课程详述

COURSE SPECIFICATION

以下课程信息可能根据实际授课需要或在课程检讨之后产生变动。如对课程有任何疑问，请联系授课教师。

The course information as follows may be subject to change, either during the session because of unforeseen circumstances, or following review of the course at the end of the session. Queries about the course should be directed to the course instructor.

1.	**课程名称 Course Title**	**计算机高级程序设计 Advanced Programming**
2.	**授课院系 Originating Department**	计算机科学与工程系 Department of Computer Science and Engineering
3.	**课程编号 Course Code**	CS219
4.	**课程学分 Credit Value**	3
5.	**课程类别 Course Type**	专业选修课 Major Elective Courses
6.	**授课学期 Semester**	秋季 Fall/春季 Spring
7.	**授课语言 Teaching Language**	英文 English
8.	**授课教师、所属学系、联系方式（如属团队授课，请列明其他授课教师） Instructor(s), Affiliation& Contact （For team teaching, please list all instructors）**	于仕琪，副教授，计算机科学与工程系，yusq@sustech.edu.cn Shiqi Yu, Associate professor, Department of Computer Science and Engineering, yusq@sustech.edu.cn 郑锋，副教授，计算机科学与工程系，zhengf@sustech.edu.cn Feng Zheng, Associate Professor, Department of Computer Science and Engineering zhengf@sustech.edu.cn
9.	**实验员/助教、所属学系、联系方式 Tutor/TA(s), Contact**	
10.	**选课人数限额(可不填) Maximum Enrolment （Optional）**	

11.	授课方式 Delivery Method	讲授 Lectures	习题/辅导/讨论 Tutorials	实验/实习 Lab/Practical	其它(请具体注明) Other（Please specify）	总学时 Total
	学时数 Credit Hours	32		32		64
12.	先修课程、其它学习要求 Pre-requisites or Other Academic Requirements	CS109 计算机程序设计基础				
13.	后续课程、其它学习规划 Courses for which this course is a pre-requisite					
14.	其它要求修读本课程的学系 Cross-listing Dept.					

教学大纲及教学日历 SYLLABUS

15. 教学目标 Course Objectives

本课程是为计算机和相关专业本科生设计，是“计算机程序设计基础”之后的第二门编程语言。本课程的目的是帮助学生进一步提升程序设计能力，加深对计算机的理解，提升解决问题的能力。本课程将主要介绍 C 和 C++编程语言，以及这两门语言在系统开发和计算中的应用；除了 C 和 C++，本课程还会介绍 Rust 入门以及 Rust 的独特之处。C 是一种面向问题的通用程序设计语言，具有语言简洁、类型丰富、结构完整、表达力强、直接操作内存单元、适用于模块化结构等特点。C 语言既具有高级语言的优点，又具有低级语言的许多特点。C++是面对对象开发方法，从 C 语言扩展而来。吸收了软件工程领域有益的概念和有效方法，它把数据和对数据的操作封装起来，集抽象性、封装性、继承性与多态性于一体，可以帮助人们开发出模块化、数据抽象程度高的、信息隐蔽好的、可复用、易修改、易扩充等特性的程序。Rust 是一种新型的编译型语言，设计准则为“安全、并发、实用”，支持函数式、并行式、过程式以及面向对象的程式设计风格，关注度和使用量与日俱增。

本课程专注 C 和 C++编程语言的独特之处，特别是在程序运行效率方面的优势。在基础知识点之外，本课程将通过案例来介绍 C 和 C++在计算领域的优势，并使学生了解计算机系统开发和深度学习底层开发。除此之外，本课程还将介绍新型的编程语言 Rust，让学生可以了解编程语言的发展趋势。

This course is designed for undergraduate students majoring in computer science and related fields, serving as the second programming language course following "Introduction to Computer Programming". This course aims to enhance students' programming skills, improve their understanding of computers, and improve their problem-solving abilities. It will introduce the C and C++ programming languages, as well as their applications in system development and computing. The Rust programming languages will also be introduced in the course. C is a general-purpose programming language known for its conciseness, rich types, complete structures, strong expressiveness, direct memory manipulation, and suitability for modular structures. It combines the advantages of high-level languages with many features of low-level languages. C++ extends from C with its object-oriented development approach, incorporating beneficial concepts and effective methods from the field of software engineering. It encapsulates data and operations on data, integrating abstraction, encapsulation, inheritance, and polymorphism, facilitating the development of modular, highly abstracted, well-hidden, reusable, easily modifiable, and expandable programs. Rust is a new programming language, designed with the principles of "performance, type safety, and concurrency". It supports functional, parallel, procedural, and object-oriented programming styles and has been gaining increasing attention and usage.

This course focuses on the unique advantages of C and C++ programming languages, particularly their advantages in efficiency. Beyond fundamental concepts, it will use case studies to demonstrate the advantages of C and C++ in computing, allowing students to know the basics of computer system development and deep learning. In addition, this course will also introduce the modern programming language Rust, allowing students to understand the trends in programming language development.

