

1. 大學部課程與研究所核心課程之區分說明如下：

本系研究所核心課程選課規定為六科中選修三科，故學生若對於某一科目已有完備核心能力，可以不用選修該科目。目前六門核心課程與大學部相對應之課程如下表：

大學部課程名稱	研究所課程名稱
作業系統	作業系統
電腦網路概論	電腦網路
計算方法概論	計算方法
編譯器設計	編譯器
計算機組織	計算機結構
(無)	多媒體系統

這些科目在大學部與研究所教學目標與內容不同處說明如下：

(1) 作業系統：

在教學目標上，大學部「作業系統」主要是以廣泛性的介紹作業系統提供的服務與應用為主，讓學生了解可以開始了解電腦的基本概念與運作原理，並能在開發應用程式時，能了解其所使用的環境與應用服務的運作模式，以充分利用作業系統提供的 system calls。由於作業系統的版本與其搭配的電腦硬體相當多元，在大學部的課程不會深入探討這些作業系統的個別差異，而是著重在一般觀念的建立。研究所的作業系統相當強調於概念與實作上的對應，於課堂上我們主要以 Linux kernel 為例子，幫助學生清楚的認知課本中所有相關的演算法與設計如何在作業系統中實現出來，並且為何要以該種方式實現，其硬體上及軟體上的限制為何。大學部「作業系統」與研究所「作業系統」的差異條列如下：

項目	大學部作業系統	研究所作業系統
教學目標	<ol style="list-style-type: none"> <li>1. OS components, OS services, system calls and the standard system design</li> <li>2. process abstraction</li> <li>3. threads</li> <li>4. scheduling concepts and algorithms</li> <li>5. critical section problem</li> <li>6. synchronization and using semaphores</li> <li>7. deadlocks</li> <li>8. swapping, memory allocation, and virtual memory</li> <li>9. file system and directory structure</li> </ol>	幫助學生從 Linux kernel 中深刻的瞭解下列主要的議題： Kernel structure Process management Interposes communication Kernel synchronization methods Memory management Virtual file systems I/O subsystem (bottom halves and deferring work)
Overview	To give an overview of OS	Introduction to thin client
Process	process concept,	Virtual machine, virtual

	multithreaded programming, scheduling, synchronization, deadlocks	server farm, Lazy context switch, Linux system call implementation, Helper threading, OpenMP, Scheduling algorithms for SMT Processors, Linux kernel synchronization mechanisms
Memory	memory management strategies, virtual memory management	Buddy system (Linux) Slab(Linux) MMU, TLB and cache
Storage	file system, secondary storage, I/O systems	Journaling file systems Flash file systems Top/bottom halves

(2) 電腦網路:

本系 96 年 11 月 6 日之課程規劃會議已將大學部課程名稱定為「電腦網路概論」，研究所課程名稱為「電腦網路」。在教學目標上，大學部「電腦網路概論」主要是以廣泛性的介紹網路各層相關網路協定與應用為主，讓學生了解電腦網路的基本概念與運作原理，並能在使用 Internet 相關網路應用時，能彼此相互呼應，了解其所使用的網路環境與應用服務的運作模式。由於網路各層協定內容相當多，在大學部的課程不會深入探討這些網路協定與應用服務的深層設計理念與實作。研究所的「電腦網路」則強調廣而深入地了解電腦網路，並能在此領域進行研究為教學目標。課程內容強調從 connectivity, scalability, efficient resource sharing 等三個面向來探討整個 Internet 的設計，並帶學生 trace open source (以 linux kernel 為主) 中與所探討的協定的相關的程式碼，強化學生畢業後從事協定軟體開發的實作能力。如果從 Internet 的四層架構來分析，大學部「電腦網路概論」與研究所「電腦網路」的差異如下:

項目	大學部電腦網路概論	研究所電腦網路
教學目標	<ol style="list-style-type: none"> <li>1. 了解電腦網路的基本概念與運作原理</li> <li>2. 了解在網路各層上的通信協定相關標準</li> <li>3. 學會基本網路程式設計</li> </ol>	<ol style="list-style-type: none"> <li>1. 能從各種面向分析了解網路設計理念與協定效能</li> <li>2. 能解釋協定如何在作業系統及應用程式中被實作</li> <li>3. 能了解並熟悉目前正在使用的各層之協定的設計理念與運作方式。</li> <li>4. 能具備學習新的網路技術與協定之能力</li> </ol>

