Institute, CPA: Programming in C++ teaches basic programming skills

CERTIFICATION

C Programming Language Certified Associate (CLA) is a professional certification that measures your ability to accomplish coding tasks related to the basics of programming in the C programming language, as well as fundamental programming techniques, customs and vocabulary, including the most common library functions and the usage of the preprocessor.



- Professional certification
- Associate level
- Delivered through the network of Pearson VUE Test Centers
- Digital transcript, badge, and paper certification
- Complete the *CLA: Programming Essentials in C* course and get a **51% discount** for the certification exam!

C++ INSTITUTE CERTIFICATION – SURVEY RESULTS

83% of respondents said that obtaining a C++ Institute certification had directly translated into receiving some career benefit.

62% of respondents said that obtaining a C++ Institute certification had a positive impact on professional image and reputation.

49% of respondents said that earning a C++ Institute certification had helped them to do their job more confidently.

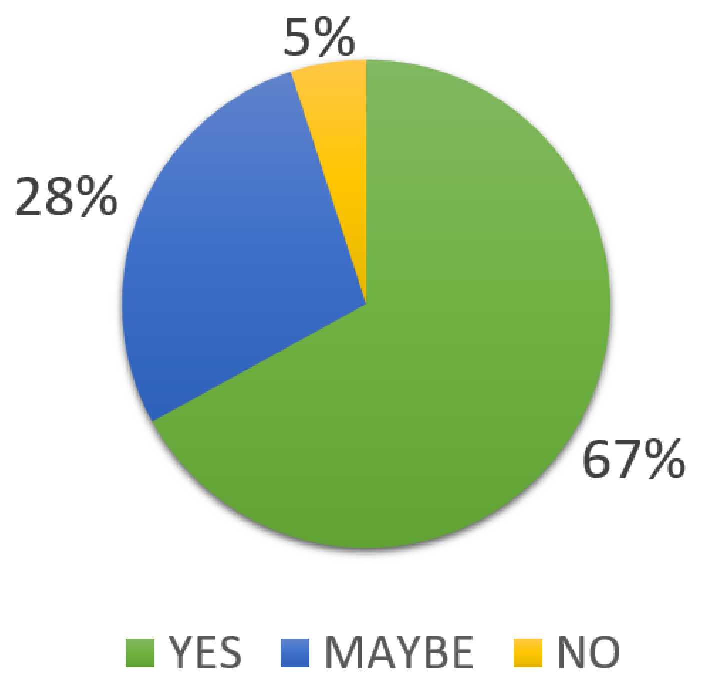
41% of respondents claimed that *learning new things* was the biggest benefit from obtaining a C++ Institute certification.

75% of respondents said they had experienced the first benefit of obtaining a C++ Institute certification within 3 months.

Source: *The 2016 Value of C++ Institute Certification Report* based on a survey conducted online from June 22 to June 24, 2016, among C/C++ certified individuals. The survey was conducted by Fundacja IT and C++ Institute.

C++ INSTITUTE CERTIFICATION – SURVEY RESULTS

Would you recommend C++ Institute certification to your friends or colleagues when discussing a career or advancement in IT/programming?



78% of respondents said that using C++ Institute study materials had **increased confidence to take the certification exam** and had **played a role in passing the certification exam**.

73% of respondents said they wanted to take another C++ Institute exam in the future.

Source: *The 2016 Value of C++ Institute Certification Report* based on a survey conducted online from June 22 to June 24, 2016, among C/C++ certified individuals. The survey was conducted by Fundacja IT and C++ Institute.

WHY LEARN PROGRAMMING

FOR SEVERAL REASONS.

- To become a **creator**: a highly **creative** and **powerful** one. Go as far as your imagination lets you.
- Strong programming skills are a **hot commodity** on the job market!
- Boost your **earning potential**!
- Programming is the language of the **future**.
- Learning to program means learning to **think in abstract** and more **precise ways**.
- It will help you **do better in other areas**!
- It will be **fun**!