16. 预达学习成果 Learning Outcomes

完成该课程，学生能够做到：

- 能够解释 C/C++语言的工作原理
- 了解 Rust 语言的基本语法和特性
- 能够阅读 C/C++程序，能够找出其中错误并修正错误
- 能够分析实际问题，构建 C/C++程序解决该问题
- 能熟练使用指针和内存管理
- 知道如何提升程序的计算效率
- 具有专业化的编程态度与习惯
- 面对实际问题时，具有编程思维

Upon completion of this course, students should be able to:

- Explain how an existing C/C++ program works
- Know the basic syntax and features of Rust
- Discover errors in a C/C++ program and fix them
- Know programming with pointers in C/C++
- Analyze a problem and construct a C/C++ program to solve it
- Know how to improve the efficiency of a program
- Professional programming attitude and habits
- Programming thinking

17. 课程内容及教学日历 （如授课语言以英文为主，则课程内容介绍可以用英文，如团队教学或模块教学，教学日历须注明主讲人）
Course Contents (in Parts/Chapters/Sections/Weeks. Please notify name of instructor for course section(s), if this is a team teaching or module course.)

Weak 1: Getting Started

- The first example
- Different programming languages
- Compile and link
- Preprocessor and macros
- Simple output and input

[Lab 1]:

- WSL configuration
- VSCode configuration

Weak 2: Data Types and Arithmetic Operators

- Integer numbers
- More integer types
- Floating-point numbers
- Arithmetic operators
- C ++ arithmetic operator

[Lab 2]:

- Printing formatting of numbers

Week 3: Branching and Looping Statements

- if statement
- ?: operator
- Conditional expressions
- while loops
- for loop
- goto statement
- switch statement

[Lab 3]:

- Introduction to Linux commands
- The compilers: gcc and g++

Week 4: Data Structures

- Arrays
- Strings: Array-style strings and the string class
- Structure, union and enumeration
- typedef operator

[Lab 4]:

- An introduction to Makefile.

Week 5: Memory and Pointers

- Pointers
- Pointers and arrays
- Allocate memory in C style
- Allocate memory in C++ style

[Lab 5]:

- An introduction to cmake

Week 6: Basics of Functions

- Functions
- Function parameters
- References
- return statement
- inline function

[Lab 6]:

- An introduction to static libraries

Week 7: Advances in Functions

- Default arguments
- Function overloading
- Function templates
- Function pointers and references
- Recursive functions

[Lab 7]:

- An introduction to dynamic libraries

Week 8: Speedup Your Program

- C and C++ with ARM
- Speedup your program
- An example with SIMD and OpenMP
- CUDA library
- Ascend C of CANN
- How to Avoid memory copy in OpenCV

[Lab 8]:

- SIMD intrinsics
- OpenMP

Week 9: Basics of Classes

- Classes and objects
- Constructors and destructors
- this pointer
- const and static members

[Lab 9]:

- Read from files and the standard input
- Write to files and the standard output

- Linux pipe
- Exercises on classes, constructors, etc

Week 10: Advances in Classes

- Operators in OpenCV
- Operator overloading
- Friend functions
- User defined type conversion
- Increment and decrement operators

[Lab 10]:

- git and GitHub
- Exercises on operator overloading

Week 11: Dynamic Memory Management in Classes

- Some default operations
- An example with dynamic memory
- Solution1: Hard copy
- Solution2: Soft copy
- Smart pointers

[Lab 11]:

- Exercises on dynamic memory management in classes.

Week 12: Class Inheritance

- Improve your source code
- Derived class
- Access control
- Virtual functions
- Inheritance and dynamic memory allocation

- Examples in OpenCV

[Lab 12]:

- Exercises on class inheritance, polymorphism and memeory managment

Week 13: Class Templates and std Library

- Class template
- Template non-type parameters
- Class template specialization
- std classes

[Lab 13]:

- Exercises on class templates

Week 14: Error handling

- Standard output stream and standard error stream
- assert
- Exceptions
- More about exceptions
- nothrow

[Lab 7]:

- Exercises on error handling

Week 15: Nested Classes and RTTI

- Friend classes
- Nested types
- RTTI and type cast operators

[Lab 15]:

- Exercises on RTTI

Week 16: In introduction to Rust

- Basic syntax of Rust
- Good features of Rust

[Lab 16]:

- Matrix addition using Rust

18. **教材及其它参考资料 Textbook and Supplementary Readings**

1. Stephen Prata, C Primer Plus (Sixth Edition), ISBN-13: 978- 0321928429
2. Stephen Prata, C++ Primer Plus (Sixth Edition), ISBN-13: 978-0321776402
3. Jim Blandy, Jason Orendorff, Leonora Tindall, Programming Rust: Fast, Safe Systems Development, ISBN-13: 978-1492052593
4. Robert Love, Linux System Programming: Talking Directly to the Kernel and C Library, ISBN-13: 978-1449339531

课程评估 ASSESSMENT

19.

评估形式 Type of Assessment	评估时间 Time	占考试总成绩百分比 % of final score	违纪处罚 Penalty	备注 Notes
出勤 Attendance				
课堂表现 Class Performance				
小测验 Quiz		5%		
课程项目 Projects		65%		
平时作业 Assignments		5%		
期中考试 Mid-Term Test				
期末考试 Final Exam		25%		
期末报告				

Final Presentation				
其它（可根据需要改写以上评估方式） Others (The above may be modified as necessary)				

20. **记分方式 GRADING SYSTEM**

☑ A. **十三级等级制 Letter Grading**
☐ B. **二级记分制（通过/不通过） Pass/Fail Grading**

课程审批 REVIEW AND APPROVAL

21. **本课程设置已经过以下责任人/委员会审议通过**
This Course has been approved by the following person or committee of authority

教学负责人签字：
日期：