Data link	Aloha, CSMA, CSMA/CD,	<ol style="list-style-type: none"> <li>1. Issues: 分析此 layer 之問題</li> <li>2. PPP</li> <li>3. From Ethernet to 10G Ethernet, VLAN</li> <li>4. WLAN (WiFi), Bluetooth, UWB, ZigBee, WiMax</li> <li>5. Device driver (network interface driver, interrupt handler)</li> </ol>
Network	Routing functions of network layers and related issues about host addressing, Internet Protocol, etc.	<ol style="list-style-type: none"> <li>1. IPv4, IPv6, ICMPv4, ICMPv6</li> <li>2. DHCP, NAT</li> <li>3. Routing(RIP, OSPF, BGP), Multicast (DVMRP, PIM-SM, SSM, MSDP, MBGP)</li> </ol>
Transport	TCP, UDP, RTSP, RTP, RTCP	<ol style="list-style-type: none"> <li>1. TCP, UDP, STCP, RTP, RTCP</li> <li>2. Performance issues of TCP</li> <li>3. Detail analysis of congestion control of Tahoe, Reno, new Reno, SACK, FACK, Vegas.</li> </ol>
Application	Popular Internet Applications, their operations and protocol standards	<ol style="list-style-type: none"> <li>1. DNS, FTP, SMTP, POP3, IMAP4, HTTP,...</li> <li>2. Web caching issues</li> <li>3. Network management</li> </ol>
Security	Introduction to encryption, authentication, digital signature, key distribution, and network threats.	<ol style="list-style-type: none"> <li>1. Data security (DES, RSA, IKE, IPsec, VPN, SSL, SET) (Frees/Wan)</li> <li>2. Firewall (tis)</li> <li>3. Intrusion detection system (IDS) (snort)</li> </ol>

(3) 計算方法:

以教學目標而言，大學部「計算方法概論」主要是廣泛地介紹各種程式效能分析技巧，認識各類難度不一的問題，以及傳授解決各種問題之基本演算法。藉由此門課，學生可以學會基本的程式效能分析技巧，並能針對各種基本問題設計有效率的演算法。但由於演算法觸及之問題類型相當廣泛，算法本身之難度又有深淺之分。因此我們在大學部的課程，不會觸及到太多較困難之題目(例如、NP-hard 難題)與較複雜之演算法(例如、Linear programming relaxation)。反之，研究所的「計算方法」則會探討各種較困難之題目，介紹較複雜之演算法，並同時教導同學演算法之設計策略。我們在大學部和研究所之計算方法課程皆規劃程式實作之練習，進以強化學生畢業後從事軟體開發所需之設計、分析、與實作能力。課程內容若根據問題的種類與算法的類型來分類，可分為 exact algorithms, randomized algorithms, approximation algorithms, and reduction techniques。如果從這四類來分析，大學部「計算方法概論」與研究所「計算方法」之差異如下表所示：

項目	大學部計算方法概論	研究所計算方法
教學目標	1. 了解演算法效能分析技巧 2. 認識各種問題之難度 3. 學習解決各種問題之基本演算法	1. 能從各種角度分析一個問題之解法並進而了解設計演算法之設計策略 2. 能了解並熟悉目前各種進階演算法之設計理念與其效能分析。 3. 能具備設計進階演算法以解決困難問題之能力
Exact Algorithms	Sorting algorithms, dynamic programming, greedy algorithms, order statistics, elementary graph algorithms.	Knuth-Morris-Pratt algorithm, Boyer-Moore algorithm, network flow, alignment algorithms.
Randomized Algorithms	Randomized selection.	Randomized max-cut, randomized rounding, randomized QuickSort.
Approximation Algorithms	Vertex Cover.	Set cover, linear programming relaxation, simplex method, semi-definite programming relaxation.
Reduction	SAT, clique, vertex cover	Hamiltonian cycle,

Techniques		traveling sales problem, set cover, minimum test collection, minimum dominating set.
------------	--	--

(4) 編譯器:

本系大學部的「編譯器設計」為選修課程，所以我們並不預期所有研究生均修過「編譯器設計」，但我們認為「編譯器」是資工系碩士生該有的核心能力之一，故列為研究所的核心課程。基本上，研究所的「編譯器」課程的每一章的內容均會較大學部的「編譯器設計」深入許多，尤其是最後一章"Code Generation"，差異非常大。此外，研究所的程式專案作業要比大學部的專案龐大許多，大學部的專案通常只處理單一函式的編譯，而研究所的專案則須處理多函式的編譯。

(5) 計算機結構:

「計算機組織」是「計算機結構」在大學部三年級開授的基礎課程。計算機組織的教學目標可以分成教學以及實作兩部分。教學部分的目的以一星期三小時的課進行，目的在於讓學生瞭解電腦硬體的基本概念和設計原理，包含指令集的原理和設計、基本運算單元的原理和設計、電腦效能的意義和如何評估效能、知道 pipelining 和 cache 的設計原理以及對電腦效能的影響、storage 以及 multiprocessor、cluster 等的概念的介紹。實作的部分是配合教學以循序漸進的方式分成三部分進行以 Verilog 完成。第一部份設計 general RISC 的部分指令和基本算數元件，第二部份完成第一部份的 datapath 和 control，第三部份則是將第二部份改進成 multicycle 以提高效能。

相較之下，研究所的「計算機結構」不再以電腦硬體的基本概念和原理的內容為主，而是深入探討改善效能的當代最新技術，以教學配合作業以及考試進行，內容包含 instruction level parallelism (ILP)、以硬體的方式開發 ILP、以軟體的方式開發 ILP、當代最新的 caching 技術、當代最新的 multiprocessor 以及 thread level parallelism (TLP) 等。這門課的值得一提的特色為都是以當代最新的 CPU 為例子如 Intel IA64 和 IBM RS6000 等，使同學更容易感受課本所提到的技術的實用性。

綜合來說，「計算機組織」的重點在於電腦硬體的概念和基本設計原理，是「計算機結構」的先修課程；而「計算機結構」強調的是電腦硬體的前瞻技術的探討，屬於進階課程。