WHY LEARN C

FOR SEVERAL REASONS.

- It is **omnipresent**, people use numerous C powered devices on a daily basis, whether they realize it or not.
- There have been millions (well, actually **billions**) of lines of code written in C, which means almost unlimited opportunities for **code reuse** and learning from well-crafted examples.
- It is **close to the machine**, which actually lets you understand how computers work!
- It has been the **backbone** of a number of other languages (e.g. C++, Java, or Python have borrowed many basic features directly or indirectly from C).
- C is the **lingua franca** of programming!
- It is **fast.**
- There is a **large** and very **active C/C++ community.**
- It will give you a **solid foundation** and allow you to learn other programming languages (e.g. C++, Java, or C#) much easier and much faster.
- It will be **fun**!

C/C++ EXAMPLES

DID YOU KNOW...?

Do you remember **Doom III**, **StarCraft**, **Master of Orion III**, or **Warcraft III**? You have probably played (or at least heard of) **Diablo I** or **Diablo II**?

If you like computer games, then you must have heard of **Electronic Arts**. All of these games have to do with C/C++ programming.

The truth is that a large majority of **computer games and game engines** have been developed in C/C++. Electronic Arts' video game engine and (probably) all **Microsoft games** are no exception.

Not surprisingly, most of the **operating systems** are written in the C/C++ languages. These not only include Windows and Linux (the Linux kernel is almost entirely written in C), but also Google Chrome OS, RIM Blackberry OS 4.x, Symbian OS, Apple Mac OS X, iPAD OS, Apple iPhone iPod Touch, and Cisco IOS (which is mainly comprised of compiled C and C++ code).

C/C++ EXAMPLES

Think of **Internet Browsers** like Microsoft Internet Explorer, Google Chrome, Mozilla Firefox, Safari, Netscape Navigator and Opera. Yes, they all, too, were developed in C/C++.

And what about the major **websites**? Google? Facebook? Twitter? YouTube? Amazon? PayPal?
Yes. They were all written, to a greater or lesser extent, in C/C++. Other examples?

- **Microsoft Office products** (Word, Excel, Access, PowerPoint, etc.)
- **e-mail clients** (Microsoft Outlook, Mozilla Thunderbird, IBM Lotus)
- **Multimedia players** (Winamp, Windows Media Player, VLC media player, and Apple iPod software)
- **Database systems** (Oracle database, MySQL, IBM DB2, Microsoft SQL Server, IBM Informix, SAP DB/MaxDB, and MongoDB)
- **Graphical User Interface** (Microsoft Windows UI, Apple MacOS UI - Aqua, and KDE)
- **compilers and virtual machines** for programming languages (such as Microsoft Visual C++ Compiler, Microsoft Visual Basic Compiler, Microsoft Visual C# Compiler, Microsoft .NET CLR, or Java Virtual Machine – JVM)
- and **thousands of other examples** including: Sun Microsystem's compilers, Solaris OS, Google File System, Google Earth and Picasa, Adobe's Photoshop, Illustrator, Acrobat Reader, InDesign, Intel's chip design and manufacturing software, IBM's OS/400 and K42, Microsoft's DirectX, Exchange Server, and Visual Studio, CERN data analysis applications, Bloomberg, Autodesk's applications, e.g. Autodesk Maya, 12D, Vodaphone infrastructure, and FlightGear...

Source: http://www.stroustrup.com/applications.html

KEY TAKEAWAYS

- *CLA: Programming Essentials in C* is developed by the C++ Institute
- The course introduces your students to computer programming using the C language
- The course aligns to the *C++ Institute CLA – C Programming Language Certified Associate* certification
- The C++ Institute provides all content
- The course is available in Netacad.com
- Students who successfully complete the course and pass the final test will receive a 51% discount for the *CLA – C Programming Language Certified Associate* certification exam at Pearson VUE